

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»

Факультет комп'ютерних наук і технологій

(повне найменування інституту, назва факультету)

Кафедра програмних засобів

(повне найменування кафедри)

Пояснювальна записка

до дипломного проекту (роботи)

бакалавр

(ступінь вищої освіти)

на тему ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ ДЛЯ ПОРТАТИВНОГО
ЕЛЕКТРОННОГО ПРИСТРОЮ НА ОСНОВІ WI-FI МОДУЛЯ
SOFTWARE FOR A PORTABLE ELECTRONIC DEVICE
BASED ON A WI-FI MODULE

Виконала: студентка 4 курсу, групи КНТ-118

Спеціальності 121 Інженерія програмного

забезпечення

(код і найменування спеціальності)

Освітня програма (спеціалізація)

Інженерія програмного забезпечення

Фоміна М.О.

(прізвище та ініціали)

Керівник Пархоменко А.В.

(прізвище та ініціали)

Рецензент Поляков М.О.

(прізвище та ініціали)

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»
(повне найменування закладу вищої освіти)

Інститут, факультет ІРЕ, ФКНТ
Кафедра програмних засобів
Ступінь вищої освіти бакалавр
Спеціальність 121 Інженерія програмного забезпечення
(код і найменування)
Освітня програма (спеціалізація) Інженерія програмного забезпечення
(назва освітньої програми (спеціалізації))

ЗАТВЕРДЖУЮ

Завідувач кафедри ПЗ, д.т.н, проф.
С.О. Субботін
“ ” 2022 року

З А В Д А Н Н Я
НА ДИПЛОМНИЙ ПРОЄКТ (РОБОТУ) СТУДЕНТА(КИ)

Фоміної Марії Олександрівни
(прізвище, ім'я, по батькові)

1. Тема проєкту (роботи) Програмне забезпечення для портативного електронного пристрою на основі Wi-Fi модуля. Software for a Portable Electronic Device Based on a Wi-Fi Module

керівник проєкту (роботи) Пархоменко Анжеліка Володимирівна, к.т.н., доцент,
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом закладу вищої освіти від “27” квітня 2022 року № 102

2. Строк подання студентом проєкту (роботи) 15 травня 2022 року

3. Вихідні дані до проєкту (роботи) рекомендована література, завдання

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити) 1. Аналіз предметної області. 2. Проєктування програмної системи. 3. Конструювання програмної системи. 4. Керівництво користувача. 5. Керівництво програміста. 6. Текст програми.

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

6. Консультанти розділів проєкту (роботи)

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	прийняв виконане завдання
1-5 Основна частина	Пархоменко А.В., доцент		
Нормоконтроль	Дейнега Л.Ю., ст. викладач		

7. Дата видачі завдання “20” лютого 2022 року.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломного проєкту (роботи)	Строк виконання етапів проєкту (роботи)	Примітка
1	Постановка завдання роботи	1 тиждень	Завдання, ТЗ
2	Аналіз предметної області	2-3 тижні	Розділ 1
3	Проектування програмної системи	4 тиждень	Розділ 2
4	Конструювання програмної системи	5-6 тижні	Розділ 3
5	Тестування та експериментальне дослідження програми	7 тиждень	Розділ 3
6	Оформлення пояснювальної записки та документів до неї. Нормоконтроль та рецензування	8 тиждень	Додатки
7	Захист роботи	9 тиждень	

Студентка

Підпис
(підпис)

Фоміна М.О.
(прізвище та ініціали)

Керівник проєкту (роботи)

Підпис
(підпис)

Пархоменко А.В.
(прізвище та ініціали)

РЕФЕРАТ

Пояснювальна записка до дипломної кваліфікаційної роботи бакалавра: 100 с., 55 рис., 11 табл., 2 дод., 86 джерел.

ПОРТАТИВНИЙ ПРИСТРІЙ, СОНЯЧНА ПАНЕЛЬ, WI-FI МОДУЛЬ, ХМАРНЕ СХОВИЩЕ, КОМП'ЮТЕРНИЙ ЕКСПЕРИМЕНТ.

Об'єкт дослідження – процес передачі даних з портативного електронного пристрою.

Предмет дослідження – протоколи передачі даних та технології живлення портативного пристрою.

Мета роботи – дослідити та розробити програмне забезпечення для енергоефективної передачі даних з портативного пристрою з використанням бездротових технологій та альтернативних джерел живлення.

Матеріали, методи та технічні засоби: структурне та об'єктно-орієнтоване програмування, Wi-Fi модуль ESP8266, монокристалічна кремнієва сонячна панель, вимірювач Keweisi KSW-V20, мова програмування C++, інтегроване середовище розробки Arduino, персональний комп'ютер з процесором Intel Core i7 під управлінням операційної системи Microsoft Windows 10.

Результати. Створено програмне забезпечення для передачі даних від Wi-Fi модуля ESP8266 у хмарне сховище на основі живлення від сонячної панелі або від ПК.

Висновки. Розроблене програмне забезпечення дозволяє мінімізувати енергоспоживання при передачі даних від Wi-Fi модуля ESP8266 у хмарне сховище, що забезпечує тривалу роботу портативного електронного пристрою.

Галузь використання – системи Інтернету речей, домашньої та офісної автоматизації.

ABSTRACT

Explanatory note to the diploma qualifying work of the bachelor: 100 pages, 55 figures, 11 tables, 2 appendixes, 86 sources.

MOBILE DEVICE, SOLAR PANEL, WI-FI MODULE, CLOUD STORAGE, COMPUTER EXPERIMENT.

Object of study is a process of data transmission from mobile electronic device.

Subject of study are communications protocols and technologies for powering mobile device.

The purpose of the work is to explore and create a software system for efficient energy data transmission with using wireless technologies and alternative sources of power.

Materials, methods, and tools: structured and object-oriented programming, Wi-Fi module ESP8266, monocrystalline silicone cell, electronic measuring device Keweisi KSW-V20, programming languages C++, integrated development environments Arduino, personal computer with an Intel Core i7 processor under the control of the Microsoft Windows 10 operating system.

Results. The software for powering Wi-Fi module ESP8266 was developed based on solar power energy as well as on personal computer energy. The software allows transmit specific data to cloud storage.

Conclusions. The developed software provides long work of the mobile electronic device and allows to minimize power consumption while data are transmitted from Wi-Fi module ESP8266 to cloud storage.

The field of use is Internet of Things systems for home and office automation.

ЗМІСТ

	С.
ПЕРЕЛІК СКОРОЧЕНЬ ТА УМОВНИХ ПОЗНАК	9
ВСТУП	10
1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ.....	10
1.1 Класифікація портативних пристроїв	11
1.2 Живлення портативних пристроїв.....	15
1.2.1 Носимі пристрої та альтернативні види енергії.....	18
1.2.2 Сонячна енергія.....	20
1.2.3 Сонячні панелі.....	21
1.3 Засоби вимірювання живлення	24
1.3.1 Keweisi KWS-V20	25
1.3.2 USB UNI-T UT658.....	25
1.3.3 Atorch u96.....	26
1.4 Способи передачі даних	27
1.4.1 Wi-Fi	28
1.4.2 Bluetooth.....	31
1.5 Засоби передачі даних	32
1.5.1 Мікроконтролер ESP8266MOD	34
1.5.2 Мікроконтролер ESP-WROOM-32	35
1.5.3 Мікроконтролер ESP32 – S2-WROOM	35
1.5.4 Мікроконтролер ARM7	36
1.5.5 Мікроконтролер AVR ATmega32	37
1.5.6 Мікроконтролер PIC18F4550	37
1.5.7 Мікроконтролер STM32G030F6P6.....	38
1.5.8 Мікроконтролер STM8	39

	7
1.6 Постановка задач бакалаврської роботи.....	41
2 ПРОЄКТУВАННЯ ПРОГРАМНОЇ СИСТЕМИ	42
2.1 Аналіз вимог	42
2.2 Розробка діаграми прецедентів.....	43
2.3 Розробка архітектури системи	44
2.4 Вибір засобів розробки	45
2.4.1 Інтегроване середовище розробки	45
2.4.2 Хмарне сховище	48
2.5 Висновки	54
3 КОНСТРУЮВАННЯ ПРОГРАМНОЇ СИСТЕМИ	55
3.1 Розробка алгоритму функціонування програми	55
3.2 Опис програмної реалізації	59
3.3 Опис комп'ютерних експериментів	61
3.4 Висновки	65
4 КЕРІВНИЦТВО КОРИСТУВАЧА	67
4.1 Призначення програми	67
4.2 Умови виконання програми	67
4.3 Виконання програми.....	67
4.4 Повідомлення користувачу	69
5 КЕРІВНИЦТВО ПРОГРАМІСТА	70
5.1 Призначення програми	70
5.2 Умови виконання програми	70
5.3 Характеристика програмної системи	70
5.4 Звернення до програми.....	71
5.5 Початкові та вихідні дані	72

	8
5.6 Повідомлення програмісту.....	73
ВИСНОВКИ	74
ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ	75
ДОДАТОК А Текст програми.....	83
ДОДАТОК Б Слайди презентації	92

ПЕРЕЛІК СКОРОЧЕНЬ ТА УМОВНИХ ПОЗНАК

CdTe	– Cadmium telluride;
CES	– Consumer Electronic Show;
CIS	– Copper Indium Selenide;
IoT	– Internet of Things;
LED	– Light-Emitting Diode;
MIMO	– Multiple In, Multiple Out;
OLED	– Organic Light-Emitting Diode;
RST	– Reset;
RTC	– Real Time Clock;
USB	– Universal Serial Bus;
WiMAX	– Worldwide Interoperability for Microwave Access;
WLAN	– Wireless Local Area Network;
ККД	– коефіцієнт корисної дії;
ОЗУ	– оперативна пам'ять;
ОС	– операційна система;
ПК	– персональний комп'ютер;
РК	– рідкокристалічний.

ВСТУП

У сучасному світі все більше обертів набирає популярність портативних та носимих пристроїв. Згідно статистичним даним, тільки у 2021 році було продано близько 533 мільйонів носимих пристроїв [1]. Такі дані кажуть не тільки про прибутковість ринку, але і про перспективність розробок в такий сфері.

Крім того, в останні роки напрямок зеленої енергії становиться не скільки альтернативою на ринку, скільки життєвою необхідністю для майбутнього планети. Такий підхід характеризується оптимізацією здобичі матеріалів, користуванням вже існуючих ресурсів та мінімізацією споживання електроенергії. За останньою статистикою можна побачити, що найпопулярніший напрямок зеленої енергії – сонячна енергія [2].

Не потребує доказів той факт, що світ безумовно рухається у бездротове майбутнє з активним використанням інтернет мереж, які розвиваються з кожним роком, що дозволяє відправляти велику кількість даних за секунди часу.

Для портативних та носимих пристроїв важливим є забезпечення тривалого часу роботи акумуляторної батареї від автономного джерела. Підхід до цієї проблеми можна розглядати як з апаратної, так і з програмної точки зору. Іншими словами, для покращення цього показника важливим є розробка оптимального програмного забезпечення, а тому тема є актуальною.

Мета роботи – дослідити та розробити програмне забезпечення для енергоефективної передачі даних з портативного пристрою з використанням бездротових технологій та альтернативних джерел живлення.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Класифікація портативних пристроїв

Портативний пристрій – це певний вид електричних приладів, які можна легко переносити з собою та які без проблем працюють у бездротовому режимі. Як правило, вони оснащені дисплеєм, мають Wi-Fi та/або Bluetooth модулі, мають модуль камери та дозволяють підключати периферійні пристрої такі як навушники [3].

Портативність пристрою можна розглядати відносно деяких характеристик:

- а) фізичні розміри;
- б) бездротова робота;
- в) мережевий зв'язок [3].

До портативних пристроїв крім мобільних телефонів, планшетів та ноутбуків [3], можна ще віднести ігрові приставки та електронні книги [4]. Крім того, портативні пристрої можуть бути медичного призначення для відстеження стану пацієнта [5].

На основі проведених досліджень, можна виділити наступну класифікацію портативних пристроїв (рис.1.1).



Рисунок 1.1 – Класифікація портативних пристроїв

До носимих пристроїв можна віднести невеликі пристрої, які легко носити на собі або з собою і вільно використовувати на руху. До цих пристроїв можна віднести:

- а) смартфони;
- б) навушники;
- в) медичні пристрої;
- г) розумні годинники.

Перший смартфон з'явився ще у 1994 році. Доступ до інтернету зі смартфона з'явився у 2001 році, а у 2007 році з'явився перший смартфон з сенсорним екраном. На сьогоднішній час у світі вже продано близько 5 мільярдів смартфонів [6]. Сучасні смартфони (рис. 1.2) мають Wi-Fi та Bluetooth модулі, камери, підтримують безліч периферійних пристроїв та які можуть працювати більше доби безперервного використання завдяки потужним акумуляторним батареям.



Рисунок 1.2 – Сучасні смартфони [7]

Перші навушники з'явилися ще у кінці 19 століття, але популярності набули на початку 21 століття. Перші бездротові навушники з'явилися у 2004 році, а у 2017 вони стали доступніші, після того, як компанія Apple спопуляризовала їх на ринку. На сьогоднішній день, бездротові навушники

(рис. 1.3) підтримують обидві технології Bluetooth та Bluetooth LE, що дозволяє їм працювати не тільки з декількома пристроями але й на великих відстанях [8].



Рисунок 1.3 – Навушники [9]

Популярність медичних пристроїв для щоденного використання почала рости тільки нещодавно і у 2020 році ця сфера відчула неймовірний успіх, через відповідні події у світі і досягла вартості 53 мільярди євро. Носимі медичні пристрої (рис. 1.4) дозволяють супроводжувати пацієнта протягом захворювання, дають можливість лікарю підтримувати зв'язок з захворюванням, ставити діагноз та відстежувати перебіг хвороби [5].

Розумні годинники відомі ще з 2014 року, але популярності набули у 2017. Частково, розумні годинники можна віднести і до носимих медичних пристроїв, адже існує багато моделей, які можуть відстежувати показників під час тренування як пульс і рівень кисню та повідомляти про можливі проблеми, але цим їхні функції не обмежуються. Розумні годинники дозволяють використовувати мапи, перекладати мови, показувати погоду та можуть виконувати часткові функції телефону [10] (рис. 1.5).

1.2 Живлення портативних пристроїв

Живлення портативних пристроїв, відбувається за допомогою встановлених в них акумуляторних батарей. Існує велика кількість видів акумуляторних батарей, які використовуються для різних галузей (таблиця 1.1) [13].

Таблиця 1.1 – Класифікація акумуляторних батарей [13]

Назва	Галузь	Напруга елемента, В
1	2	3
Літієво-іонний	Транспорт, комунікаційні мережі, системи сонячної енергії, автономне та резервне електропостачання, мобільні пристрої, електромобілі, електричні інструменти та ін.	3,4 - 4,2
Нікелево-сільовий	Автомобілі, залізничний транспорт, телекомунікації, альтернативна енергетика, системи накопичення енергії	2,58
Нікелево-кадмієвий	Електромобілі, річний та морський транспорт, авіація	1,2 – 1,35
Залізо-нікелевий	Резервне живлення, тягові електротранспорту, ланцюги живлення	1,2
Нікелево-водень	Космос	1,5
Нікелево-метало-гідридний	Електромобілі, дефібрилятори, ракетно-космічна техніка, системи автономного живлення, радіоапаратура, техніка освітлення	1,2 – 1,25
Нікелево-цинковий	Фотоапарати	1,65

Продовження таблиці 1.1

1	2	3
Свинцево-кислотний	Системи резервного живлення, побутова техніка, джерела безперебійного живлення, альтернативні джерела живлення, промисловість	2,11 – 2,17
Срібно-цинковий	Воєнні технології	1,85
Срібно-кадмієвий	Космос, мережа, воєнні технології	1,6

Здебільшого, майже всі портативні пристрої оснащені саме літійовими або літійово-іонними акумуляторами. Проте, літійові акумулятори недешеві, тому що у виготовленні використовується кобальт, який коштує дуже дорого. Крім того, літійові батареї великих розмірів мають високий ризик перегріву. Але, введення ніобію і цирконію збільшує безпеку у експлуатації і, крім всього, надає більшу провідність та енергозбереження, але збільшує виробничу ціну [14].

Принцип роботи акумулятора заснований на зворотності хімічної реакції. Тобто, для заряду акумулятора струм пропускається у напрямку, що зворотній напрямку струму при розряді. На катоді (від'ємному електроді) реакція окиснення замінюється реакцією відновлення, а на аноді (позитивному електроді) реакція відновлення замінюється на реакцію окиснення [15]. Найбільше поширення отримали свинцеві і лужні акумулятори (залізо-нікель та кадмій-нікель), але, крім того, популярні цинк-срібляні, цинк-повітряні і марганцеві [16].

Акумулятори портативних пристроїв мають ті ж самі загальні принципи роботи, що звичайні акумулятори і електрохімічні джерела живлення, як наприклад паливні елементи. Вони складаються з двох електродів, які електрично під'єднанні до активних матеріалів, що занурювані у електроліт з пористим роздільником, який розташований між цими активними матеріалами так, що він запобігає електричний контакт, але дозволяє іонам вільно протікати (рис. 1.6) [17].

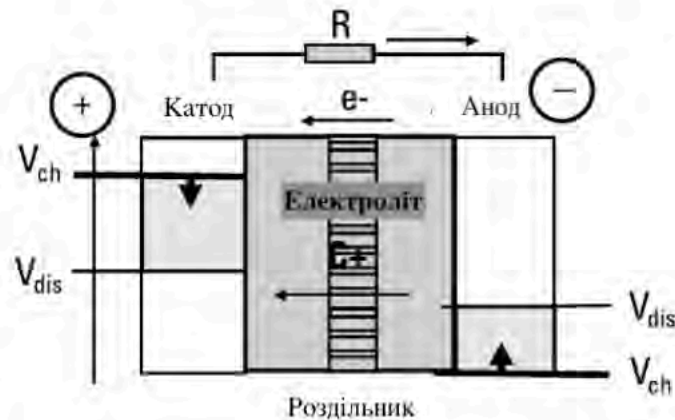


Рисунок 1.6 – Влаштування акумуляторної батареї портативного пристрою [17]

Окрім живлення напругу через електрохімічні реакцію багато компаній намагаються знайти альтернативні джерела живлення акумуляторних батарей для збільшення життя акумулятора, провідності, оптимізації ресурсів і підтримки екологічного напрямку розвитку. Так, компанія Motorola і Freeplay Energy Group у 2002 році створили пристрій Freecharge, що дозволяє заряджати мобільний телефон просто механічно прокрутивши ручку пристрою [18].

Інший підхід до альтернативних джерел живлення представили вчені з Японського національного технічного дослідницького університету. Вони запропонували технологію спиртових батарей для мобільних телефонів [19].

В останній час, популярності набуває технологія абсорбації енергії (Energy Harvesting). Ця технологія полягає у зборі зовнішньої енергії, яка була отримана шляхом застосування різних технологій і конвертації у електричну енергію і збереження її. [20].

Окрім нестандартних підходів, дуже популярним для зарядки портативних пристроїв є технологія сонячних батарей. Цікавим є те, що ця технологія розповсюджена не тільки для побутового використання, але і активно використовується у космосі. Сонячні панелі часто встановлюють на супутники для збору електроенергії [21].

Можна підсумувати, що живлення портативних пристроїв може відбуватись за допомогою різних акумуляторів, які можна заряджати не тільки дротовим шляхом, але і альтернативними екологічними способами.

1.2.1 Носимі пристрої та альтернативні види енергії

Як показали проведені дослідження, на ринку відсутні будь які пристрої для генерації енергії з альтернативних джерел для портативних та носимих пристроїв, окрім електродинамічних ліхтарів та сонячних панелей. Хоча, є дослідження і напрацювання для інших джерел. Так, є розробка недорогого портативного вітряного генератора для зарядки телефону [22].

Електродинамічні ліхтарі працюють за допомогою встановленої динамо машини. Струм виникає у обмотці котушки, яка обертається навколо металевого магнітного сердечника, після чого електричний струм накопичується у акумуляторній батареї [23] (рис. 1.7).



Рисунок 1.7 – Електродинамічний ліхтар [24]

Один з найстаріших популярних механізмів з альтернативним джерелом енергії використовується у механічних годинниках з автопід заводом. На відміну від інших годинників з батареями, такі годинники

не мають потреби заряду акумулятора, адже вони працюють від пружини, яка виступає джерелом енергії. Пружина заводиться ротором, який обертається навколо осі годинника. В результаті під час звичайного руху механізм підтримує потрібний рівень енергії і підзавода [25] (рис. 1.8).



Рисунок 1.8 – Механізм годинника з автопідзаходом [26]

Сонячні зарядні пристрої відрізняються компактністю, невеликою вагою, міцністю, стійкістю к змінам у зовнішньому середовищу і довгим строком роботи. При всьому цьому, вони не потребують техобслуговування і можуть бути розміщенні на рухомих об'єктах [14]. Можна побачити, що останнім часом набули популярності сонячні панелі на зарядних пристроях (рис. 1.9). Вони оснащені портом Micro-USB, що дозволяє заряджати не тільки телефон, а і інші портативні пристрої. Потужність заряду залежить від кількості поєднаних сонячних панелей, типу панелі та ефективності сонячного світла



Рисунок 1.9 – Портативна сонячна панель [27]

Зараз, ця галузь на ринку тільки розвивається, адже для повної зарядки смартфона потрібно декілька годин постійного сонячного світла. Крім того, смартфони з сонячною панеллю на протилежній екрану стороні показали свою неефективність, тому що задня панель часто закрита рукою і сонячне світло для генерації електроенергії просто не може потрапити [28].

Можна зробити висновок, що галузь альтернативних джерел для живлення портативних і носимих пристроїв тільки розвивається і ринку потрібен час для нових рішень.

1.2.2 Сонячна енергія

До альтернативних джерел сонячної енергії відносять сонячні панелі, електростанції, геліосистеми, що використовують енергію світлового потоку, який прямим шляхом потрапляє на встановлені фотоелементи. Самі ці фотоелементи перетворюють сонячну енергію або в електричну, або у термальну для підігріву води [29].

Принцип генерації сонячної енергії відомий ще з 1970х років. Він заснований на конвертації сонячної радіації у електроенергію. Сонячна матерія випромінює неоднорідну електромагнітну радіацію, яка відома як сонячне або теплове випромінювання. Сила потоку крізь одну одиницю

простору, що відома як енергетична світність, залежить від температури, матеріалу та поверхні на яку потрапляє світло. Важливо, що за законами термодинаміки існує певний ліміт сили освітлення який залежить виключно від температури [30].

Використання сонячних батарей невеликої потужності в якості джерела електроживлення – найпопулярніший напрямок у фотоенергетиці. Як правило, в таких системах сонячні батареї використовуються разом з буферними акумуляторами, що забезпечує стабільне енергоживлення в будь який час. Фотоелектрична система, окрім сонячних батарей і акумуляторів, зазвичай містить в своєму складі прибор електронного контролю, який виключає перезаряд акумулятора і його повний розряд [14].

Сонячна енергія має свої плюси і мінуси. Ця енергія безсумнівно і ефективна і екологічна, адже при конвертації енергії відсутні будь які шкідливі викиди у атмосферу. І, не дивлячись на плюси, цей вид енергії нерівномірний і нерегулярно потужний, тому що енергія може вироблятися тільки вдень та за відсутністю похмурої погоди, а у дуже сонячні дні вироблення електроенергії є надмірним, що змушує купувати додаткові акумулятори для збереження її [30].

1.2.3 Сонячні панелі

Сонячні панелі поділяються на два основних типи і п'ять додаткових (рис. 1.10). Все залежить від технології виготовлення основного елементу – фотоелемента. У виробництві кристалічних фотоелементів використовуються хімічні розчини для створення кристалів, які потім обробляються для придання потрібної форми. Тоді як у виробництві тонкоплівкових фотоелементів окрім хімічних розчинів додаються ще металічні пластини для створення електричної провідності [31].

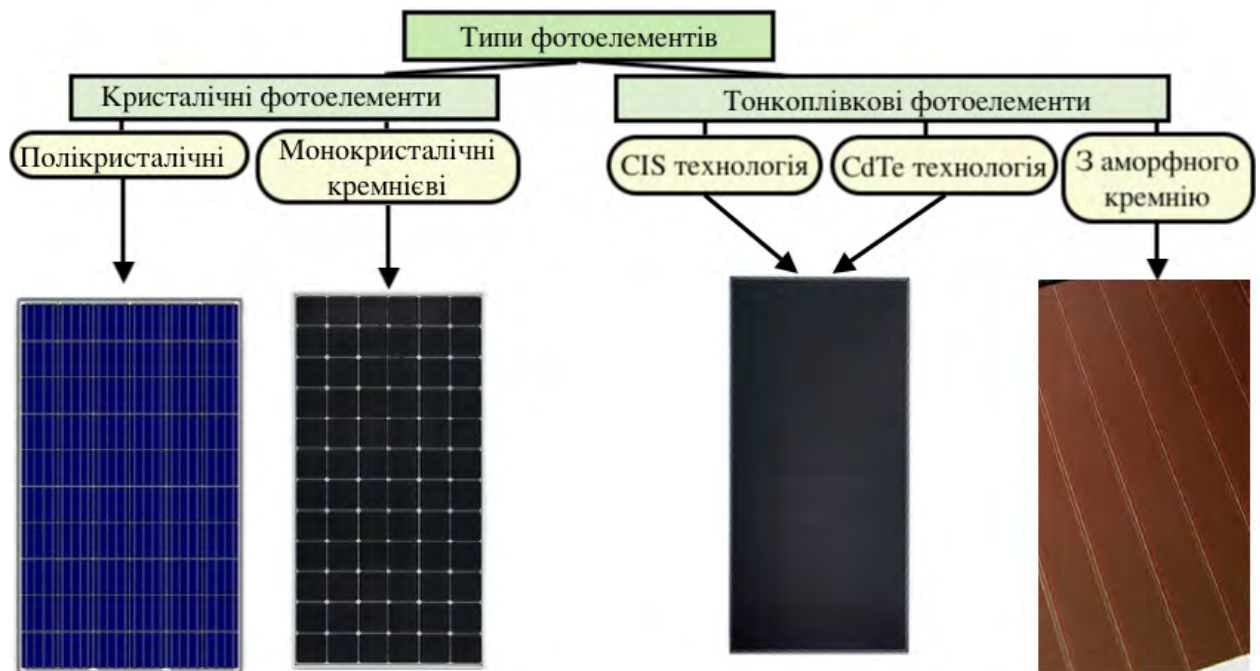


Рисунок 1.10 – Сонячні панелі

Панелі поділяються на:

а) кристалічні:

- 1) полікристалічні;
- 2) монокристалічні кремнієві;

б) тонкоплівкові:

- 1) CIS(Copper Indium Selenide) технологія;
- 2) CdTe (Cadmium telluride) технологія;
- 3) технологія на основі аморфного кремнію [31].

У виробництві кристалічних фотоелементів використовуються хімічні розчини для створення кристалів, які потім механічно обробляються для придання потрібної форми. В той час, як у виробництві тонкоплівкових фотоелементів використовуються інші хімічні розчини і додаються металічні пластини для створення електричної провідності [31].

Особливості виготовлення фотоелементів напряму впливають на коефіцієнт корисної дії (ККД) панелей. ККД для фотоелементів – це ефективність перетворення сонячної енергії в електричну. Це означає, що чим менший ККД, тим більша площа фотоелементів необхідна для забезпечення тієї ж потужності у порівнянні з елементами, у яких ККД має більш високе

значення. Найбільше ККД мають монокристалічні панелі, в той час як найменше ККД мають аморфно-кремнієві панелі. Хоча, серед тонкоплівкових панелей найбільше ККД мають панелі створені за CIS технологією, вони схильні до корозії. Важливо зазначити, що кадмій, який використовується у виробництві CdTe панелей, є токсичним елементом, що викликає гострі та хронічні отруєння. Тому, використані або зіпсовані CdTe сонічні панелі підлягають обов'язковій утилізації, що значно здорожує їхню експлуатацію. Термін служби CIS і CdTe модулів такий же, як і у кристалічних [31].

Сонячні панелі з аморфного кремнію не схильні до корозії і не токсичні, але мають дуже низький ККД і їх активні елементи вигоряють на сонці. Зазвичай в перебігу 6-12 місяців після установки відбувається зниження потужності, потім ці модулі виходять на сталу потужність. Термін служби таких модулів становить близько 10 років [31].

Панелі з кристалічних фотоелементів найчастіше використовуються в будівництві сонячних електростанцій. Зазвичай, термін служби фотомодулів з кристалічних елементів складає 25 років. Через 25 років потужність фотоелементів складе 80% від поточної потужності [31].

Тонкоплівкові сонячні панелі найчастіше застосовуються у фасадних системах і дизайнерських рішеннях. Швидше за все, тонкоплівкові панелі замінять кристалічні, тому що їх виробництво дешевше і менш енергоємне. Тому що, це дуже не ефективна витрата ресурсів, адже на їхнє виробництво яких витрачається більше енергії ніж вони здатні виробити за термін служби [31].

З цього, можна побачити, що найпопулярнішими панелями є полікристалічні.

1.3 Засоби вимірювання живлення

Для вимірювання живлення існує багато різних інструментів різного рівня складності і призначення. Найелементарнішим є прилад під назвою пробник (рис. 1.11). Він має металевий кінець, який дозволяє перевірити є напруга у електричному колі чи ні. Якщо струм є, то загоряється червона лампочка. Додаткових показників він не зчитує [32].



Рисунок 1.11 – Пробник [32]

Більш просунутими є стрілочні прилади як вольтметри чи амперметри (рис. 1.12). Принцип дії таких вимірювачів заснований на магнітах і їхній реакції на струм. Чим більше струм, тим більше обертається магніт, який ще більше натягує пружину яка рухає стрілкою на приборі [32].



Рисунок 1.12– Стрілочний прилад [32]

Найсучаснішими приладами є – USB-тестери (Universal Serial Bus). Не дивлячись на невеликі розміри вони можуть виміряти силу струму, напруження і рівень заряду. У вимірюванні йому допомагають мікросхеми і підключені до них батареї [33] (рис. 1.13).



Рисунок 1.13 – USB-тестер [34]

1.3.1 Keweisi KWS-V20

Одна з найпопулярніших моделей серед USB тестерів є модель Keweisi KWS-V20 (рис. 1.14).

Цей тестер дозволяє вимірювати напругу, силу струму, час заряду-разряду та ємність. Крім того, для зручності він має OLED (Organic Light-Emitting Diode) екран, де відображаються як поточні показники, так і споживання струму за певний час [35].



Рисунок 1.14 – Keweisi KSW-V20 [35]

1.3.2 USB UNI-T UT658

Тестер моделі USB UNI-T UT658 (рис. 1.15), як і попередній, може вимірювати силу струму, напругу, час заряду і розряду та ємність. Управляти цим тестером можна за допомогою однієї кнопки. Має LED (Light-Emitting Diode) екран. Головною відмінною є діапазон вимірювання напруги та ціна [36].



Рисунок 1.15 – Keweisi KSW-V20 [36]

1.3.3 Atorch u96

Іншою популярною моделлю є Atorch u96 (рис. 1.16). Цей тестер дуже зручно відображає одразу багато різних даних, та, на відміну від попередніх, може охоплювати значно більші діапазони вимірювань. Має РК (рідкокристалічний) дисплей [37].



Рисунок 1.16 – Atorch u96 [37]

З таблиці 1.2 можна зробити відстежити, що тестер Atorch u96 має найширший діапазон вимірювань, але, відповідно, має і найбільшу ціну. USB

UNI-T UI658 має майже однакові показники з Keweisi KWS-V20, але коштує набагато більше. Тож, можна зробити висновок, що USB-тестер Keweisi KWS-V20 – найоптимальніший варіант.

Таблиця 1.2 – Порівняння USB-тестерів

Назва	Діапазон вимірювання, В	Діапазон вимірювання, А	Дисплей	Збереження даних	Ціна
Keweisi KWS-V20	3.7 – 20	0.00 – 3.00	OLED екран	>99999 мАч	107 грн
USB UNI-T UT658	3.7 – 10	0.00 – 3.00	LED екран	>9999 мАч	206 грн
Atorch u96	3.6 – 32.5	0.00 – 5.00	РК-дисплей	>99999 мАч	360 грн

1.4 Способи передачі даних

Передачу даних можна розділити на два основних способи:

- а) телефонна мережа;
- б) комп'ютерна мережа [38].

Ці два способи вже розділяються на різні підвиди, які включають в себе інші протоколи передачі даних [38] (рис 1.17).

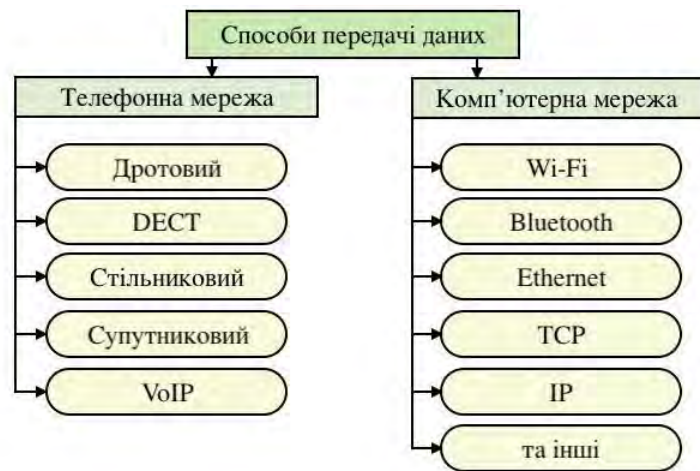


Рисунок 1.17 – Класифікація способів передачі даних

При реалізації передачі даних у комп'ютерній мережі, як і у телефонній мережі, реалізуються різні протоколи, серед яких такі відомі як Wi-Fi та Bluetooth. Доступ до мережі Wi-Fi, станом на 2019 рік, є у 4 мільярдів людей [39] і це дає нам змогу припустити, що на сьогоднішній час це число навіть набагато більше. Крім того, не менш популярною є технологія Wi-Fi, адже продаж пристроїв з Bluetooth модулями досягнув 1.1 мільярда у 2020 [40].

Можна зробити висновок, що мережі передачі даних Wi-Fi і Bluetooth є найпопулярнішими комп'ютерними мережами у світі.

1.4.1 Wi-Fi

Wi-Fi це одна з технологій WLAN (Wireless Local Area Network), або, інакше кажучи це бездротова локальна мережа, до якої, крім Wi-Fi, входить ще WiMAX (Worldwide Interoperability for Microwave Access) [41]. Технологія Wi-Fi відома з 1997, коли стандарт 802.11 був затверджений для загального використання [42]. Стандарт 802.11 дозволяє під'єднуватись до мережі Інтернет бездротовим шляхом. Фактично, стандарт 802.11 це стандарт 802.3 (Ethernet) але без зайвих дротів [43].

Поява WLAN значно облегшило налаштування мереж, оскільки зникла потреба у великій кількості дротів, які підключались до комп'ютерів. З іншого

боку, через особливості технології бездротового зв'язку виросла втрата у швидкості передачі даних [43].

Wi-Fi використовує електромагнітні хвилі для того, щоб передавати дані [42]. Існує близько 9 розповсюджених стандартів, які виходили в різні роки і, через що, мають різну швидкість але все ще використовуються (таблиця 1.3) [44].

Таблиця 1.3 – Стандарти бездротової мережі Wi-Fi [44]

Стандарт	Рік виходу	Частота, ГГц	Швидкість, Мбіт/с	Діапазон роботи, м
IEEE 802.11	1997	2.4	2	Всередині: 20 Зовні: 100
Wi-Fi 1 802.11a	1999	5/3.7	54	Всередині: 35 Зовні: 120/5000
Wi-Fi 2 802.11b	1999	2.4	11	Всередині: 35 Зовні: 120
Wi-Fi 3 802.11g	2003	2.4	54	Всередині: 38 Зовні: 140
Wi-Fi 4 802.11n	2009	2.4/5	600	Всередині: 70 Зовні: 250
Wi-Fi 5 802.11ac	2013	2.4/5	450/1300	Всередині 35
WiGig	2012	60	6.7 Гбіт/с	3.3
HaLow	2016	0.9	347	1000
Wi-Fi 6 802.11ax	2019	2.4/5	450/10.53 Гбіт/с	З'ясовується

Крім, цих відомих існують ще десятки змінених державами стандартів окремо для різних країн [41].

Стандарт 802.11a (або Wi-Fi 1) працює на частоті 5 ГГц і може надсилати дані зі швидкістю 54 Мбіт/с. В той час, як наступний стандарт 802.11b (або Wi-Fi 2) повернувся до частоти 2.4 ГГц, на якій працював

найперший стандарт 802.11, але зі швидкістю, яка майже в два рази більша – 11 Мбіт/с [44].

Наступне оновлення стандарту вийшло у 2003 році – стандарт 802.11g (або Wi-Fi 3). Він, як і попередній стандарт, працює на частоті 2.4 ГГц але має швидкість, яка у 4 рази перебільшує попередню – 54 Мбіт/с. Можливість роботи на частоті 2.4 ГГц дозволяє працювати цьому стандарту на зворотній сумісності з попереднім стандартом [44].

Стандарт 802.11n (Wi-Fi 4) на час виходу у 2009 був революційним, адже він міг працювати одночасно на частотах 2.4 ГГц і 5 ГГц, швидкість передачі даних могла досягати 300 Мбіт/с, мав кращий опір до зовнішнього втручання, мав зворотну сумісність зі всіма попередніми стандартами і, крім цього, це був перший стандарт, який використовував принцип MIMO (Multiple In, Multiple Out) [44]. Принцип MIMO заснований на системі зв'язку з передавальними і приймальними антенами, що розташовані в різних містах. Таке використання дозволяє проводити просторову і часову обробку сигналів і ефективніше використовувати випромінювану передавачем потужність [45].

Наступний стандарт 802.11ac (або Wi-Fi 5) вийшов у 2013 році і хоча не мав революційних оновлень, але, як і попередній, може працювати на частотах 2. ГГц і 5 ГГц зі швидкостями 1300 Мбіт/с і 450 Мбіт/с відповідно [44].

Останній стандарт 802.11ax (або Wi-Fi 6) був представлений компанією Intel відносно недавно – на виставці CES (Consumer Electronic Show) у 2018 році [46], а сертифікований лише 9 лютого 2021 року [47]. Згідно відомим даним з офіційного сайту компанії Intel, новий стандарт являється революційним. Крім високої швидкості у 1700 Мбіт/с, новий стандарт може використовувати не тільки частоти 2.4 ГГц і 5 ГГц але й нову частоту 6 ГГц у оновленні Wi-Fi 6E [48].

Як правило, дані можуть передаватися на двох частотах 2.4 ГГц (802.11b) і 5Ghz (802.11a). Дуже багато років стандарт 802.11b залишався популярним для користувачів Wi-Fi так як продовжував підтримувати нові пристрої і був значно дешевшим за 802.11a. Проте, в останні роки, стандарт 802.11a набирає популярність, так як частота 2.4 ГГц стає переповненою, що

значно сповільнює роботу мережі. Важливо зазначити, що для уникнення перевантаження частот були створенні спеціальні роутери, які можуть працювати одночасно на двох частотах [42].

1.4.2 Bluetooth

Технологія Bluetooth схожа на технологію Wi-Fi з однією відмінністю – сила сигналу в тисячі раз слабкіша і, через що, може подолати тільки дуже маленьку відстань між пристроями [49]. Крім того, через те, що при використанні технології Bluetooth сигнал проходить від одного пристрою на інший і не відправляє дані на окремий сервер, це дозволяє споживати менше енергії і, таким чином, покращує роботу акумуляторів і строк їхнього використання [50].

Існує два стандарти Bluetooth: Bluetooth Classic; Bluetooth Low Energy (LE) [51]. Обидва ці стандарти використовують для роботи частоту 2.4 ГГц, але для роботи частіше обирають Bluetooth LE так як він потребує менше енергії і може бути використаний для Mesh-мереж [51]. У таблиці 1.4 можна побачити порівняння відповідних стандартів.

Таблиця 1.4 – Порівняння стандартів технології Bluetooth [51]

	Bluetooth LE	Bluetooth Classic
Частота	2.4 ГГц	2.4 ГГц
Канали зв'язку	40	79
Спосіб з'єднання	Тільки два прилади, Mesh-мережі	Тільки два прилади
Потужність сигналу	≤ 100 мВт	≤ 100 мВт
Передача даних	≤ 2 Мбіт/с	≤ 3 Мбіт/с

Bluetooth Classic має на 39 каналів більше ніж Bluetooth LE. Самі канали потрібні для зв'язку між пристроями. Коли одні пристрої працюють на одному каналі, які обираються випадково і відомі тільки передавачу і приймачу, інші

пристрої поряд можуть вільно працювати з цією технологією на інших каналах не заважаючи передачі даних [52].

Хоча технологія Bluetooth Classic і може надсилати трохи більше даних ніж Bluetooth LE (≤ 3 Мбіт/с порівнюючи з ≤ 2 Мбіт/с) вона може використовуватись суто між двома пристроями [51].

Можна підсумувати, що обидві стандарти технології Bluetooth мають однакову популярність і відрізняються тільки кількістю пристроїв для одночасної роботи і об'ємами даних для передачі по мережі.

1.5 Засоби передачі даних

Для передачі даних за допомогою технології Wi-Fi або Bluetooth використовуються відповідні модулі або мікроконтролери. Існує багато фірм виробників Wi-Fi модулів але найбільші і найпопулярніші з них це Texas Instruments, Espressif та Microchip [53].

Проте, на основі проведених досліджень Wi-Fi та Bluetooth мікроконтролерів, можна побачити, що на українському ринку лідирують інші три фірми виробники – Espressif, STMicroelectronics та Microchip.

Як і кожна компанія виробник, приведені вище компанії мають багато різних лінійок свого продукту. Компанія Espressif має п'ять актуальних лінійок Wi-Fi та Bluetooth мікроконтролерів: ESP8266, ESP32, ESP32-C, ESP32-S2, ESP32-S3. В самих лінійках мікроконтролери відрізняються комплектацією та розмірами [54]. Компанія STMicroelectronics має основну лінійку Wi-Fi та Bluetooth мікроконтролерів STM32 та STM8 [55]. А фірма Microchip випускає мікроконтролери фірми AVR ATmega та PIC. Для порівняння найпопулярніших та сучасних модулів всіх серій можна виділити таблицю 1.5.

Таблиця 1.5 – Порівняння популярних на українському ринку мікроконтролерів

Назва/характеристика	Напруга, В	Споживання у сплячому режимі	CPU	RAM	Ethernet	USB-Host	Bluetooth	Wi-Fi	Ціна, грн
ESP8266	3.3	20 мА	32-bit, 160 МГц	160 Кб	–	–	–	+	90-150
ESP32	2.3 – 3.6	5 мА	32-bit, 80-240 МГц	520 Кб	+	–	+	+	130-150
ESP32-S2	1.8 – 3.3	20 мА	32-bit, 240 МГц	320 Кб	–	+	–	+	130-180
ARM7	1.6 – 3.6	Немає даних	32-bit, 60 МГц	512 Кб	+	+	–	–	270-300
AVR ATmega32	4.5 – 5.5	Немає даних	8-bit, 16 МГц	32 Кб	–	–	–	–	117
PIC18F4550	4.2– 5.5	Немає даних	8-bit 48 МГц	2к x 8	–	+	–	–	335
STM32G030F6P6	2 – 3.6	Немає даних	32-bit, 64 МГц	8 Кб	–	+	–	–	50-80
STM8L051F3P6	1.6 – 3.6	Немає даних	8-bit, 16 МГц	1к x 8	–	–	–	–	160

1.5.1 Мікроконтролер ESP8266MOD

Серія мікроконтролерів ESP8266 одна з найпопулярніших серед аналогів. Сам мікроконтролер підтримує USB, має багато 20 входів/виходів для підключення до плати. Важливо зазначити, що цей мікроконтролер оснащений Wi-Fi антеною, яка розташована поряд з процесором. Крім того, модуль має індикаторний світлодіод, який мерехтить при підключенні [56] (рис. 1.18). Однією з основних особливостей модулів фірми Espressif є наявність різних сплячих режимів. Мікроконтролер ESP8266MOD має чотири сплячих режиму: Без сну, сон у режимі модему, слабкий сон і глибокий сон [57].

Режим без сну змушує модуль і весь його функціонал працювати повністю. Це, очевидно, неефективний режим. За замовченням завжди працює режим модему але він включається, тільки коли є бездротове підключення до точки доступу. Режим слабого сну працює так само, як і режимі модему, але відключає внутрішній таймер і зупиняє роботу процесору. Режим глибокого сну вимикає абсолютно весь функціонал модулю і залишає увімкненим лише RTC (Real Time Clock) – внутрішній годинник [57].

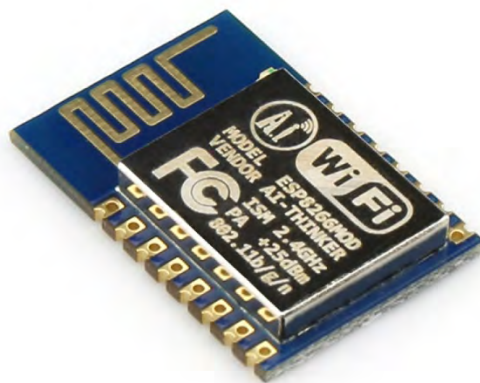


Рисунок 1.18 – Мікроконтролер ESP8266MOD [58]

1.5.2 Мікроконтролер ESP-WROOM-32

Серія мікроконтролерів ESP32 не менш популярна ніж попередня серія ESP8266. Ці мікроконтролери так само оснащені Wi-Fi антенами і додатково підтримують Bluetooth v4.2. ESP-WROOM-32 має 38 входів/виходів, широкий діапазон температур від -40 до +125, дозволяє підключати Flash-пам'ять і, як інші модулі фірми Espressif має багато режимів сну, проте відрізняється найнижчим споживанням серед інших модулів. [56] (рис. 1.19).



Рисунок 1.19 – Мікроконтролер ESP-WROOM-32 [59]

1.5.3 Мікроконтролер ESP32 – S2-WROOM

Серія ESP32-S2 – покращена попередня серія ESP-32. Головні відміни цієї версії від попередньої це відсутність підтримки Bluetooth та Ethernet. Проте, сам процесор потужніше, а зміни у архітектурі дозволяють знизити споживання енергії і підвищити працездатність [56] (рис.1.20).



Рисунок 1.20 – Мікроконтролер ESP-S2-WROOM [60]

1.5.4 Мікроконтролер ARM7

Компанія ARM Limited виготовляє мікроконтролери ARM з 1990х років. Це одна з найпопулярніших серії 16-bit/32-bit, яка завоювала велику частину ринку ще у середині 2000х. [61].

Через особливості архітектури, ядро не може напряму працювати з пам'яттю. Замість цього, всі операції повинні бути виконані у регістрах. ARM7 має 37 регістри – 31 загального призначення і 6 регістри стану. Крім того, процесор має температурні сенсори для контролю нагріву процесору [61] (рис. 1.21).



Рисунок 1.21 –Мікроконтролер ARM7 [62]

1.5.5 Мікроконтролер AVR ATmega32

AVR ATmega32 це високопродуктивні 8-бітні мікроконтролери з дуже низьким споживанням енергії. Підтримує 10000 циклів перепрограмування, має 32 входи/виходи та внутрішню систему для програмування і при цьому потребує лише 4.5-5.5 В електроенергії. Крім того, при виключеному режимі, процесор зберігає дані всіх регістрів, а у сплячому режимі вмикає режим шумозаглушення [63] (рис. 1.22).

1.5.6 Мікроконтролер PIC18F4550

Мікроконтролер PIC18F4550 має високу обчислювальну провідність і програмну flash пам'ять. Контролер підтримує USB 2.0, має 35 входів/виходів і оснащений 8-bit процесором. Через особливості архітектури і конструювання цей мікроконтролер – чудовий вибір для розробок та додатків, що чутливі до живлення [64] (рис.1.23).



Рисунок 1.22 –Мікроконтролер ARV ATmega32 [65]



Рисунок 1.23 –Мікроконтролер PIC18F4550 [66]

1.5.7 Мікроконтролер STM32G030F6P6

Серія мікроконтролерів STM32 одна з найпопулярніших і нараховує десятки різних версій продукту. STM32G030F6P6 один з останніх мікроконтролерів цієї серії. Мікроконтролер оснащений потужним 32-bit

процесором, flash-пам'яттю, температурним сенсором, що дозволяє йому працювати від -40 до +86 та має 20 входів/виходів [67] (рис. 1.24).

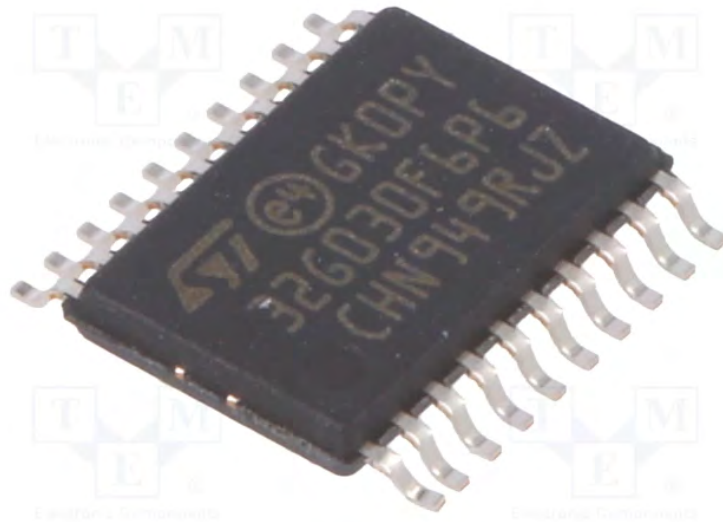


Рисунок 1.24 –Мікроконтролер STM32G030F6P6 [68]

1.5.8 Мікроконтролер STM8

Мікроконтролери серії STM8L – це мікроконтролери загального значення. Модуль має 18 входів/виходів, 8-bit архітектуру процесору і встановленні таймери. Особливості архітектури цих мікроконтролерів дозволяє максимально мінімізувати живлення. Це досягається за рахунок додаткових режимів зниженого живлення і зменшення мінімальної напруги до 1,65 В [69] (рис.1.25).

З отриманих даних, можна провести аналіз. Кожна лінійка мікроконтролерів має безліч рішень на будь який випадок але всіх їх об'єднують певні особливості.

Мікроконтролери серії ARM вважаються одними з найпотужніших на ринку через особливості архітектури і можливості програмування на різних платформах, але їх майже неможливо купити окремо, адже вони продаються або у платах, або в інших пристроях.



Рисунок 1.25 – Мікроконтролер STM8L051F3P6 [70]

PIC, AVR та STM8 випускають мікроконтролери з 8-bit архітектурою процесору, що дуже зменшує область застосування але чудово підходить для проєктів з невеликими розрахунками. STM32 має не тільки потужний процесор, але й відрізняється своєю ціною доступністю на українському ринку та підтримкою, адже велика кількість проєктів робиться саме на базі STM32. Дуже важливо значний недолік, адже у всіх них відсутні встановлені Wi-Fi або Bluetooth модулі, їх треба купувати окремо і встановлювати самостійно.

Єдині мікроконтролери, які мають вже встановлені Wi-Fi антени це мікроконтролери фірми Espressif. Для порівняння, серія ESP32-S2 має майже однакові характеристики з серією ESP8266 і виграє лише потужністю процесору, але їх майже неможливо достати на українському ринку і потрібно замовляти з закордону, що дуже впливає ціну. В той же час, мікроконтролери серії ESP32 не тільки новіші, але й мають найкращі характеристики процесору, оперативної пам'яті і найменше споживають у сплячому режимі. З іншого боку, вони найдорожчі серед всіх і за ціну одного модуля ESP32 можна купити два модуля ESP8266.

Попри те, що ESP8266 має велике споживання у сплячому режимі і найменше оперативної пам'яті, він коштує найдешевше, доступний на українському ринку, а через невелику потужність процесора, модуль буде споживати менше електричного струму.

1.6 Постановка задач бакалаврської роботи

Як показали проведені дослідження, ринок портативних пристроїв швидко розвивається та потребує використання тривалого живлення на основі альтернативних джерел енергії.

В результаті проведеного аналізу було вирішено використовувати Wi-Fi модуль на основі мікроконтролера ESP8266, що відрізняється наявністю встановленої антени та доступністю на ринку. До переваг обраної монокристалічної сонячної панелі можна віднести найбільший ККД, тривалий термін роботи, не токсичність, доступність і безпеку у використанні. Вимірювач живлення й струму Keweisi KSW-V20 забезпечує необхідну точність вимірювання, є зручним у використанні завдяки невеликому об'єму.

Мета роботи – дослідити та розробити програмне забезпечення для енергоефективної передачі даних з портативного пристрою з використанням бездротових технологій та альтернативних джерел живлення.

Задачі роботи:

- виконати аналіз вимог до програмного забезпечення;
- розробити архітектуру системи та обрати інструментарій розробки;
- виконати програмну реалізацію;
- провести серію експериментів та виконати оптимізацію програмного коду;
- розробити програмну документацію.

2 ПРОЄКТУВАННЯ ПРОГРАМНОЇ СИСТЕМИ

2.1 Аналіз вимог

Вимоги до проєкту можна розділити на дві групи – функціональні та нефункціональні.

До функціональних вимог можна віднести:

- бездротова передача даних;
- використання альтернативних джерел живлення;
- зміна джерела живлення;
- збереження даних роботи системи;
- зміна програмних налаштувань.

До нефункціональних вимог можна віднести

- забезпечення мінімального споживання електроенергії;
- структурування результатів експериментів;
- аналіз результатів роботи;
- дослідження споживання електроенергії.

Для забезпечення функціональних вимог були обрані наступні способи реалізації. Для здійснення бездротової передачі даних була обрана саме Wi-Fi мережа. Після розгляду інструментів для передачі даних по мережі Wi-Fi було вирішено обрати відповідний мікроконтролер. Через широкий ринок мікроконтролерів, був проведений аналіз доступних засобів і вибір був зроблений на користь мікроконтролеру з встановленою Wi-Fi антеною – ESP8266.

Після дослідження галузі альтернативного живлення та ринку відповідних рішень, було вирішено обрати сонячні панелі. Серед всіх типів сонячних панелей найкращим варіантом для реалізації обраної задачі стала монокристалічна кремнієва панель.

Для забезпечення мінімального споживання електроенергії окрім обраних рішень відповідно мікроконтролера і альтернативного джерела живлення, велика увага приділена саме програмної реалізації.

Задля фіксації результатів роботи програми було вирішено структурувати результуючі значення протягом експериментів у вигляді таблиць з метою подальшого аналізу.

Для реалізації нефункціональних вимог по структуризації даних було вирішено використовувати сервіс Thinger.io.

2.2 Розробка діаграми прецедентів

Після аналізу предметної області була розроблена діаграма прецедентів, яка представляє собою основні можливості системи (рис. 2.1).

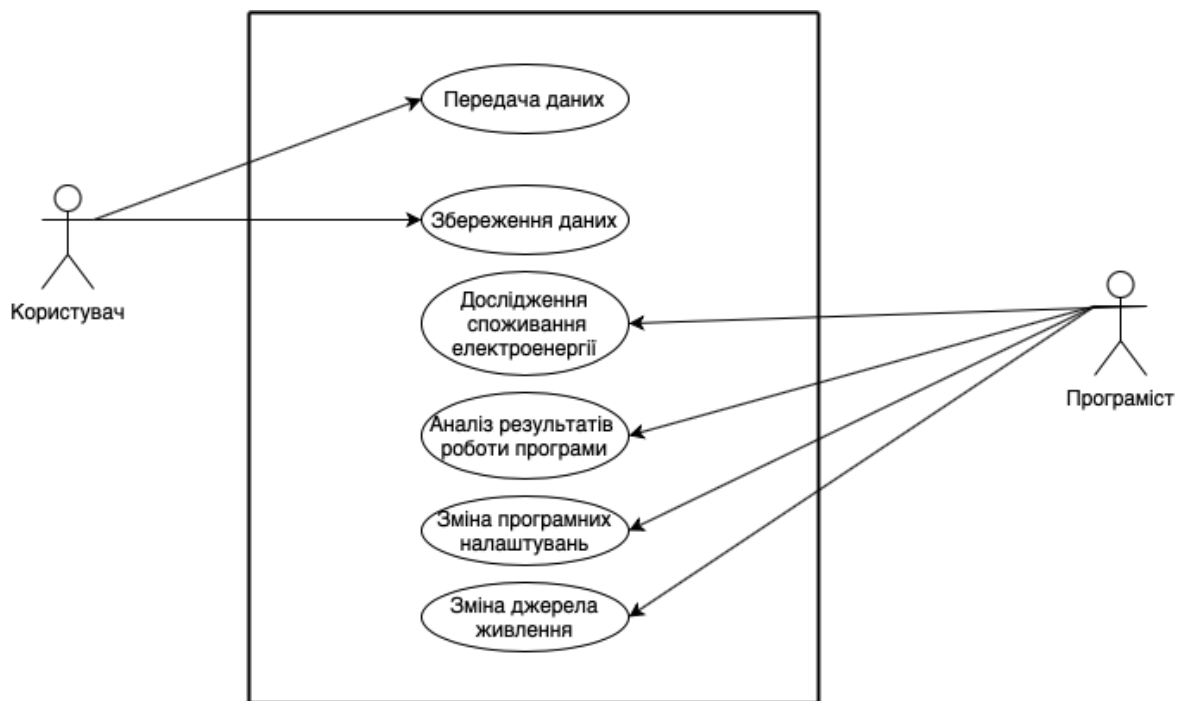


Рисунок 2.1 – Діаграма прецедентів

Передача даних включає в себе можливості відправлення обраних даних користувачем.

Збереження даних включає в себе збереження користувачем даних, отриманих з сервісу Thingerio.

Дослідження споживання електроенергії включає в себе виміри, проведені програмістом, використовуючи тестер Keweisi KSW-V20.

Аналіз результатів роботи програми включає в себе оцінку роботи системи, дані з сервісу Thingertio та стан роботи Wi-Fi модуля.

Зміна програмних налаштувань включає в себе форматування коду програми задля необхідних змін.

Зміна джерела живлення включає в себе фізичну заміну пристрою для живлення системи.

2.3 Розробка архітектури системи

Архітектура системи включає в себе не тільки програмну реалізацію, а також і особливості налаштування системи, як живлення. Вибір живлення системи – найважливіший компонент, адже від цього напряму залежить як саме буде працювати Wi-Fi модуль: від комп'ютера або від сонячної панелі. Крім того, архітектура включає в себе програмний код, який завантажується на Wi-Fi модуль і базу даних хмарного сховища, куди відправляються і зберігаються дані Wi-Fi модуля.

Архітектура системи не має зайвих компонентів, які заважали б проведенню експериментів, і, крім того, відповідає всім цілям дипломного проєкту для успішної розробки системи (рис. 2.2).

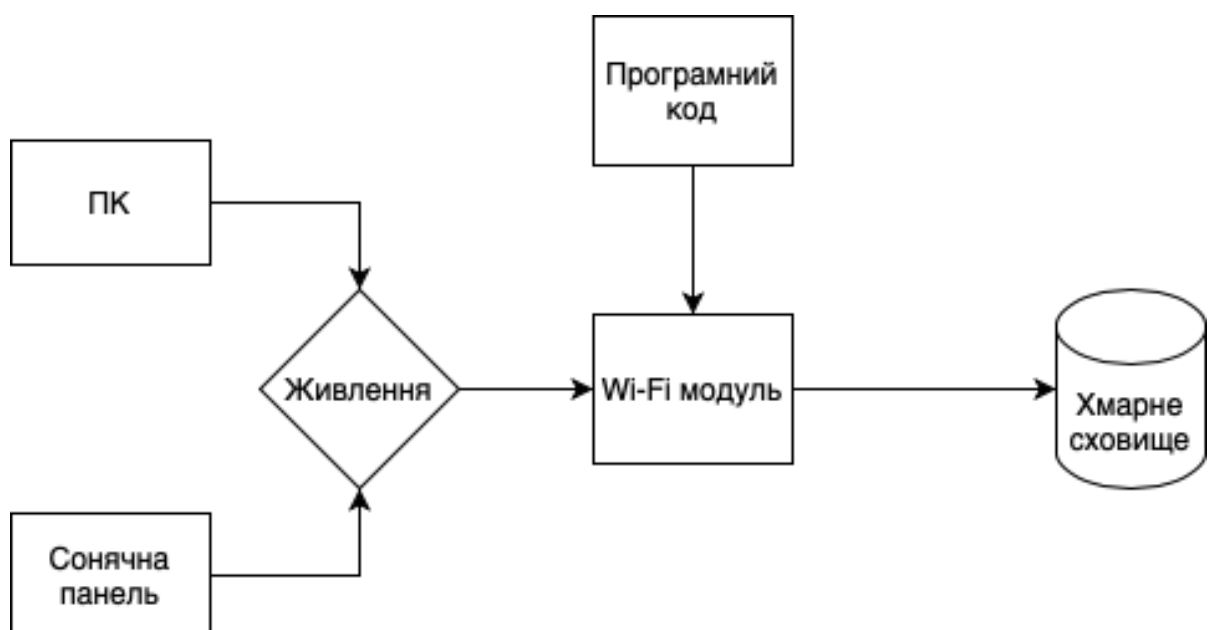


Рисунок 2.2 – Архітектура системи

2.4 Вибір засобів розробки

Для розробки системи необхідно інтегроване середовище розробки, яке має можливість працювати з мікроконтролерами та модулями. Тобто, для порівняльного аналізу інструментарію необхідно звертати увагу саме на сервіси, які можуть працювати з елементами систем розумного будинку, або відомі як IoT (Internet of Things).

2.4.1 Інтегроване середовище розробки

Visual Micro від компанії Microsoft є одним з найпопулярніших середовищ розробки. Фактично, Visual Micro – це плагін Visual Studio для роботи з платами Arduino. В той час, як Visual Studio відносно безкоштовне середовище, Visual Micro має безкоштовний період у 45 днів, після чого необхідно платити ціну в залежності від потреб розробника. Однією з основних переваг середовища Visual Micro є можливість поєднувати інструменти Visual Studio. Крім того, унікальною є опція бездротового налагодження програми, зокрема для мікроконтролерів з Wi-Fi антенами [71] (рис. 2.3). В залежності від кількості встановлених бібліотек, розмір середовища в середньому складає 1 Гб. Але, для встановлення плагіна для роботи з мікроконтролерами необхідно обов'язково встановити середовище Visual Studio та мову C/C++, на якій виконується розробка. Згідно інформації на офіційному сайті, потрібно мінімум 800 Мб вільного місця, але, в середньому, інструментарій для розробки буде займати близько 20 Гб [72]. З цього можна зробити висновок, що для встановлення потрібно мінімум 20 Гб. Visual Micro працює з версії Visual Studio 2019 [71]. Мінімальні потреби для встановлення плагіну Visual Micro відповідають мінімальним потребам для встановлення середовища Visual Studio 2019, а саме: ОС (Операційна система) Windows 7; процесор 1.8 ГГц; ОЗУ 2 Гб [72].

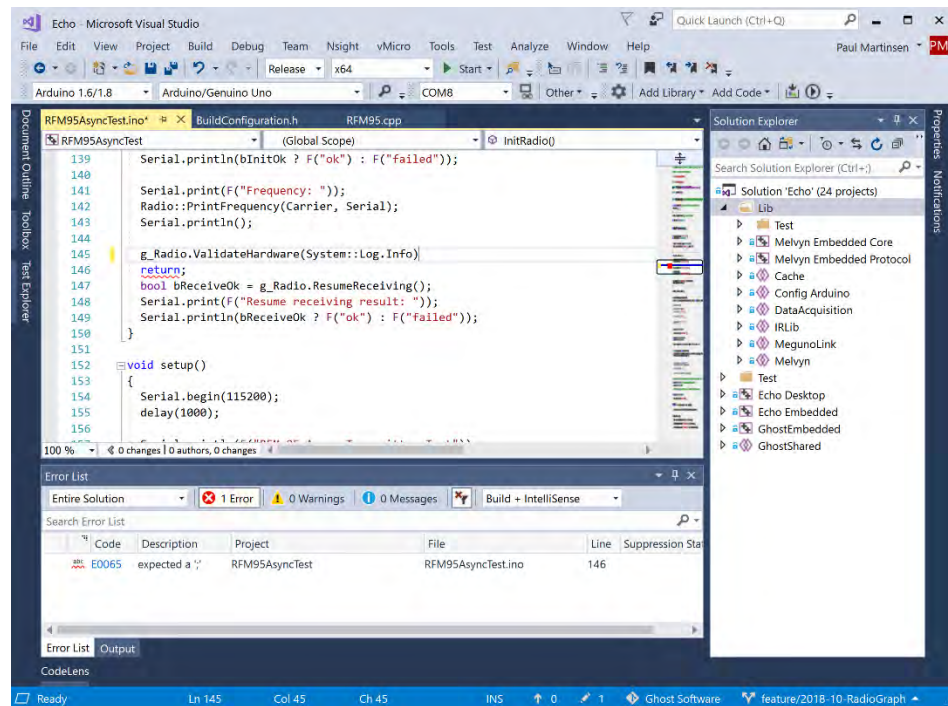


Рисунок 2.3 – Інтерфейс Visual Micro [73]

Platformio – не менш популярне середовище розробки українського виробництва. Як і Visual Micro, фактично, є плагіном для іншого продукту компанії Microsoft для середовища Visual Code. Platformio розроблено на мові Python, а тому може працювати без будь якої залежності від додаткових плагінів. Однією з переваг є кросплатформеність платформи і можливість встановити її на трьох найпопулярніших операційних системах. Крім того, платформа повністю безкоштовна для особистого використання [74] (рис. 2.4).

Важливо зазначити, що для роботи у середовищі Platformio необхідним є встановлення продукту Visual Code компанії Microsoft. Крім того, необхідним буде встановити окремо мову Python версії 3.6 і вище.

Мінімальні потреби Platformio відповідають мінімальним потребам платформи Visual code, а саме: ОС Windows 8.0, ОС Mac OS 10.11, ОС Linux 7; процесор 1.6 ГГц; ОЗУ (Оперативна пам'ять) 1 ГБ [75].

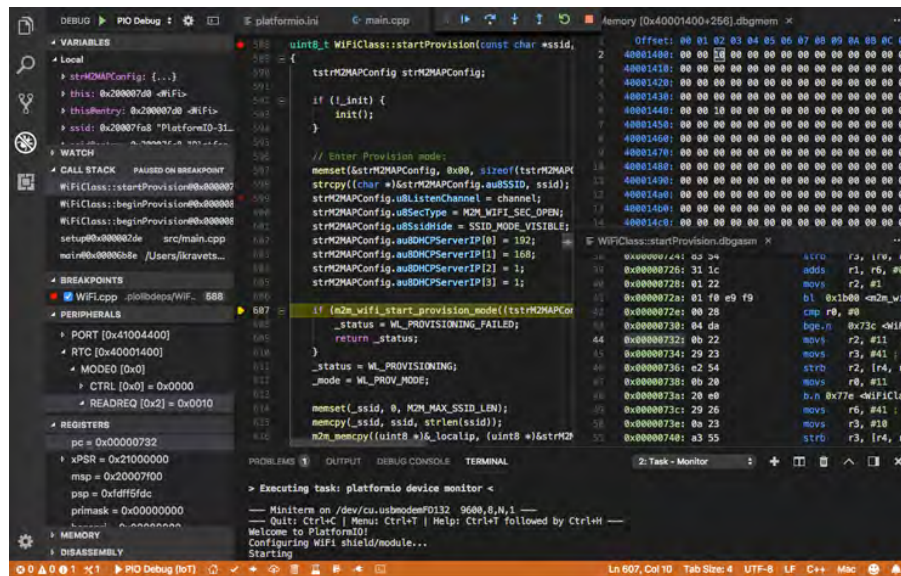


Рисунок 2.4 – Інтерфейс Platformio [74]

Arduino є найпопулярнішим середовищем для розробки систем розумного дому, до яких входить програмування мікроконтролерів. Крім того, система повністю безкоштовна і не потребує додаткових витрат [76] (рис. 2.5).



Рисунок 2.5 – Інтерфейс Arduino

Arduino є окремим середовищем розробки і не потребує встановлення додаткових допоміжних програм і може бути встановлено з офіційного сайту.

Крім того, особливістю сервісу є можливість програмувати через офіційний сайт. Важливо зазначити, що у цього середовища розробки є відкритий програмний код, що дозволяє розробляти свої бібліотеки для роботи з різними мікроконтролерами і модулями.

Мінімальні потреби системи Arduino є: ОС Windows 7.0, ОС Mac OS 10.8.3, ОС Linux 5.5; процесор 1.3 ГГц; ОЗУ 256 Мб [77].

Аналіз середовищ розробки можна побачити на таблиці 2.1 .

Таблиця 2.1 – Порівняльний аналіз середовищ розробки

Платформа \ характеристика	ОС	Процесор, ГГц	ОЗУ, ГБ	Розмір, ГБ	Ціна	Необхідність встановлення додаткових середовищ
Visual Micro	Windows	1.8	2	20	12 \$	так
Platformio	Windows, Mac, Linux	1.6	1	500 мб	Безкоштовно	так
Arduino	Windows, Mac, Linux	1.3	256	540 мб	Безкоштовно	ні

Таким чином, можна зробити висновок, що для розробки дипломного проєкту середовище розробки Arduino підходить краще всього, адже воно відповідає всім потребам для розробки: безкоштовне, не потребує встановлення додаткових програм або плагінів і займає небагато місця.

2.4.2 Хмарне сховище

Для фіксації роботи розроблювальної системи необхідним є вибір хмарного сховища. Особлива увага приділена хмарним сховищам, які можуть працювати з системами IoT, в особливості з Wi-Fi модулями.

Datadog – один з найпотужніших хмарних сервісів для роботи з даними. Сервіс пропонує різні підходи аналізу всіх можливих розроблювальних

систем, як до баз даних, так і до штучних мереж. Ціна продукту залежить від потреб користувача, так як для різних підходів аналізу ціна різна. Сервіс пропонує безкоштовний пробний період з обмеженими опціями [78]. (рис. 2.6).

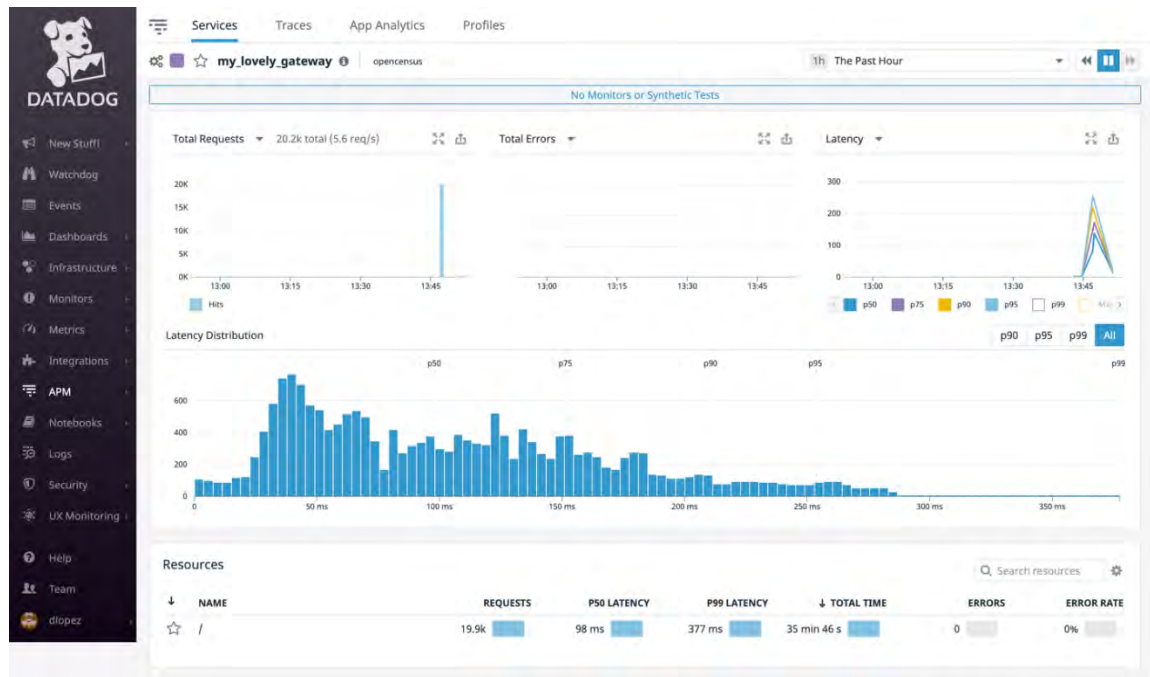


Рисунок 2.6 – Інтерфейс Datadog [78]

Datadog підтримує багато мов програмування так само, як і інших додатків і систем. Сервіс має авторську бібліотеку для підтримки Arduino[79].

Для роботи сервісу необхідно встановити його агент відповідно до платформи та мови, яку використовує користувач. Для кожної мови програмування сервіс пропонує індивідуальні вимоги. Для роботи з мовою C++ необхідною умовою є встановлення Visual Studio 2019, Git та Vcpkg [80]. Агент потрібно встановлювати через консоль, що може викликати складності у більшості користувачів і привести до ряду помилок.

Google Cloud – хмарне сховище компанії Google для роботи з великими об'ємами даних. Так само, як і Datadog, сервіс пропонує низку алгоритмів аналізу даних, як для роботи штучного інтелекту, так і для аналізу звичайних даних. Для використання сервісу можна оформити безкоштовне використання

з обмеженнями у роботі з інструментами сервісу, а повна ціна складає 300\$ (рис. 2.7) [81].

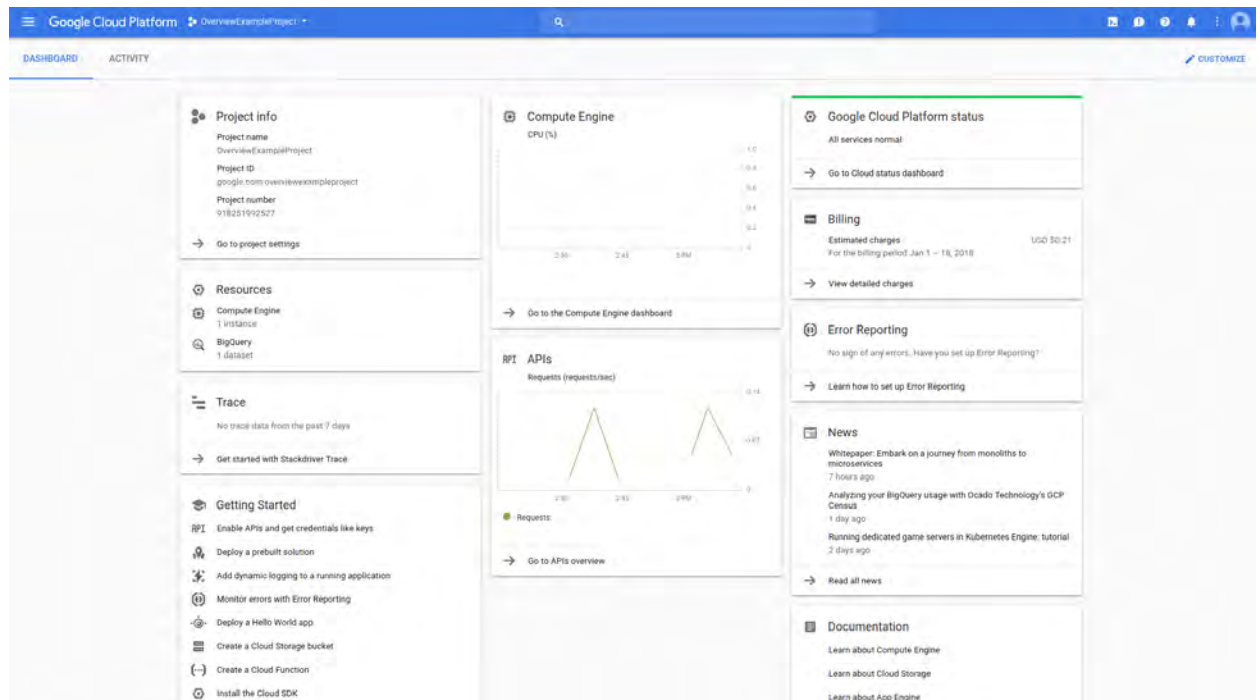


Рисунок 2.7 – Інтерфейс Google Cloud [82]

Для роботи з сервісом потрібно зареєструватися і встановити систему на робочий комп'ютер. Агент для встановлення можна скачати з офіційного сайту або встановити систему через консоль. Google Cloud займає 110 мб. Індивідуально, відповідно до потреб адміністратора, сервіс може займати до 50 гб.

Для роботи з Wi-Fi модулями існує відповідна бібліотека. Проте, вона є не офіційною і, крім того, має обмеження у користуванні, а тому підтримує тільки чотири Wi-Fi модулі ESP32, ESP8266, MKR1000 та WiFi1010 [83].

Thingier.io – найвідоміший хмарний сервіс для роботи з IoT технологіями, в особливості з Wi-Fi модулями. Сервіс пропонує зручну реєстрацію кожного мікроконтролера, відстеження інформації про них, організацію отриманих даних по окремим папкам, інструменти для роботи з даними, як візуалізація через графіки, та пропонує індивідуальну статистику для користувачі [84] (рис. 2.8).

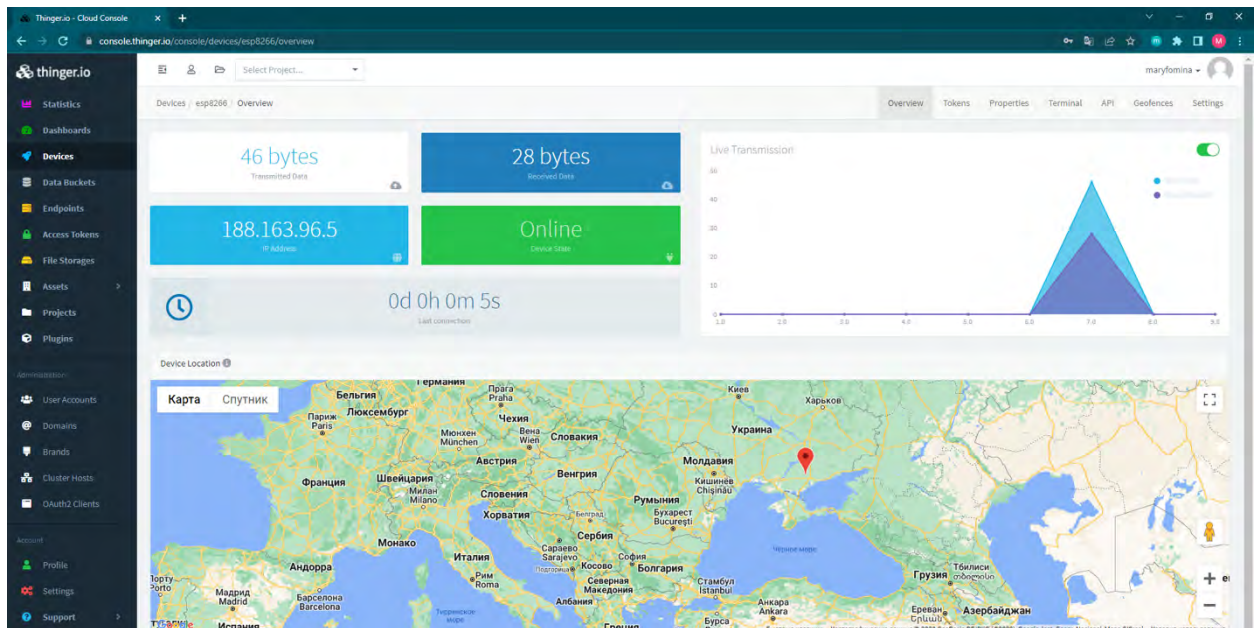


Рисунок 2.8– Інтерфейс Thingier.io

Для роботи з Arduino є офіційні бібліотеки від компанії і для кожного модуля вона розроблена індивідуально.

Однією з особливостей сервісу є те, що він не потребує додаткових програм для роботи та фізичного встановлення на робочий комп'ютер. Щоб почати працювати з цим хмарним сховищем достатньо реєстрації на сервісі і введення потрібних даних у код програми для зв'язку.

ThingSpeak – платформа, яка була створена виключно для роботи з IoT технологіями. Так само, як і Thingier.io, платформа може зберігати дані, аналізувати їх, та працювати з ними. Для більш зручної роботи з результатами програми сервіс надає низку віджетів для зручної роботи. Сервіс дозволяє працювати з такими платформами як Arduino, MATLAB, Raspberry Pi, ESP8266 і ESP32 та інші. Крім того, сервіс дозволяє публікувати результати своїх експериментів та переглядати роботи інших [85] (рис 2.9).



Рисунок 2.9 – Інтерфейс ThingSpeak [86]

Для роботи з системою не потрібно встановлювати додаткових плагінів або сервісів на робочий пристрій. Так само, як і у Thinger.io, достатньо підключити відповідну бібліотеку, зареєструвати пристрій і ввести потрібні дані у код.

Сервіс безкоштовний для персонального використання і має всі потрібні функції, тільки у обмеженому варіанті. А ціна за платну версію не постійна і варіюється від частоти відправки даних. Чим частіше відправляються дані – тим більше ціна.

Огляд всіх хмарних сервісів можна побачити у таблиці 2.2.

З таблиці можна зробити висновок, що сервіс Thinger.io найкраще підходить для поставлених задач. Не дивлячись на те, що Thinger.io і ThingSpeak дуже схожі по характеристикам, одна з головних відмін – різна програмна реалізація. Коли ThingSpeak орієнтований на роботу з даними, їхню обробку і структурування, Thinger.io надає можливість переглядати і відстежувати детально кожне надіслане повідомлення на сервіс у папках «bucket».

Таблиця 2.2 – Порівняння хмарних сервісів

Платформа \ характеристика	Можливість підключати напряму ESP8266	Встановлення додаткових програм	Обробка даних у сховищі	Ціна	Віджет	Детальний перегляд даних
ThingSpeak	+	–	–	Безкоштовно	+	–
Thingier.io	+	–	+	Безкоштовно	+	+
Google Cloud	–	+	+	Пробний період, 300\$	+	+
Datadog	–	+	+	Пробний період, 15\$	+	+

2.5 Висновки

В рамках проведених досліджень були розроблені діаграма прецедентів, архітектура системи та функціональні і нефункціональні вимоги. Крім того, можна зробити висновки, що найкращим вибором для розробки програмного забезпечення на основі поставлених задач до дипломного проєкту будуть інтегроване середовище розробки Arduino та хмарне сховище для Thinger.io.

3 КОНСТРУЮВАННЯ ПРОГРАМНОЇ СИСТЕМИ

3.1 Розробка алгоритму функціонування програми

В основу розробленої системи входить основний алгоритм проведення обчислювальних експериментів.

Першим і найважливішим кроком є перевірка підключення до Wi-Fi мережі. Неможливо почати виконувати обчислювальні експерименти доки не буде налагоджений Wi-Fi зв'язок. Після налаштування бездротової мережі можна починати налагодження системи під необхідні потреби. Спочатку, потрібно обрати інтервал часу – як часто будуть надсилатися данні до хмарного сховища. Другим кроком буде вибір необхідного режиму роботи Wi-Fi мікроконтролеру – Modem Mode або DeepSleep Mode. Від цього напряму буде залежати час роботи мікроконтролеру. Третім кроком буде вже вимірювання живлення за допомогою відповідного пристрою, що фіксує не тільки скільки струму було використано, але і час роботи мікроконтролеру. Останнім, але не менш важливим кроком є фіксування вимірів. Після того, як експеримент був проведений користувач повинен вирішити, чи необхідні йому повторні експерименти з такими ж самими характеристиками, чи він бажає їх змінити або навіть закінчити роботу з системою. Ілюстрацію алгоритму розроблювальної системи можна побачити на рис. 3.1.

Крок 1. Початок роботи.

Крок 2. Перевірка роботи інтернет зв'язку.

Крок 3. Вибір інтервалу часу.

Крок 4. Вибір режиму Wi-Fi мікроконтролера.

Крок 5. Вибір необхідних повідомлень у консоль.

Крок 6. Вимір живлення.

Крок 7. Фіксація результатів.

Крок 8. Чи необхідні додаткові виміри.

Крок 9. Кінець роботи



Рисунок 3.1 – Алгоритм роботи системи

Під час проведення експериментів дуже важливим є пильне спостереження за експериментом, адже існує вірогідність незалежного відключення від мережі Wi-Fi або зникнення живлення, адже в цій серії експериментів мікроконтролер працює напряду від комп'ютеру.

Для проведення серії експериментів с сонячною панеллю, алгоритм системи майже не змінюється, лише додаються кроки перевірки умов, потрібних для проведення експериментів і кроки налаштування сонячної панелі (рис 3.2).

Крок 1. Початок роботи.

Крок 2. Перевірка наявності сонячного світла.

Крок 3. Перевірка роботи інтернет зв'язку.

Крок 4. Вибір інтервалу часу.

Крок 5. Вибір режиму Wi-Fi мікроконтролера.

Крок 6. Вибір необхідних повідомлень у консоль.

Крок 7. Вимір живлення.

Крок 8. Фіксація результатів.

Крок 9. Вирішення чи необхідні додаткові виміри.

Крок 10. Кінець роботи.

Для проведення досліджень з сонячною панеллю важливим є саме перевірка достатніх умов для вдалих експериментів. Сонячну панель потрібно встановити під належним кутом, щоб сонячне світло потрапляло на неї. Крім того, потрібно впевнитись, що погода на вулиці і час доби дозволяють провести експеримент, який буде тривати мінімум годину. Якщо ця умова не виконується – потрібно перевіряти це до тих пір, доки не буде достатньо сонячного світла. Після цього, можна виконувати алгоритм, який майже не відмінюється від алгоритму перших трьох серії експериментів.



Рисунок 3.2 – Алгоритм роботи системи з сонячною панеллю

3.2 Опис програмної реалізації

Основною бібліотекою, яка використовувалась для програмної реалізації була бібліотека `ThingESP8266.h`. Ця бібліотека допомагає підключити Wi-Fi мікроконтроллер ESP8266 до сервісу `Thingier.io`.

Для того, щоб його правильно підключити, потрібно об'явити на початку коду всі необхідні для цього дані. Для цього використовується команда `define` і назва змінної, до якої потрібно внести дані. Всі зміни можна побачити у таблиці 3.1

Таблиця. 3.1 – Змінні у програмі

Назва змінної	Пояснення
<code>USERNAME</code>	Ім'я на сервісі <code>Thingier.io</code>
<code>DEVICE_ID</code>	Назва пристрою на сервісі
<code>DEVICE_CREDENTIAL</code>	Унікальний номер пристрою, який був згенерований автоматично сервісом
<code>LED_BUILTIN</code>	Номер піну світлодіоду для мікроконтролері
<code>SSID</code>	Назва Wi-Fi мережі, до якої буде підключатися пристрій
<code>SSID_PASSWORD</code>	Пароль Wi-Fi мережі, до якої буде підключатися пристрій.

Крім того, для повного підключення до сервісу необхідно об'явити екземпляр класу `ThingESP8266` під назвою `thing`. В дужках потрібно вказати всі необхідні дані для підключення (ім'я на сервісі, назва пристрою, унікальний код пристрою).

У програмній реалізації були використані певні функції для реалізації поставлених задач. Всі функції можна побачити у таблиці 3.2.

Таблиця. 3.2 – Функції програми

Функція	Пояснення
<code>Serial.begin(115200);</code>	Відкриває послідовний порт і в дужках встановлює швидкість, з якою будуть передаватися дані
<code>Serial.println();</code>	Записує в послідовний порт повідомлення, що написано в дужках
<code>pinMode()</code>	Встановлює режим роботи заданого піна
<code>thing.add_wifi();</code>	Підключення Wi-Fi до мікроконтролеру
<code>digitalPin();</code>	Зчитує значення з певного піна, який вказаний в дужках
<code>thing.handle();</code>	Підключення до сервісу Thinger.io
<code>thing.write_bucket();</code>	Запис даних у конкретну папку на сервісі, назва до якої вказується у дужках
<code>ESP.deepSleep();</code>	Включення режиму DeepSleep. У дужках вказується час виключення у форматі мілісекунд
<code>digitalWrite();</code>	Зміна стану певного піна, назва якого і стан (HIGH або LOW) вказуються у дужках

Крім того, важливою особливістю програмування мікроконтролерів у середовищі Arduino є наявність обов'язковою функції `loop` типу `void`. Увесь програмний код записується саме у функції `loop`, яка запускає цикл безкінечної роботи програмного коду і виконання всього, що там написано.

3.3 Опис комп'ютерних експериментів

Протягом розробки системи було проведено декілька серій експериментів. Їх можна розділити за різними характеристиками, але головна класифікація відбувається на базі вибору джерела живлення для розроблюваної системи – від комп'ютера або він сонячної панелі.

Перші три серії експериментів відбувалися за допомогою прямого живлення від комп'ютера через відповідний кабель. Результати перших трьох експериментів можна побачити у таблиці 3.3

Таблиця 3.3 – Результати серій експериментів 1 – 3

№ серії експериментів	№	Інтервал, хв	Живлення, mAh	Т, хв	Режим	Відправлення даних у консоль
1	1.	1	22	60	Modem	так
	2.	2	21	60	Modem	так
	3.	6	21	60	Modem	так
	4.	10	21	60	Modem	так
2	5.	1	21	60	Modem	ні
	6.	2	20	60	Modem	ні
	7.	6	20	60	Modem	ні
	8.	10	20	60	Modem	ні
3	9.	1	5	13	DeepSleep	ні
	10.	2	4	11	DeepSleep	ні
	11.	6	1	3	DeepSleep	ні
	12.	10	1	2	DeepSleep	ні

Для зручності, результати серій експериментів 1 – 3 продемонстровані на рис. 3.3.

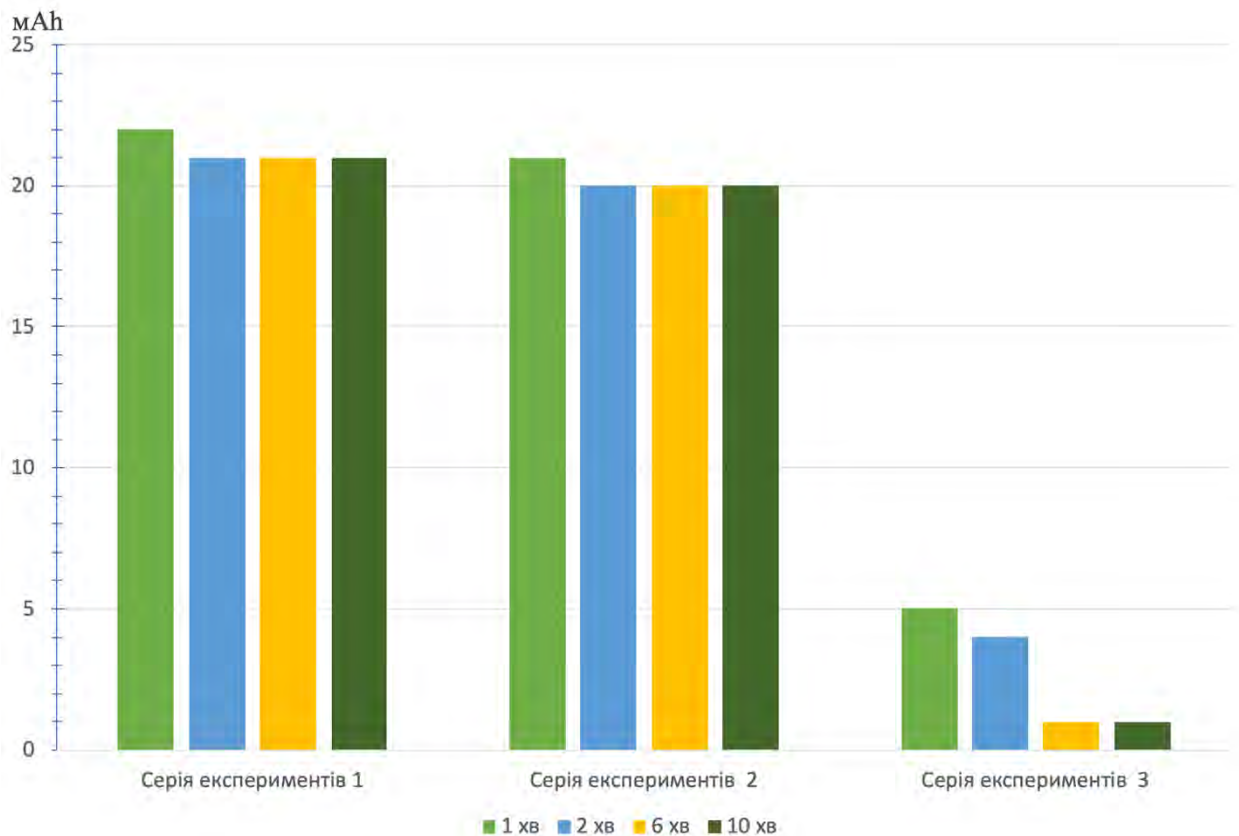


Рисунок 3.3 – Вимірювання живлення, експерименти 1 – 3

Одразу можна зробити декілька висновків. По-перше, споживання електроенергії і паралельною відправкою даних не тільки на сервер хмарного сховища, але і до порту комп'ютера – найвище (21 mAh – 22 mAh). По-друге, найменше споживання електроенергії вийшло при використанні режиму DeepSleep. Обидва ці висновки можна зробити не тільки за даними таблицями, але і згідно зображенню графіку.

Крім того, не зовсім очевидними є висновок, що відправка даних кожену хвилину у абсолютно всіх випадках споживала більше електроенергії ніж відправка даних кожні 10 хвилин.

З результатів експериментів 1 – 3 можна зробити висновок, що найефективнішим і найменш затратним варіантом є експеримент №12 – відправка даних кожні 10 хвилин з режимом DeepSleep.

Серії експериментів 4 – 6 проводилися так само, як і 1 – 3, але з єдиним виключенням – живлення Wi-Fi модулю відбувалося через сонячну панель. Результати серій експериментів 4 – 6 можна побачити у таблиці 3.4.

Таблиця 3.4 – Результати серій експериментів 4 – 6

№ серії експериментів	№	Інтервал, хв	Живлення, mAh	T, хв	Режим	Відправлення даних у консоль
4	1.	1	18	50	Modem	так
	2.	2	17	50	Modem	так
	3.	6	17	49	Modem	так
	4.	10	17	48	Modem	так
5	5.	1	19	55	Modem	ні
	6.	2	18	50	Modem	ні
	7.	6	17	49	Modem	ні
	8.	10	18	51	Modem	ні
6	9.	1	2	7	DeepSleep	ні
	10.	2	3	1	DeepSleep	ні
	11.	6	0	2	DeepSleep	ні
	12.	10	0	1	DeepSleep	ні

Так само, як і для попередніх експериментів, для зручності, результати серій 4 – 6 продемонстровані на рис. 3.4.

Важливо зазначити, що протягом проведення експериментів, існує похибка, котра залежить від сонячних променів, але у таблицю були записані саме середні значення.

Результати серій експериментів 4 – 6 виявилися трохи несподіваними. Перший висновок, який можна зробити – для відправки Wi-Fi модулем всіх потрібних даних у хмарне сховище протягом однієї години, на справді потрібно на 10 хвилин менше, ніж фактично години. Іншими словами, майже у всіх випадках, при прямому дротовому підключенні Wi-Fi модуля до комп'ютера, він працює з надлишком у як мінімум 10 хвилин, або з надлишком у 2 – 3 mAh.

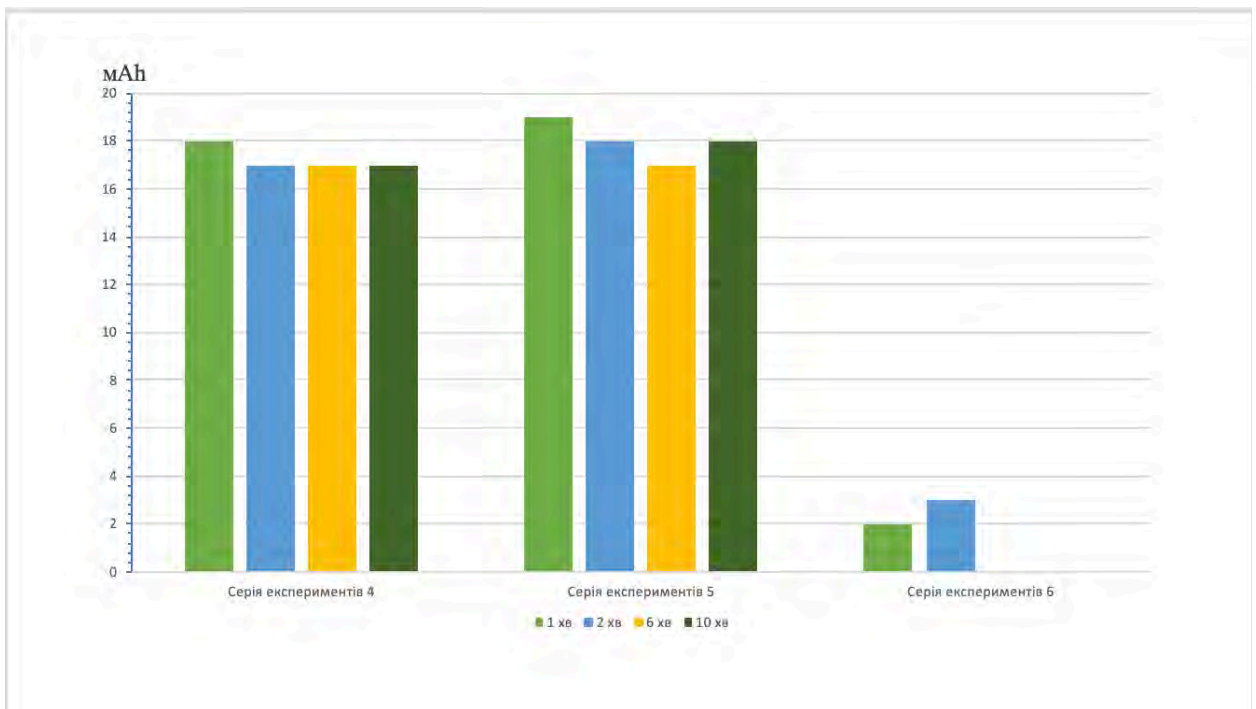


Рисунок 3.4 – Вимірювання живлення, експерименти 4 – 6

Крім того, цікавим є те, що протягом експериментів з сонячною панеллю, середня напруга була більшою, ніж у експериментах з дротом. Якщо при підключенні до комп'ютера середнє значення напруги було в межах 4.99 V – 5.03 V, то з сонячною панеллю цей інтервал становиться значно ширшим і відповідає 4.95 V – 5.12 V.

Ще один висновок, який можна зробити не тільки з результатів у таблиці, але і з графіку, це те, що, так само, як і у серіях експериментів 1 – 3, у серіях експериментів 4 – 6 найефективнішим виявились умови: режим DeepSleep, інтервал 10 хвилин. Саме при таких умовах, Wi-Fi модуль відправив всі необхідні дані до сервісу і працював лише 1 хвилину. Але навіть таке споживання можна ще краще оптимізувати.

Як було виявлено під час аналізу предметної області, однією з проблем використання сонячної енергії є проблема її нерівномірного вироблення. Виробка сонячної енергії є не однорідною. В одні дні, через невдалі умови, не вдається виробити нічого, а в інші – сонячна енергія виробляється з надлишком. Щоб уникнути цієї проблеми, можна ще краще оптимізувати такий алгоритм і додати до системи зовнішній акумулятор, який буде

накопичувати цей надлишок і дозволяти працювати системі під час невдалих умов.

Для такої оптимізації єдиною важливою умовою для зовнішнього акумулятора буде наявність двох USB-портів для одночасного заряду від сонячної панелі і роботи Wi-Fi модулю. Для розміру акумулятора особливих умов не існує, тому можна використовувати акумулятор з найменшим існуючим розміром на ринку – 3000 mAh.

Якщо за основу взяти, що в середньому, Wi-Fi модуль споживає кожні три хвилини 1 mAh, а у режимі DeepSleep з інтервалом відправки даних кожні 10 хвилин такий модуль працював 1 хвилину протягом однієї години, можна вирахувати, що за підключенням такого зовнішнього акумулятора, який був заряджений через сонячну панель від сонця, Wi-Fi модуль може працювати близько 9000 хвилин, або, іншими словами, 9000 годин реальних годин. Тобто, якщо зарядити такий акумулятор розміром 3000 mAh від сонця, він дозволить працювати Wi-Fi модулю у режимі DeepSleep з інтервалом у 10 хвилин більше ніж рік, або 375 днів без зупинки.

Таким чином, вся система буде максимально оптимізована. Живлення буде відбуватися тільки через сонячну енергію, не витрачаючи електроенергію, що вироблена на електростанціях, а система буде виконувати всі необхідні задачі.

3.4 Висновки

В цьому розділі були розроблені алгоритми роботи системи і проведені 6 важливих комп'ютерних експериментів. Проведені експерименти можна класифікувати за різними характеристиками: дві групи – живлення від ПК та від сонячної панелі; три групи – відповідно до програмного коду. Кожен такий експеримент відрізнявся режимом роботи модуля та навантаженням на нього. На основі результатів даних експериментів можна зробити декілька висновків.

По-перше, при порівнянні результатів серій експериментів можна побачити, що у всіх випадках споживання від сонячної панелі було меншим, ніж від ПК.

По-друге, найефективнішим показав себе режим режимом DeepSleep, у експериментах можна побачити, що з цим режимом модуль працює без надлишкового споживання електроенергії.

Таким чином, найвдалішим експериментом виявився експеримент під номером 6, споживання від сонячної панелі та з використанням режиму DeepSleep. Протягом години роботи за таких умов модуль фактично працював лише хвилину і спожив лише 1 mAh.

4 КЕРІВНИЦТВО КОРИСТУВАЧА

4.1 Призначення програми

В першу чергу, програма виконує функцію відправки та збереження даних, за умовами, що були встановлені програмістом. Саме ці умови повинні бути найбільш оптимальними та відповідати поставленим задачам.

4.2 Умови виконання програми

Для найбільш мінімальних умов виконання програми достатньо робочого комп'ютера, налаштованою програмістом системою і мікроконтролером ESP8266 та стабільної мережі Wi-Fi. Від користувача вимагається лише обрати дані для відправки, завантажити програму на мікроконтролер ESP8266 і після зберегти дані.

Якщо користувачу буде необхідно надіслати якісь конкретні дані або інші повідомлення, про це необхідно обов'язково повідомити програмісту, так як на даному етапі внесення змін до системи виконується виключно через програмний код.

4.3 Виконання програми

Виконання програми потребує попереднього налаштування її програмістом. Після чого, користувач може відстежувати відправку даних та їхнє збереження на сервісі Thinger.io, де, окрім статичного перегляду даних є можливість, за бажанням, графічного відображення результатів.

Після запуску програмістом системи, користувач може зайти на сайт сервісу Thinger.io, де можна відстежити коли саме підключився Wi-Fi мікроконтролер до сервісу, скільки даних було передано та інші дані, які залежать від локації інтернет-провайдера (рис. 4.1).

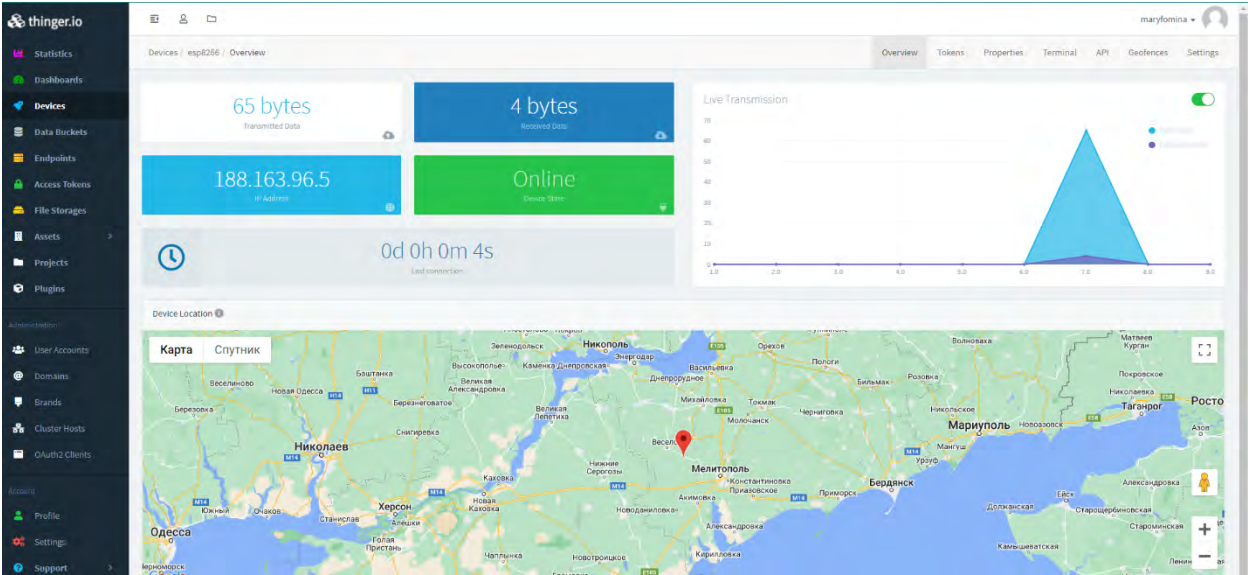


Рисунок 4.1 – Підключення мікроконтролера ESP8266 до сервісу Thingier.io

Для того, щоб побачити передані дані, користувачу потрібно перейти у розділ DataBucket і там обрати папку, створену програмістом. У цій папці можна буде побачити всю доступну інформацію по переданим даним (рис. 4.2).

Bucket Data

Date	Value
2022-05-09T14:22:40.112Z	9536
2022-05-01T19:24:00.925Z	7215
2022-05-01T19:14:27.242Z	7173
2022-05-01T19:04:57.401Z	11221
2022-05-01T18:55:23.975Z	11187
2022-05-01T18:45:47.988Z	7188
2022-05-01T18:36:16.055Z	7753
2022-05-01T18:26:52.199Z	11038
2022-05-01T17:44:07.493Z	7392

 Refresh

Рисунок 4.2 – Передані дані до сервісу Thingier.io

4.4 Повідомлення користувачу

ПОВІДОМЛЕННЯ: Дані не відправляються на сервіс Thinger.io.

ДЛЯ КОРИСТУВАЧА: Якщо користувач побачить, що дані не відправляються на сервіс, треба зайти у Data Bucket – Bucket – Setting та перевірити інтервал відправки даних.

ПОВІДОМЛЕННЯ: Стан підключеного пристрою «offline».

ДЛЯ КОРИСТУВАЧА: Натиснути кнопку Reset на мікроконтролері.

5 КЕРІВНИЦТВО ПРОГРАМІСТА

5.1 Призначення програми

У першу чергу система виконує експериментальну функцію, а значить має можливості для модернізації задля подальших експериментів. В першу чергу, програма надсилає прості дані до хмарного сервісу, а програміст відстежує процес відправки та живлення. Тож, головним призначенням програми є відправка даних при найбільш оптимальних умовах.

5.2 Умови виконання програми

Найголовніше для виконання програми це стабільна працююча мережа Wi-Fi, робочий мікроконтролер ESP8266 та ПК (Персональний комп'ютер). Крім того, програмісту необхідно зареєструватися у системі Thinger.io та внести дані про пристрій ESP8266.

Серед основні програмні засоби, які використовуються системою, окремого встановлення потребує тільки інтегровано середовище розробки Arduino. Доступ до результатів роботи програми здійснюється через будь який браузер з доступом до серверу Thinger.io.

Для коректної роботи одної з опціональних можливостей системи є живлення від сонячної панелі, яке можливе тільки за наявності сонця у межах робочої системи. Живлення від ПК може відбуватися за будь яких умов. Виключенням є нестача електроенергії, але в такому випадку робота системи неможлива.

5.3 Характеристика програмної системи

Проект був розроблений за допомогою мови програмування C++ у середовищі розробки Arduino з використанням хмарного сервісу Thinger.io, мікроконтролера ESP8266, монокристаличної кремнієвої сонячної панелі та вимірювача струму Keweisi KSW-V20.

Розроблений проєкт працює у ручному режимі та режимі очікування, тобто користувач повинен запускати систему за необхідністю та спостерігати за роботою певний проміжок часу.

Працездатність програми перевіряється як під час роботи, так і на основі отриманих даних. Якщо протягом роботи з програмою мікроконтролер не встигав підключитися до мережі Wi-Fi, з'являлись проблеми з сонячним світлом, або виникали якісь інші технічні труднощі, які ніяк не залежать від користувача, система все ще може продовжувати виконувати роботу, але такий експеримент потребує додаткових вимірів для точної коректності.

Якщо протягом роботи з програмою були проблеми з електропостачанням або у користувача існують проблеми з доступом до інтернету, а, через це, і з доступом до хмарного середовища, то програма не буде здатна виконувати свою роботу.

За підтримки сервісу Thingier.io система має зрозумілий інтерфейс та зручну структуру даних.

5.4 Звернення до програми

Для того, щоб запустити програму, окрім встановленого середовища розробки Arduino потрібно провести його налаштування відповідно до мікроконтролера ESP8266. Для цього, необхідно встановити відповідну бібліотеку. Після цього, необхідно ввести у код програми персональні дані, необхідні для підключення до сервісу Thingier.io. Коли всі дані були введені, а код програми був написаний або скопійований з сервісу GitHub можна приступати к налаштуванню системи. З цього моменту, налаштування поділяється на два варіанти.

Перший варіант налаштування умови роботи системи включають себе живлення від ПК. Для цього необхідно підключити вимірювач струму і мікроконтролер ESP8266 до ПК, завантажити скомпільовану програму і запустити її. Після чого, дані почнуть відправлятися до сервісу за обраним

інтервалом. Для відстеження результатів програми необхідно власноруч відстежувати час і зупинити роботу програми за необхідністю.

Другий варіант налаштування умов роботи системи майже не відрізняється від попереднього окрім перших кроків. В цьому випадку, спочатку треба підключити мікроконтролер для ПК тільки для завантаження програмного коду. Після, само підключення мікроконтролера і вимірювача струму відбувається не до ПК, а до сонячної панелі, яка попередньо встановлюється у місце з сонячним світлом.

Важливо зазначити, що у при використанні режиму DeepSleep необхідно встановити контакт на мікроконтролері між пінами RST (reset) та D0 для коректної роботи циклу у коді програми.

5.5 Початкові та вихідні дані

Серед вхідних даних є тільки персональні дані для доступу до сервісу Thinger.io, такі як:

- ім'я на сайті;
- унікальна назва мікроконтролера;
- унікальний код мікроконтролера;
- назва мережі Wi-Fi;
- пароль до мережі Wi-Fi;
- інтервал відправки даних системи.

Серед вихідних даних є повідомлення про стан системи виключно для програміста та надіслані дані у сервіс Thinger.io, в даному випадку – мілісекунди.

Опціонально, вихідними даними можуть виступати будь які необхідні користувачу дані, як наприклад з інших IoT пристроїв. Але за таких умов система потребує ручної модернізації та зміни програмного коду, яке може виконати лише програміст.

5.6 Повідомлення програмісту

ПОВІДОМЛЕННЯ: esptool.FatalError: Failed to connect to ESP8266:
Timed out waiting for packet header

ДЛЯ ПРОГРАММІСТА: Перевірити стан дроту, який підключається до ПК. Повторити завантаження програмного коду на мікроконтролер.

ПОВІДОМЛЕННЯ: serial.serialutil.SerialException: could not open port 'COM3':

ДЛЯ ПРОГРАММІСТА: Перевірити порт підключення.
Перепідключити дрот у інший USB-порт.

ВИСНОВКИ

В результаті проведеної роботи було розроблено програмне забезпечення для системи передачі даних з портативного пристрою на основі Wi-Fi мікроконтролера ESP8266. Для цього була проаналізована предметна область, розроблена архітектура системи та керівництва для користувача та програміста, проведені комп'ютерні експерименти та проаналізовані результати роботи програмного забезпечення. Основні положення й результати роботи були представлені в тезах доповіді НПК Тиждень науки – 2022.

Протягом роботи над проектом були успішно виконані всі поставлені задачі. Були проведені 6 серій експериментів при різних умовах відправки даних: як від різних джерел живлення, так і з різними інтервалами часу.

На основі експериментальних даних, можна побачити, що розроблене програмне забезпечення для системи живлення має найоптимальніше споживання електроенергії з умовами живлення від сонячної панелі та з використанням режиму DeepSleep.

Однак, система має свої недоліки. На даному етапі розробки користувач не має можливості вільно обирати дані для відправки у хмарне сховище і потребує звертатися для програміста, який у ручному режимі повинен змінювати тип даних або їхнє джерело.

Згодом, на правах доопрацювання системи необхідним буде підключити інші модулі для вимірювання даних задля відправки у хмарне середовище і провести експерименти з зарядом портативної акумуляторної батареї від сонячної енергії. Додатково, у майбутньому можна провести експерименти з іншими модулями.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Laricchia, F. Wearables - statistics & facts [Electronic resource] / F. Laricchia. – Access mode: https://www.statista.com/topics/1556/wearable-technology/#dossierContents__outerWrapper.
2. Jaganmohan, M. Production of concentrated solar power worldwide from 2009 to 2019 [Electronic resource] / M. Jaganmohan. – Access mode: <https://www.statista.com/statistics/1032894/concentrated-solar-power-production-globally/>.
3. Mobile device [Electronic resource]. – Access mode: https://en.wikipedia.org/wiki/Mobile_device.
4. Perkins, S. Mobile devices: examples, impact & trends [Electronic resource] / S. Perkins, L. Chapel. – Access mode: <https://study.com/academy/lesson/mobile-devices-examples-impact-trends.html>.
5. Leavy, M. The use of mobile devices in healthcare [Electronic resource] / M. Leavy. – Access mode: <https://www.elitelearning.com/resource-center/health-systems-management/the-use-of-mobile-devices-in-healthcare/>.
6. Tocci, M. History and evolution of smartphones [Electronic resource] / M. Tocci. – Access mode: <https://simpletexting.com/where-have-we-come-since-the-first-smartphone/>.
7. Stürmer, T. Die besten smatrphones 2021: test und vergleich [Elektronische ressource] / T. Stürmer. – Access mode: <https://www.imtest.de/die-besten-smartphones-test-kamera-akku-display/>.
8. When were headphones invented: the complete history [Electronic resource]. – Access mode: <https://www.headphonesty.com/2021/05/when-were-headphones-invented/>.
9. Чи варто купувати бездротові навушники - особливості вибору блютуз гарнітури [Електронний ресурс]. – Режим доступу: <https://myt.com.ua/ua/news/stoit-li-pokupat-besprovodnye-naushniki>.

10. Ozadam, O. The pros & cons of using a smartwatch - is it worth getting a smartwatch? [Electronic resource] / O. Ozadam. – Access mode: <https://smartifylife.com/worth-getting-a-smartwatch-benefits/>.
11. Носимые медицинские устройства для контроля над здоровьем [Электрон. ресурс]. – Режим доступа: <http://moitabletki.ru/info/news-gadget-zdorovie.html>.
12. Подробный обзор умных часов с описанием их функций [Электрон. ресурс]. – Режим доступа: <https://gadgetpage.ru/gadzhety/2348-dumaete-kakie-umnye-chasy-vybrat-togda-jeta-statja-dlja-vas-podrobnyj-obzor-smart-chasov-s-opisaniem-ih-funkcij.html>.
13. Типы аккумуляторных батарей [Электрон. ресурс]. – Режим доступа: <https://best-energy.com.ua/support/battery/414-vidy-i-tipy-akkumulyatornykh-batarej-v-podrobnostiakh#battery-info-type>.
14. Татарников, О. Альтернативные источники питания для мобильных устройств [Электрон. ресурс] / О. Татарников // КомпьютерПресс, 2003. – Режим доступа: <https://compress.ru/article.aspx?id=11827>.
15. Прикладная электрохимия / [Алабышев А. Ф., Вячеславов П. М., Гальнбек А. А. и др.]. – Санкт-Петербург: Химия, 1962. – 536 с.
16. Девис, С. Электрохимический словарь / С. Девис, А. Джеймс. — М.: Мир, 1979 – 287 с.
17. Barsukov, Y. Battery power management for portable devices / Y. Barsukov, J. Qian. – Norwood: Artech House, 2013. – 268 p.
18. Motorola and freeplay energy intro FreeCharge [Electronic resource]. - Access mode: <https://eepower.com/news/motorola-and-freeplay-energy-intro-freecharge/#>.
19. Tomonari, T. High capacity sulfurized alcohol composite positive electrode materials applicable for lithium sulfur batteries [Electronic resource] / [T. Tomonari, K. Toshikatsu, K. Hiroyuki et al.]. – Access mode: <https://iopscience.iop.org/article/10.1149/2.0501701jes/meta>.
20. Priya, S. Energy Harvesting Technologies / S. Priya, D. J. Inmann. – New York: Springer, 2009. – 544 p.

21. Flournoy, D. Solar power satellites / D. M. Flournoy. – Ohio: Springer, 2012. – 105 p.
22. Rajan, R. Low cost portable wind power generation for mobile charging applications / [R. Rajan, V. Panchalogaranjan, V. Aravinthan et al.]. – 2018. – 26 p.
23. Вечные фонари с динамо-машиной [Электрон. ресурс]. – Режим доступа: <https://kremenchug.ua/news/consumer/gadgets/39699-vechnye-fonari-s-dinamo-mashinoy.html>.
24. 10 способов зарядки телефона без электричества [Электрон. ресурс]. – Режим доступа: <https://newcivilization.mirtesen.ru/blog/43357162092/10-sposobov-zaryadki-telefona-bez-elektrichestva>.
25. Чому зупиняється механічний годинник з автопід заводом? [Електронний ресурс]. – Режим доступу: <https://secunda.com.ua/ua/articles/cho-mu-zupinyayetsya-mekhanichniy-godinnik-z-avtopidzavodom/10631/>.
26. Полное руководство для производителей часовых механизмов [Электрон. ресурс]. – Режим доступа: <https://www.opswatches.com/ru/полное-руководство-по-производителя/>.
27. Power Bank с солнечной батареей – расчет на безграмотность [Электрон. ресурс]. – Режим доступа: <https://electricadom.com/power-bank-s-solnechnojj-batareejj-raschet-na-bezgramotnost.html>.
28. Solar charges – power on the go for mobile devices [Electronic resource]. – Access mode: <https://www.solup.com/solar-chargers-power-go-mobile-devices/>.
29. Принцип роботи сонячної електростанції: види й особливості використання [Електронний ресурс]. – Режим доступу: <https://nenergy.com.ua/ua/statti/princip-roboti-sonyachno-elektrostanc-konstrukcyn-nyuansi-zastosuvannya/>.
30. Winter, C. Solar power plants / C. Winter, R. Sizmann, L. Vant-Hull. – Berlin: Springer-Verlag, 2011. – 437 p.
31. Типи сонячних панелей [Електронний ресурс]. – Режим доступу: <https://www.atmosfera.ua/uk/sonyachni-elektrostancii/tipi-sonyachnix-panelej/>

32. Будо́ва та принцип дії електровимірювальних приладів [Електронний ресурс]. – Режим доступу: <https://disted.edu.vn.ua/courses/learn/1807>.
33. USB тестер [Електронний ресурс]. – Режим доступу: <https://gsm-komplekt.ua/category/usb-tester/>.
34. USB на́грузка на 4 резистора [Електронний ресурс]. – Режим доступу: <https://gsm-komplekt.ua/product/39295/>.
35. Keweisi KWS-V20 USB тестер измерение емкости, тока, напряжения [Электрон. ресурс]. – Режим доступа: <https://pokupochka.net.ua/p730461020-keweisi-kws-v20.html>.
36. Тестер USB unit UT658 [Електронний ресурс]. – Режим доступу: <https://rozetka.com.ua/54504648/p54504648/>.
37. Usb тестер Atorch u96 [Електронний ресурс]. – Режим доступу: <https://rozetka.com.ua/301941218/p301941218/>.
38. Briere, D. Wireless home networking for dummies / D. Briere, P. Hurley, E. Ferris. – Indianapolis: Wiley, 2008. – 408 p.
39. What do Wi-Fi statistics tell us and what can we do with that information? [Electronic resource]. – Access mode: <https://www.avsystem.com/blog/wifi-statistics/>.
40. Vailshery, L. Bluetooth - statistics and facts [Electronic resource] / L. S. Vailshery. – Access mode: <https://www.statista.com/statistics/1226712/global-bluetooth-audio-entertainment-device-shipment-forecast/>
41. Korowajczuk, L. LTE, WiMAX and WLAN network design, optimization and performance Analysis / L. Korowajczuk. – Reston: Wiley, 2011. – 782 p.
42. Thomas, J. The history of Wi-Fi [Electronic resource] / J. Thomas. – Access mode: <https://purple.ai/blogs/history-wifi/>.
43. Chandra, P. Wi-Fi telephony: challenges and solutions for voice over WLANs / P. Chandra, D. Lide. – Burlington: Elsevier, 2007. – 288 p.
44. Faster. Farther. Better. The evolution of 802.11. [Electronic resource]. – Access mode: <https://www.bbxservices.com/resources/blog/bbns/2018/04/30/802.11-wireless-standards-explained>.

45. Sibille, A. MIMO: from theory to implementation / A. Sibille, C. Oestges, A. Zanella. – Oxford: Elsevier, 2010. – 360 p.
46. Sag, A. 802.11AX is coming from Intel in 2018 [Electronic resource] / A. Sag – Forbes, 2018. – Access mode: <https://www.forbes.com/sites/moorinsights/2018/01/04/802-11ax-is-coming-from-intel-in-2018/?sh=e467f84dcc5b>.
47. IEEE SA standards board approvals [Electronic resource]. – Access mode: <https://standards.ieee.org/about/sasb/sba/feb2021.html>.
48. Gulasekaran, S. Wi-Fi 6: protocol and network / S. Gulasekaran, S. Sankaran. – Norwood: Artech House, 2021. – 227 p.
49. Ferrara, J. How do Bluetooth devices work? [Electronic resource] / J. Ferrara. – Access mode: <https://scienceline.org/2016/04/how-do-bluetooth-devices-work/>.
50. Franklin, C. How Bluetooth Works [Electronic resource] / C. Franklin, C. Pollette. – Access mode: <https://electronics.howstuffworks.com/bluetooth.htm>.
51. Bluetooth wireless technology [Electronic resource]. – Access mode: <https://www.bluetooth.com/learn-about-bluetooth/tech-overview/>.
52. Cheruvu, S. Demystifying internet of things security: successful IoT device/edge and platform security deployment / [S. Cheruvu, A. Kumar, N. Smith et al.]. – New York: Apress Open, 2019. – 516 p.
53. Hübschmann, I. How to choose the best IoT Wi-Fi module [Electronic resource] / I. Hübschmann. – Access mode: <https://www.nabto.com/how-to-choose-best-wifi-module-for-iot/>.
54. Modules [Electronic resource]. – Access mode: <https://www.espressif.com/en/products/modules>.
55. Microcontrollers & microprocessors [Electronic resource]. – Access mode: <https://www.st.com/en/microcontrollers-microprocessors.html>.
56. WiFi modules [Electronic resource]. – Access mode: <https://www.espressif.com/en/products/modules>.
57. Foxworth, T. Making the ESP8266 low-powered with deep sleep [Электронный ресурс] / T. Foxworth. – Access mode: <https://www.losant.com/blog/making-the-esp8266-low-powered-with-deep-sleep>.

58. ESP8266MOD WiFi module [Electronic resource]. – Access mode: https://www.tinyosshop.com/index.php?route=product/product&product_id=908.
59. Wi-Fi модуль ESP32 ESP-Wroom-32 [Электронный ресурс]. – Режим доступа: <https://www.mini-tech.com.ua/esp32-wifi-modul>.
60. ESP32 S2 WROOM – Low Power WiFi module [Electronic resource]. – Access mode: <https://www.gridconnect.com/products/esp32-s2-wroom-wi-fi-module>.
61. Why ARM is most popular? ARM architecture [Electronic resource]. – Access mode: <https://www.elprocus.com/arm-architecture/>.
62. AT91SAM7S256-AU microchip (Atmel) [Electronic resource]. – Access mode: <https://www.tme.eu/ru/details/at91sam7s256-au/lineika-arm-32-bit/microchip-atmel/>.
63. ATmega32, ATmega32L documentation [Electronic resource]. – Access mode: <https://ww1.microchip.com/downloads/en/DeviceDoc/doc2503.pdf>.
64. PIC18F2455/2550/4455/4550 Data Sheet [Electronic resource]. – Access mode: <https://ww1.microchip.com/downloads/en/devicedoc/39632e.pdf>.
65. Silicon TechnoLabs atmel ATmega32 microcontroller [Electronic resource]. – Access mode: <https://www.amazon.in/Silicon-TechnoLabs-ATMEL-ATMEGA32-Microcontroller/dp/B018LHXQO0>.
66. PIC18F4550 microcontrolador [Electronic resource]. – Access mode: <https://uelectronics.com/producto/pic18f4550-microcontrolador/>.
67. STM32G030F6 [Electronic resource]. – Access mode: <https://www.st.com/en/microcontrollers-microprocessors/stm32g030f6.html>.
68. STM32G030F6P6 Stmicroelectronics [Electronic resource]. – Access mode: <https://www.tme.eu/ru/details/stm32g030f6p6/mikrokontrollery-st/stmicroelectronics/>.
69. STM8L051F3 [Electronic resource]. – Access mode: <https://www.st.com/resource/en/datasheet/stm8l051f3.pdf>.
70. STM8L051F3P6 [Electronic resource]. – Access mode: <https://aliradar.com/item/33014309340-Mcu-8-bitnyj-Stm8L-Stm8-Cisc-8Kb-Flash-20-kontaknyj-Tssop-TR-Stm8L051-Tssop20-Stm8L051F3P6>.

71. Arduino IDE for Visual Studio [Electronic resource]. – Access mode: <https://www.visualmicro.com>.
72. Visual Studio 2019 product family system requirements [Electronic resource]. — Access mode: <https://docs.microsoft.com/en-us/visualstudio/releases/2019/system-requirements>
73. Visual Micro: Arduino Programming with Visual [Electronic resource]. – Access mode: <https://www.megunolink.com/articles/coding/visual-micro-arduino-visual-studio/>.
74. Platformio [Electronic resource]. – Access mode <https://platformio.org>.
75. Requirements for Visual Studio Code [Electronic resource]. – Access mode: <https://code.visualstudio.com/docs/supporting/requirements>.
76. Arduino software [Electronic resource]. – Access mode: <https://www.arduino.cc/en/software>.
77. Kools, P. The Arduino IDE [Electronic resource] / [P. Kools, I. Wilms, D. Babu et al.]. – Access mode: <https://se.ewi.tudelft.nl/desosa2019/chapters/arduino-ide/>.
78. Data Dog: modern monitoring & security [Electronic resource]. – Access mode: <https://www.datadoghq.com>.
79. DataDog: libraries [Electronic resource]. – Access mode: <https://docs.datadoghq.com/developers/community/libraries/>.
80. Datadog OpenTracing C++ client [Electronic resource]. – Access mode: <https://github.com/DataDog/dd-opentracing-cpp>.
81. Google Cloud [Electronic resource]. – Access mode: <https://cloud.google.com>.
82. About the API gateway development environment [Electronic resource]. – Access mode: <https://cloud.google.com/api-gateway/docs/about-dev-env>.
83. Google Cloud IoT JWT documentation [Electronic resource]. – Access mode: <https://github.com/GoogleCloudPlatform/google-cloud-iot-arduino>.
84. Thinger.io platform [Electronic resource]. – Access mode: <https://thinger.io>.

85. ThingSpeak for IoT projects [Electronic resource]. – Access mode: <https://thingspeak.com>.

86. Scharler, H. Libelium now supports ThingSpeak with MatLab enabled IoT Analytics [Electronic resource] / H. Scharler. – Access mode: <https://is.gd/0yVy0y>

ДОДАТОК А
Текст програми

A.1 Експеримент 1,4

```

#include <ThingerESP8266.h>

#define USERNAME "maryfomina" // оголошення змінних для
підключення мікроконтролера до сервісу Thinger.io

#define DEVICE_ID "esp8266"
#define DEVICE_CREDENTIAL "fd122008"
#define LED_BUILTIN 2
#define SSID "Home_Sweet_Home"
#define SSID_PASSWORD "1231234567"

unsigned int sendRandom;

ThingerESP8266 thing(USERNAME, DEVICE_ID,
DEVICE_CREDENTIAL); // Створення об'єкту класу ThingerESP8266 для
підключення до сервісу

void setup() {
    Serial.begin(115200); // об'ява швидкості підключення до порту
    Serial.println("pinMode Led - Output"); // виведення повідомлення
підключення світлодіоду
    pinMode(LED_BUILTIN, OUTPUT); // підключення світлодіоду
    Serial.println("Set Setting Wi-Fi"); // виведення повідомлення про
підключення до Wi-Fi
    thing.add_wifi(SSID, SSID_PASSWORD); // підключення
мікроконтролера до Wi-Fi
}

void loop() { // початок циклу
    Serial.println("-loop-"); // нескінченне виведення повідомлення у порт
для додаткового навантаження
    if ((millis() - sendRandom) >= 60000) //кожні 60 секунд відбувається
відправка даних
    {

```

Serial.println("Send/Get data:"); // виведення повідомлення в про
відправку даних

thing["led"] << digitalPin(LED_BUILTIN); // відправка стану
світлодіода до мікроконтролера

thing["millis"] >> outputValue(millis()); // відправка до об'єкту класу
даних про скільки мілісекунд працював модуль

Serial.println("Send data - ok"); // відправка повідомлення, що все
спрацювало

```
    sendRandom = millis();
}
thing.handle(); // підключення модулю до сервісу
}
```

A.2 Експеримент 2,5

```
#include <ThingerESP8266.h>

#define USERNAME "maryfomina" // оголошення змінних для  
підключення модулю до сервісу Thinger.io
```

```
#define DEVICE_ID "esp8266"
#define DEVICE_CREDENTIAL "fd122008"
#define LED_BUILTIN 2
#define SSID "Home_Sweet_Home"
#define SSID_PASSWORD "1231234567"
unsigned int sendRandom;
```

```
ThingerESP8266 thing(USERNAME, DEVICE_ID,  
DEVICE_CREDENTIAL); // Створення об'єкту класу ThingerESP8266 для  
підключення до сервісу
```

```
void setup() {
    Serial.begin(115200); // об'ява швидкості підключення до порту
    pinMode(LED_BUILTIN, OUTPUT); // підключення світлодіоду
```

```

thing.add_wifi(SSID,    SSID_PASSWORD);    //    підключення
мікроконтролеру до Wi-Fi
}
void loop() { // початок циклу
    if ((millis() - sendRandom) >= 60000) //кожні 60 секунд відбувається
відправка даних
    {
        thing["led"]  <<  digitalPin(LED_BUILTIN); //  відправка  стану
світлодіода до мікроконтролеру
        thing["millis"] >> outputValue(millis()); // відправка до об'єкту класу
даних про скільки мілісекунд працював модуль
        sendRandom = millis();
    }
    thing.handle(); // підключення модулю до сервісу
}

```

A.3 Експеримент 3,6

```

#include <ThingerESP8266.h>
#define  USERNAME  "maryfomina" // оголошення змінних для
підключення модулю до сервісу Thinger.io
#define DEVICE_ID "esp8266"
#define DEVICE_CREDENTIAL "fd122008"
#define LED_BUILTIN 2
#define SSID "Home_Sweet_Home"
#define SSID_PASSWORD "1231234567"

ThingerESP8266          thing(USERNAME,          DEVICE_ID,
DEVICE_CREDENTIAL); // Створення об'єкту класу ThingerESP8266 для
підключення до сервісу

```

```

void setup() {
    Serial.begin(115200); // об'ява швидкості підключення до порту
    pinMode(LED_BUILTIN, OUTPUT); // підключення світлодіоду
    thing.add_wifi(SSID,    SSID_PASSWORD);    //    підключення
мікроконтролеру до Wi-Fi
}
void loop() {
    thing.handle(); // підключення модулю до сервісу
    thing["millis"] >> outputValue(millis()); // відправка до об'єкту класу
даних про скільки мілісекунд працював модуль
    thing.write_bucket("bucket_sleepMode", "millis"); //відправка до сервісу
даних про час роботи у мілісекундах
    Serial.println("I'm awake, but I'm going into deep sleep mode for 60
seconds"); // виведення повідомлення про включення режиму DeepSleep
    ESP.deepSleep(60e6); // режим DeepSleep на 60 секунд
    digitalWrite(D0, LOW);
}

```

A.4 Змінений код для експерименту 1,4

```

#include <ThingyESP8266.h>
#define USERNAME "maryfomina" // оголошення змінних для
підключення мікроконтролера до сервісу Thingy.io
#define DEVICE_ID "esp8266"
#define DEVICE_CREDENTIAL "fd122008"
#define LED_BUILTIN 2
#define SSID "Home_Sweet_Home"
#define SSID_PASSWORD "1231234567"
unsigned int sendRandom;

```

```

    ThingerESP8266          thing(USERNAME,          DEVICE_ID,
    DEVICE_CREDENTIAL); // Створення об'єкту класу ThingerESP8266 для
    підключення до сервісу

    void setup() {
        Serial.begin(115200); // об'ява швидкості підключення до порту
        Serial.println("pinMode Led - Output"); // виведення повідомлення
    підключення світлодіоду
        pinMode(LED_BUILTIN, OUTPUT); // підключення світлодіоду
        Serial.println("Set Setting Wi-Fi"); // виведення повідомлення про
    підключення до Wi-Fi
        thing.add_wifi(SSID,    SSID_PASSWORD);    //    підключення
    мікроконтролеру до Wi-Fi
    }

    void loop() { // початок циклу
        Serial.println("-loop-"); // нескінченне виведення повідомлення у порт
    для додаткового навантаження
        if((millis() - sendRandom) >= 120000) //кожні 120 секунд відбувається
    відправка даних
        {
            Serial.println("Send/Get data:"); // виведення повідомлення в про
    відправку даних
            thing["led"]  <<  digitalPin(LED_BUILTIN); // відправка стану
    світлодіода до мікроконтролеру
            thing["millis"] >> outputValue(millis()); // відправка до об'єкту класу
    даних про скільки мілісекунд працював модуль
            Serial.println("Send data - ok"); // відправка повідомлення, що все
    спрацювало
            sendRandom = millis();
        }
        thing.handle(); // підключення модулю до сервісу
    }

```

A.5 Змінений код для експерименту 2,5

```
#include <ThingerESP8266.h>

#define USERNAME "maryfomina" // оголошення змінних для
підключення модулю до сервісу Thinger.io

#define DEVICE_ID "esp8266"
#define DEVICE_CREDENTIAL "fd122008"
#define LED_BUILTIN 2
#define SSID "Home_Sweet_Home"
#define SSID_PASSWORD "1231234567"

unsigned int sendRandom;

ThingerESP8266          thing(USERNAME,          DEVICE_ID,
DEVICE_CREDENTIAL); // Створення об'єкту класу ThingerESP8266 для
підключення до сервісу

void setup() {
    Serial.begin(115200); // об'ява швидкості підключення до порту
    pinMode(LED_BUILTIN, OUTPUT); // підключення світлодіоду
    thing.add_wifi(SSID,    SSID_PASSWORD);    //    підключення
мікроконтролеру до Wi-Fi
}

void loop() { // початок циклу
    if ((millis() - sendRandom) >= 12000) //кожні 120 секунд відбувається
відправка даних
    {
        thing["led"]  <<  digitalPin(LED_BUILTIN); // відправка  стану
світлодіода до мікроконтролеру
        thing["millis"] >>  outputValue(millis()); // відправка до об'єкту класу
даних про скільки мілісекунд працював модуль
        sendRandom = millis();
    }
}
```

```
thing.handle(); // підключення модулю до сервісу
}
```

A.6 Змінений код для експерименту 3,6

```
#include <ThingerESP8266.h>

#define USERNAME "maryfomina" // оголошення змінних для
підключення модулю до сервісу Thinger.io

#define DEVICE_ID "esp8266"
#define DEVICE_CREDENTIAL "fd122008"
#define LED_BUILTIN 2
#define SSID "Home_Sweet_Home"
#define SSID_PASSWORD "1231234567"

ThingerESP8266 thing(USERNAME, DEVICE_ID,
DEVICE_CREDENTIAL); // Створення об'єкту класу ThingerESP8266 для
підключення до сервісу

void setup() {
  Serial.begin(115200); // об'ява швидкості підключення до порту
  pinMode(LED_BUILTIN, OUTPUT); // підключення світлодіоду
  thing.add_wifi(SSID, SSID_PASSWORD); // підключення
мікроконтролеру до Wi-Fi
}

void loop() {
  thing.handle(); // підключення модулю до сервісу
  thing["millis"] >> outputValue(millis()); // відправка до об'єкту класу
даних про скільки мілісекунд працював модуль
  thing.write_bucket("bucket_sleepMode", "millis"); //відправка до сервісу
даних про час роботи у мілісекундах
```

```
Serial.println("I'm awake, but I'm going into deep sleep mode for 60  
seconds"); // виведення повідомлення про включення режиму DeepSleep  
ESP.deepSleep(120e6); // режим DeepSleep на 120 секунд  
digitalWrite(D0, LOW);  
}
```

ДОДАТОК Б
Слайди презентації

Дипломна робота бакалавра

**Програмне забезпечення для портативного
електронного пристрою на основі Wi-Fi модуля**

Виконала
ст. гр. КНТ-118

М. О. Фоміна

Керівник
к.т.н., доцент

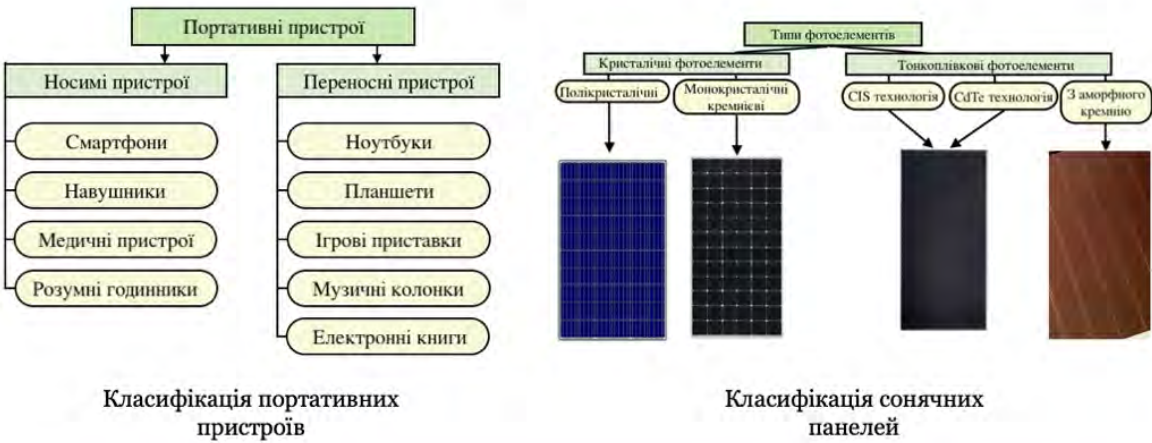
А. В. Пархоменко

2022

1

Рисунок Б.1 – Слайд 1

Актуальність роботи



2

Рисунок Б.2 – Слайд 2

Актуальність роботи

Назва / характеристики	Напруга, В	Споживання у сплячому режимі	CPU, МГц	RAM	Ethernet	USB-Host	Bluetooth	Wi-Fi	Ціна, грн
ESP8266	3.3	20 мА	32-bit, 160 МГц	160 Кб	–	–	–	+	90-150
ESP32	2.3 – 3.6	5 мА	32-bit, 80-240 МГц	520 Кб	+	–	+	+	130-150
ESP32-S2	1.8 – 3.3	20 мА	32-bit, 240 МГц	320 Кб	–	+	–	+	130-180
ARM7	1.6 – 3.6	Немає даних	32-bit, 60 МГц	512 Кб	+	+	–	–	270-300
AVR ATmega32	4.5 – 5.5	Немає даних	8-bit, 16 МГц	32Кб	–	–	–	–	110 - 125
PIC18F4550	4.2 – 5.5	Немає даних	8-bit 48 МГц	2к x 8	–	+	–	–	335
STM32G030 F6P6	2 – 3.6	Немає даних	32-bit, 64 МГц	8 Кб	–	+	–	–	50-80
STM8L051F 3P6	1.6 – 3.6	Немає даних	8-bit, 16 МГц	1к x 8	–	–	–	–	160

Аналіз існуючих мікроконтролерів

3

Рисунок Б.3 – Слайд 3

Мета та задачі

Мета роботи – дослідити та розробити програмне забезпечення для енергоефективної передачі даних з портативного пристрою з використанням бездротових технологій та альтернативних джерел живлення.

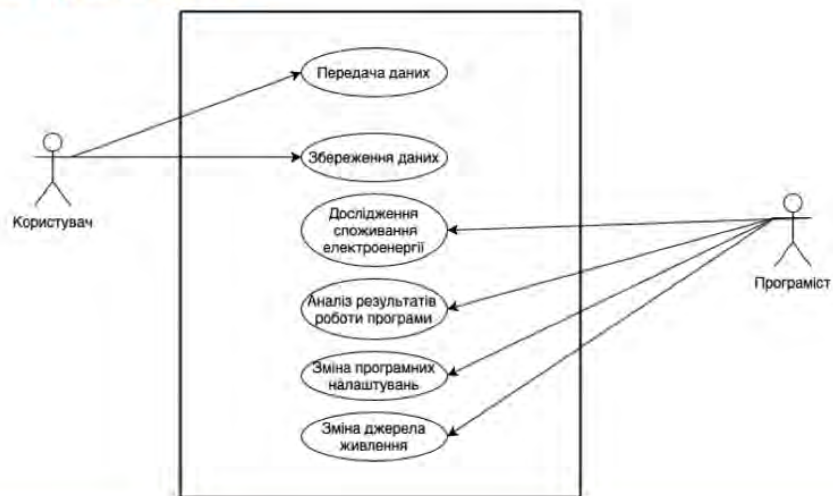
Задачі:

- виконати аналіз вимог до програмного забезпечення;
- розробити архітектуру системи та обрати інструментарій розробки;
- виконати програмну реалізацію;
- провести серію експериментів та виконати оптимізацію програмного коду;
- розробити програмну документацію.

4

Рисунок Б.4 – Слайд 4

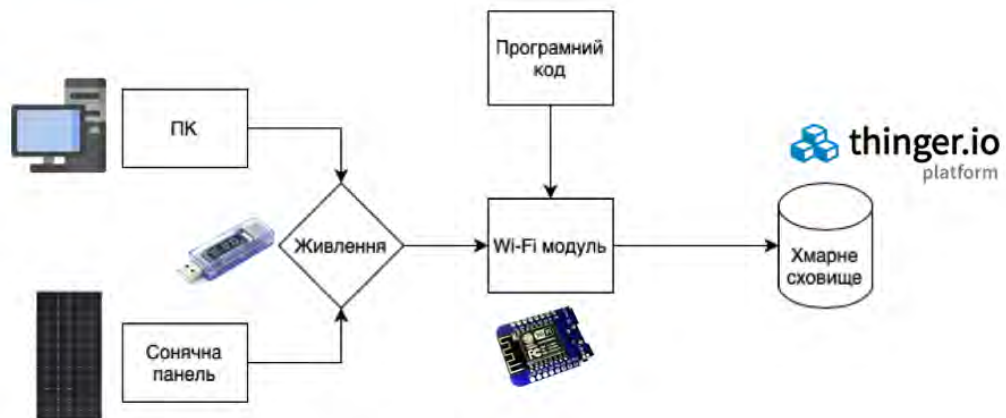
Діаграма прецедентів



5

Рисунок Б.5 – Слайд 5

Архітектура системи



6

Рисунок Б.6 – Слайд 6

Аналіз та вибір інтегрованого середовища розробки

Платформа\ характеристика	ОС	Процесор, ГГц	ОЗУ, ГБ	Розмір, ГБ	Ціна	Необхідність встановлення додаткових середовищ
Visual Micro	Windows	1.8	2	20	12 \$	так
Platformio	Windows, Mac, Linux	1.6	1	500 мб	Безкоштовно	так
Arduino	Windows, Mac, Linux	1.3	256	540 мб	Безкоштовно	ні

7

Рисунок Б.7 – Слайд 7

Аналіз та вибір хмарних сховищ

Платформа \ характеристика	Можливість підключати напряму ESP8266	Встановлення додаткових програм	Обробка даних у сховищі	Ціна	Віджети	Детальний перегляд даних
ThingSpeak	+	–	–	Безкоштовно	+	–
Thingier.io	+	–	+	Безкоштовно	+	+
Google Cloud	–	+	+	Пробний період, 300\$	+	+
Datadog	–	+	+	Пробний період, 15\$	+	+

8

Рисунок Б.8 – Слайд 8

Розроблені алгоритми

Алгоритм роботи програми При живленні від ПК



Алгоритм роботи програми при живленні від сонячної панелі

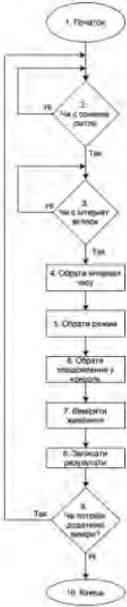


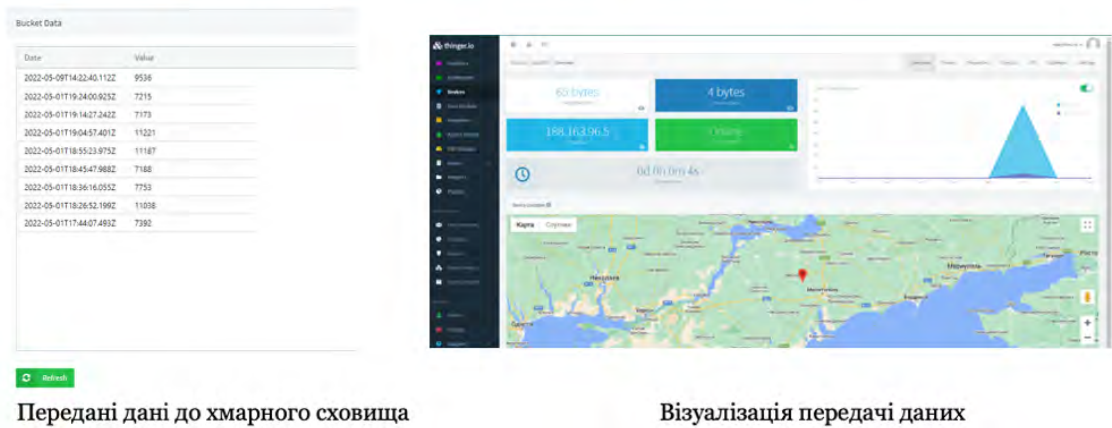
Рисунок Б.9 – Слайд 9

Програмна реалізація

Функція	Опис
Serial.begin(115200);	Відкриває послідовний порт і в дужках встановлює швидкість, з якою будуть передаватися дані
Serial.println();	Записує в послідовний порт повідомлення, що написано в дужках
pinMode()	Встановлює режим роботи заданого піна
thing.add_wifi();	Підключення Wi-Fi до мікроконтролеру
digitalPin();	Зчитує значення з певного піна, який вказаний в дужках
thing.handle();	Підключення до сервісу Thinger.io
thing.write_bucket();	Запис даних у конкретну папку на сервісі, назва до якої вказується у дужках
ESP.deepSleep();	Включення режиму DeepSleep. У дужках вказується час виключення у форматі мілісекунд

Рисунок Б.10 – Слайд 10

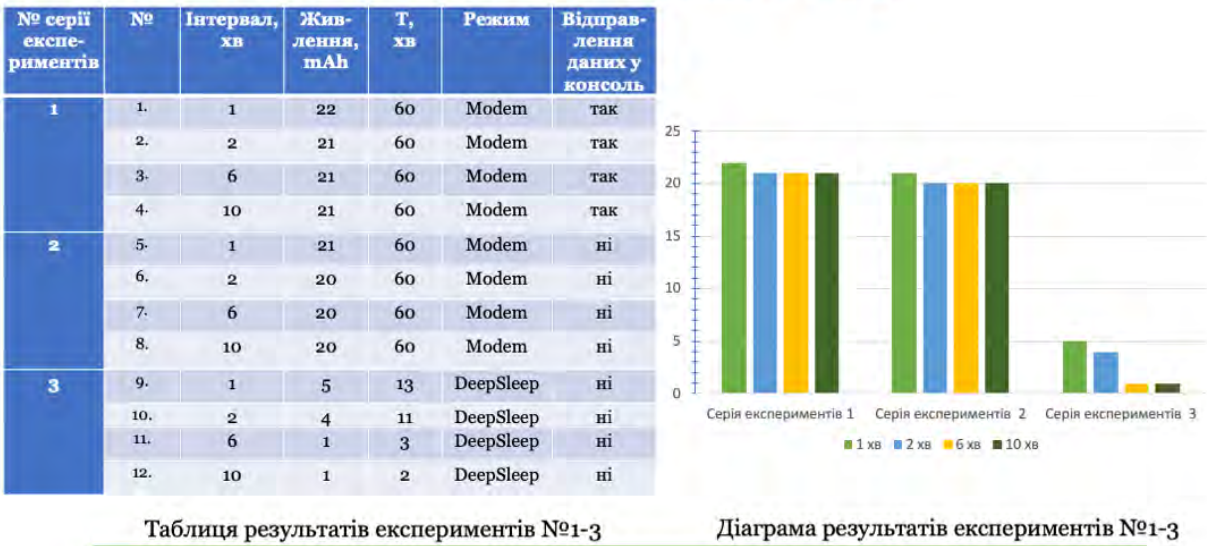
Візуалізація результатів роботи програми у хмарному сховищі



11

Рисунок Б.11 – Слайд 11

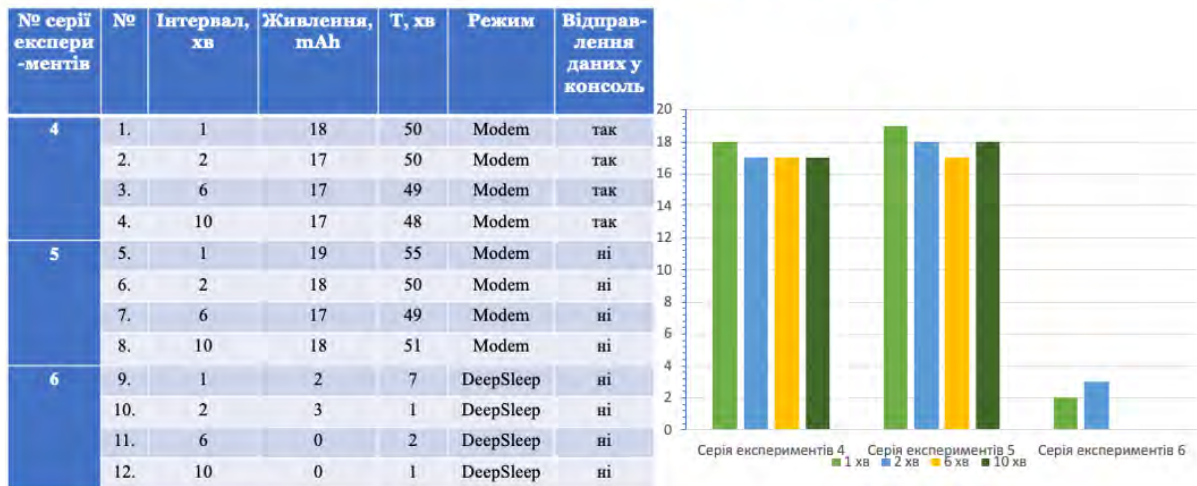
Результати комп’ютерних експериментів №1-3



12

Рисунок Б.12 – Слайд 12

Результати комп'ютерних експериментів №4-6



Таблиця результатів експериментів №4-6

Діаграма результатів експериментів №4-6

13

Рисунок Б.13 – Слайд 13

Висновки

- ✓ В результаті проведеної роботи було розроблено програмне забезпечення для системи передачі даних з портативного пристрою на основі Wi-Fi мікроконтролера ESP8266.
- ✓ Протягом роботи над проектом були успішно виконані всі поставлені задачі.
- ✓ Були проведені 6 серій експериментів при різних умовах відправки даних: як від різних джерел живлення (ПК та сонячна панель), так і з різними інтервалами часу.
- ✓ З використанням режиму DeepSleep та сонячної панелі живлення можливо оптимізувати і зменшити споживання електроенергії майже у 20 разів, порівнюючи зі стандартним режимом роботи.
- ✓ Основні положення та результати роботи було представлено на НПК Тиждень науки-2022.

14

Рисунок Б.14 – Слайд 14

Дякую за увагу!

15

Рисунок Б.15 – Слайд 15