МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Сумський державний університет
Факультет електроніки та інформаційних технологій
Кафедра електроніки, загальної та прикладної фізики

«До захисту допущено»
Завідувач кафедри

_____ Іван ПРОЦЕНКО
____ _____ 2023 р.

КВАЛІФІКАЦІЙНА РОБОТА
на здобуття освітнього ступеня бакалавр

зі спеціальності 171 Електроніка освітньої програми «Електронні інформаційні системи»
на тему: **Принципи організації MESH-мереж: топології та протоколи**

Здобувача (ки) групи ЕП-91 Пелих Радислав Костянтинович

Кваліфікаційна робота містить результати власних досліджень. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело.

_____ Радислав Пелих

Керівник ст.викладач, канд. ф.-м.н. Костянтин Тищенко _____

Суми – 2023

СУМСЬКИЙ ДЕРЖАВНИЙ УНІВЕРСИТЕТ
ФАКУЛЬТЕТ ЕЛЕКТРОНІКИ ТА ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ
Кафедра електроніки, загальної та прикладної фізики
Спеціальність 171 – Електроніка, освітньо-професійна програма
«Електронні інформаційні системи»

ЗАТВЕРДЖУЮ
Зав. кафедри ЕЗПФ

І.Ю.Проценко

«29» травня 2023 року

ІНДИВІДЦАЛЬНЕ ЗАВДАННЯ

НА КВАЛІФІКАЦІЙНУ РОБОТУ БАКАЛАВРА

Пелиха Радислава Костянтиновича

1. Тема роботи: Принципи організації Mesh мереж: топології та протоколи

затверджена наказом по університету від «15» травня 2023 р., № 0499-VI

2. Термін здачі студентом закінченої роботи: 09 червня 2023 року

3. Вихідні дані до роботи (актуальність, мета)

Розподілені безпровідні автоматизовані системи є основною складовою інтернету речей. Окремі вузли розподіленої системи зв'язуються у мережі, розгортання яких може бути реалізоване із застосуванням багатьох стандартних протоколів безпровідного зв'язку, кожен з яких матиме перевагу над іншими у залежності від функціоналу кінцевої системи, та апаратних засобів, які її реалізовують. Використання розподілених мереж, що здатні до самоорганізації (mesh-мереж) дозволяє будувати системи моніторингу зі значною площею покриття і стійкістю до динамічних змін топології. Окрім того, важливим є питання правильної організації маршрутизації трафіку та безпеки в середині мережі, що також може бути реалізовано із застосуванням поширених транспортних та безпекових протоколів.

Метою роботи є розробка програмного функціоналу окремого вузла mesh-мережі у рамках концепції Інтернету речей на базі мікроконтролерних модулів з підтримкою стандартних безпровідних протоколів зв'язку.

4. Зміст розрахунково-пояснювальної записки (перелік питань, що належить їх розробити)

1. Архітектура mesh-мереж в системах типу Інтернет речей
2. Протоколи безпровідного зв'язку для побудови розподілених мереж

3. Порівняння технологій для побудови безпровідних мереж моніторингу із аналізом переваг і недоліків кожної із них
4. Розробка програмного забезпечення вузла mesh-мережі.
5. Висновки.

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень)
Слайди № 1-2 – Загальна інформація
Слайди № 3-6 – Архітектура Mesh-мереж, особливості реалізації розподілених мереж
Слайди № 7-9 – Протоколи зв'язку в Mesh-мережах
Слайди № 10 – Методика експерименту
Слайди № 11-13 – Експериментальні результати
Слайд № 14 – Висновки

6. Дата видачі завдання 30.05.2023 р.

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів кваліфікаційної роботи магістрів	Термін виконання етапів роботи	Примітка
1.	Аналіз літературних даних	до 02.06.2023 р.	*вик*.
2.	Проведення експерименту, моделювання, розрахунків, обробка результатів	до 06.06.2023 р.	*вик*.
3.	Оформлення тексту кваліфікаційної роботи.	до 09.06.2023 р.	*вик*.
4.	Попередній захист роботи	12.06.2023 р., $10^{\underline{05}}$ (дистанційно)	*вик*.
5.	Захист роботи в екзаменаційній комісії	20-23.06.2023 р., $10^{\underline{05}}$ (дистанційно)	*вик*.

Здобувач вишої освіти Р.К. Пелих

Керівник роботи К.В. Тищенко

АНОТАЦІЯ

Кваліфікаційна робота викладена на 42 сторінках, зокрема, містить 17 рисунків, 1 таблицю, список використаних джерел із 18 найменувань.

Актуальність теми: застосування mesh-мереж насьогодні обширне настільки, що їх застосування охоплює майже всі сфери суспільних процесів, та галузей виробництва. Вони застосовуються не лише у галузі керування, а й інтелектуальних процесів, що охоплюють взаємодію людини з її оточенням. Такі системи характеризуються високим ступенем самоорганізації топології, що забезпечується використанням протоколів для mesh. Комірчаста структура мережі також забезпечує високу загальну відмововостійкість системи, оскільки маршрутизація трафіку може динамічно перерозподілятись між найменш завантаженими вузлами.

Мікроконтролерна платформа ESP може бути використана для побудови mesh-мереж, оскільки вона підтримує підключення датчиків фізичних величин, таких як температурні, гігрометри, магнітометри, фотометри; а також дозволяє реалізувати взаємодію з іншими вузлами mesh за допомогою стандартизованих мережевих комунікаційних протоколів, таких, як Wi-Fi, LORA, та інших. Такі контролери можуть бути запрограмовані для виконання окресленого кола задач за найбільш оптимальним алгоритмом.

Мета кваліфікаційної роботи бакалавра полягає в розробці схеми та програмного забезпечення розподіленої mesh-мережі моніторингу оточуючого середовища на базі мікроконтролерної платформи ESP-8266.

Під час виконання роботи використовували онлайн середовище проєктування EasyEDA з інтегрованими модулями платформи ESP-8266. Програмний код розроблявся мовою C++ в середовищі програмування Arduino IDE.

У результаті проведеного проєктування та розробки програмного забезпечення отримано mesh-мережу моніторингу фізичних величин на основі мікроконтролерних модулів ESP з апаратною підтримкою Wi-Fi.

Ключові слова: mesh-мережа, мережевий протокол, топологія мережі, марштутизація, мікроконтролер.

ЗМІСТ

РОЗДІЛ 1.
ОСОБЛИВОСТІ АРХІТЕКТУРИ MESH-МЕРЕЖ (ЛІТЕРАТУРНИЙ ОГЛЯД)

1.1. Огляд розповсюджених мережевих архітектур

У сучасному світі, де зв'язок та обмін інформацією є невід'ємною складовою нашого повсякденного життя, мережеві архітектури відіграють важливу роль у забезпеченні надійної та ефективної комунікації. Вони використовуються в різних сферах, від домашніх мереж і малих офісів до великих підприємств, дата-центрів та глобальних мереж Інтернет.

Мережа – це група комп’ютерів або апаратних пристроїв (вузлів, nodes), які з’єднані між собою каналами зв’язку, які можна використовувати для передачі інформації між пристроями. Кожна окрема мережа має фізичну реалізацію (спосіб з'єднання вузлів, топологію), вузли підключаються до мережі, і додатково використовує апаратне забезпечення, відповідне даній структурі. Традиційні мережі з’єднують обмежену кількість вузлів [1].

Мережа і Інтернет є двома взаємопов'язаними концепціями, де мережа є фундаментальною складовою Інтернету. Мережа дозволяє нам побудувати комунікаційну інфраструктуру, яка з'єднує пристрої та користувачів, дозволяючи обмінюватися даними. Інтернет в свою чергу розширює цю ідею, з'єднуючи мільйони мереж у всесвітню систему зв'язку, що надає доступ до широкого спектру ресурсів і послуг.

Інтернет – це сукупність мереж, які використовують однаковий набір мережевих протоколів. Мережеві протоколи — це правила, які визначають формат даних, які надсилаються через мережу. Фізична структура різних мереж, які є частиною Інтернету, може відрізнятися. Такі різнорідні мережі з’єднані одна з одною маршрутизаторами, які пересилають пакети з однієї мережі в іншу відповідно до адрес призначення, одночасно перетворюючи пакети між форматами кожної мережі. Маршрутизатори підтримують міжмережевий зв'язок [1].

Рисунок 1.1 – Розповсюджені мережеві архітектури.

Ethernet був першою технологією локальної мережі (LAN) і залишається найважливішою технологією у світі. Він був розроблений Xerox PARC на початку 1970-х років. Оригінальний Ethernet дозволяв комп’ютерам, розташованим на відстані сотень ярдів один від одного, обмінюватися повідомленнями. Цю відстань було збільшено до тисяч ярдів завдяки додаванню повторювачів і мостів між кількома локальними мережами. Тому він підходить для підключення комп'ютерів в університетських будівлях або на території кампусу [2].

Архітектура Ethernet базується на концепції з’єднання кількох комп’ютерів за допомогою довгих кабелів (іноді їх називають Ethernet), утворюючи структуру шини. Кожен комп’ютер оснащений адаптером Ethernet, який містить унікальну 48-розрядну адресу для цього комп’ютера. Кожен комп'ютер підключений до Ethernet через трансивери, що утворюють логічну букву "T". Трансивер отримує повідомлення Ethernet через кабель, шукає адресу, передає повідомлення на комп’ютер, якщо адреса збігається, і передає його по кабелю, якщо адреса не

збігається [2].

(Логічний) кабель Ethernet утворює локальну мережу. Дві локальні мережі можна з'єднати за допомогою так званих мостів. Міст - це спеціальний комп'ютер, який з'єднує дві локальні мережі. Коли він отримує повідомлення, він визначає, у якій із двох мереж знаходиться адресований комп’ютер, і пересилає повідомлення у відповідну мережу [2].

Повідомлення — це записи змінної довжини, що варіюються від 64 до трохи більше тисячі п’ятсот байтів (або «октетів») даних. Інтернет формується шляхом з'єднання двох або більше локальних мереж (зазвичай Ethernet) через маршрутизатори. Після завершення IP-пакет інкапсулюється як частина даних кадру Ethernet (або іншого протоколу локальної мережі) [2].

Ethernet спочатку працював зі швидкістю 10 Мбіт/с. Поточний стандарт становить 100 Мбіт/с, і ми можемо очікувати, що швидкість досягне Гбіт/с у найближчі кілька років. Тому, оскільки Ethernet продовжує розвиватися, багато людей вважають, що він більше не потребуватиме складніших мережевих архітектур [2].

Token Ring визначається як протокол зв’язку в локальній мережі, де всі станції в мережі з’єднані кільцевою топологією. На рисунку 1.2 можемо побачити схематичне зображення роботи Token Ring. Токен - це невелика область в 3 байти. Він їздить між кільцевими станціями. Станція може надсилати ділянку лише за наявності токена. Після успішної передачі кадру він звільняє токен для подальшої передачі іншими станціями [3].

Appletalk — це стек протоколів, розроблений компанією Apple Computer для комп’ютерних мереж. Спочатку він входив до Macintosh (1984), але тепер компанія відмовилася від нього на користь TCP/IP. Існує дві версії AppleTalk: AppleTalk Phase I та AppleTalk Phase II [4].

Версія AppleTalk Phase I розроблена для невеликих робочих груп і не має можливості працювати у великих мережах. Мережа AppleTalk Phase I обмежена 135 хостами [4].

У версії AppleTalk Phase II це число було збільшено до 253 у сегменті мережі,

що дозволило підтримувати великі розміри мережі.

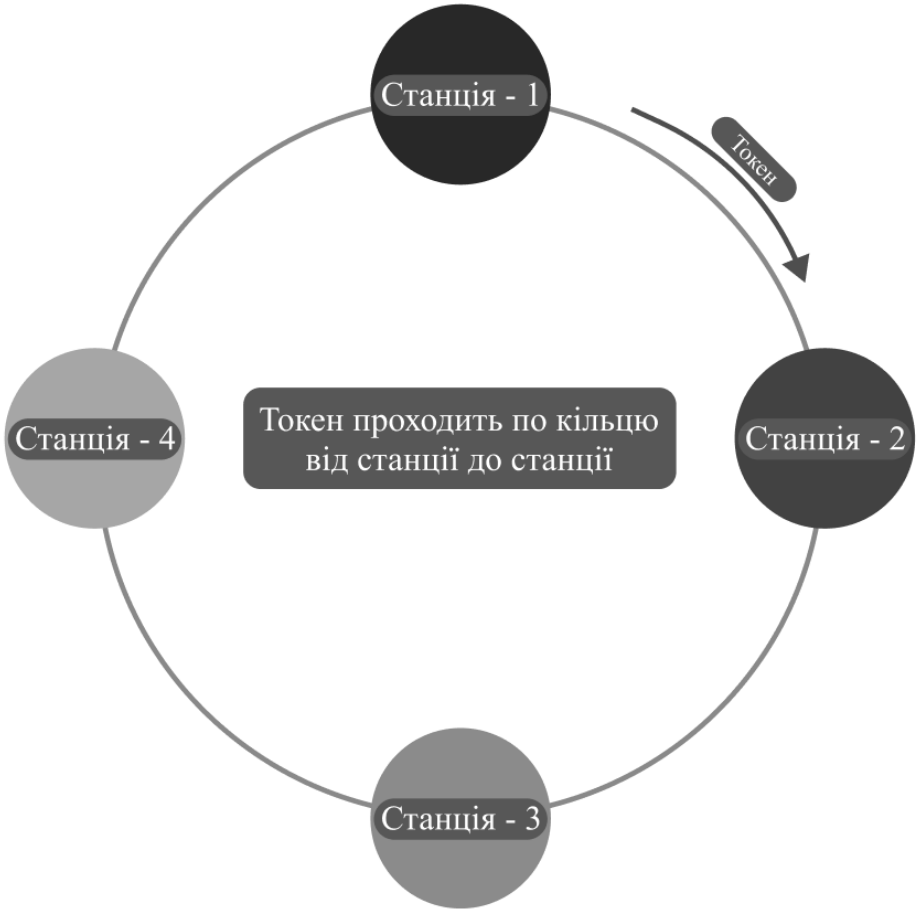

Рисунок 1.2 – Схема роботи Token Ring. Адаптовано з роботи [2]

Apple Talk був розроблений як клієнт-серверна розподілена мережева система. Іншими словами, користувачі спільно використовують такі мережеві ресурси, як файли та принтери [4].

Як і TCP/IP, AppleTalk використовує 32-розрядні адреси; які IP-адреси, адреси AppleTalk складаються з двох компонентів: мережевої адреси та адреси комп'ютера. На відміну від IP, довжина кожного компонента фіксована 16 з 32 біт виділяються для мережевої адреси, а останні 16 біт використовуються для ідентифікації комп'ютера Мережа AppleTalk підтримує процедуру узгодження мережевої адреси комп'ютера. Ця процедура усуває необхідність для адміністратора вказувати адресу явно (якщо потрібно, адресу можна вказати явно або запросити з діапазону, але зазвичай в цьому немає необхідності) [4].

ArcNET Архітектура представлена двома основними топологіями: шинна та

зіркова. У якості середовища передачі використовується коаксіальний кабель RG-62 з хвильовим опором 93 Ом, обтиснутий на BNC-вилки з відповідним діаметром закладення [5].

Архітектура ARCnet використовує процес «реконфігурації» мережі для додавання або видалення станцій у мережі, коли вони стають активними або неактивними. Зміна конфігурацій є нормальною частиною роботи ARCnet. Однак зміни конфігурації також можуть відбутися, коли виходить з ладу мережева інтерфейсна карта чи хаб, або коли виходить з ладу кабель [4].

Бездротова меш мережа - це тип однорангової бездротової мережі з самовідновленням, самоконфігурацією [6]. Без дорогих стаціонарних базових станцій WMN можна встановити швидко, легко та гнучко за низьку вартість.
Бездротові меш мережі в свою чергу ділиться на три категорії:

- Інфраструктурна меш-мережа
- Клієнтська меш-мережа
- Гібридна меш-мережа

1.2. Переваги та недоліки Mesh-мереж

В Mesh-мережах немає центрального хабу, комутатора чи комп’ютера, який обробляє весь трафік комп’ютера. Натомість кожен пристрій у сітчастій мережі може спілкуватися з будь-яким іншим пристроєм. Ці кілька ретрансляторів можуть швидко спрямовувати трафік між пристроями. Це створює сітчастий шаблон з’єднання.

Повна mesh-мережа описана вище. Часткова мережа є більш обмеженою. В ній секції вузлів будуть повністю з’єднані між собою, але ці секції будуть спілкуватися через комутатори або центральні хаби.

Бездротова меш мережа це тип однорангової бездротової мережі з самовідновленням, самоконфігурацією [7].
Бездротові меш мережі в свою чергу ділиться на три категорії:

- Інфраструктурна меш-мережа

– Клієнтська меш-мережа
– Гібридна меш-мережа

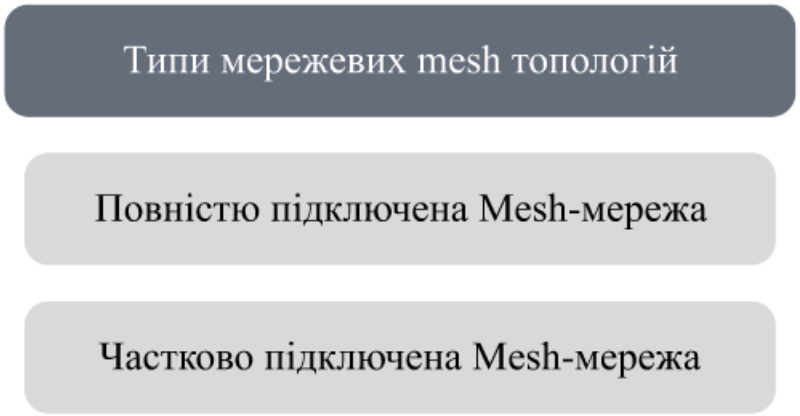

Рисунок 1.3 – Типи мережевих mesh топологій.

Переваги Mesh-мереж:

Легка масштабованість. Mesh-мережі не потребують додаткових маршрутизаторів. Замість цього кожен вузол діє як маршрутизатор. Це означає, що ви можете швидко та легко змінити розмір мережі.

Наприклад, ви можете легко додати купу технологій до конференц-залу на короткий період часу. Ноутбуки, принтери та інше можна перенести в кімнату, і вони автоматично підключаться до мережі.

Навіть не комп'ютерні пристрої можуть отримати користь від цього типу мережі. Наприклад, сітчасту мережу можна використовувати для освітлення. Це дозволяє просто додавати джерела світла, як вам подобається, з можливістю керувати всією мережею з будь-якого місця.

Стійкість до проблем. Кожен вузол у сітчастій мережі як отримує, так і транслює інформацію. Це забезпечує значну надлишковість вузлів, яка допомагає підтримувати роботу мережі навіть у разі виникнення проблеми. Якщо один вузол виходить з ладу, мережа може використовувати інші вузли для завершення сітки.

Легко додати діапазон. Додавання діапазону до сітчастої мережі зазвичай не

є проблемою. Ви просто підключаєте вузли до шлюзів, які дозволяють повідомленням проходити до решти мережі. Крім того, сітчасті мережі можуть самостійно оптимізуватись і знаходити найшвидший маршрут для доставки повідомлення.

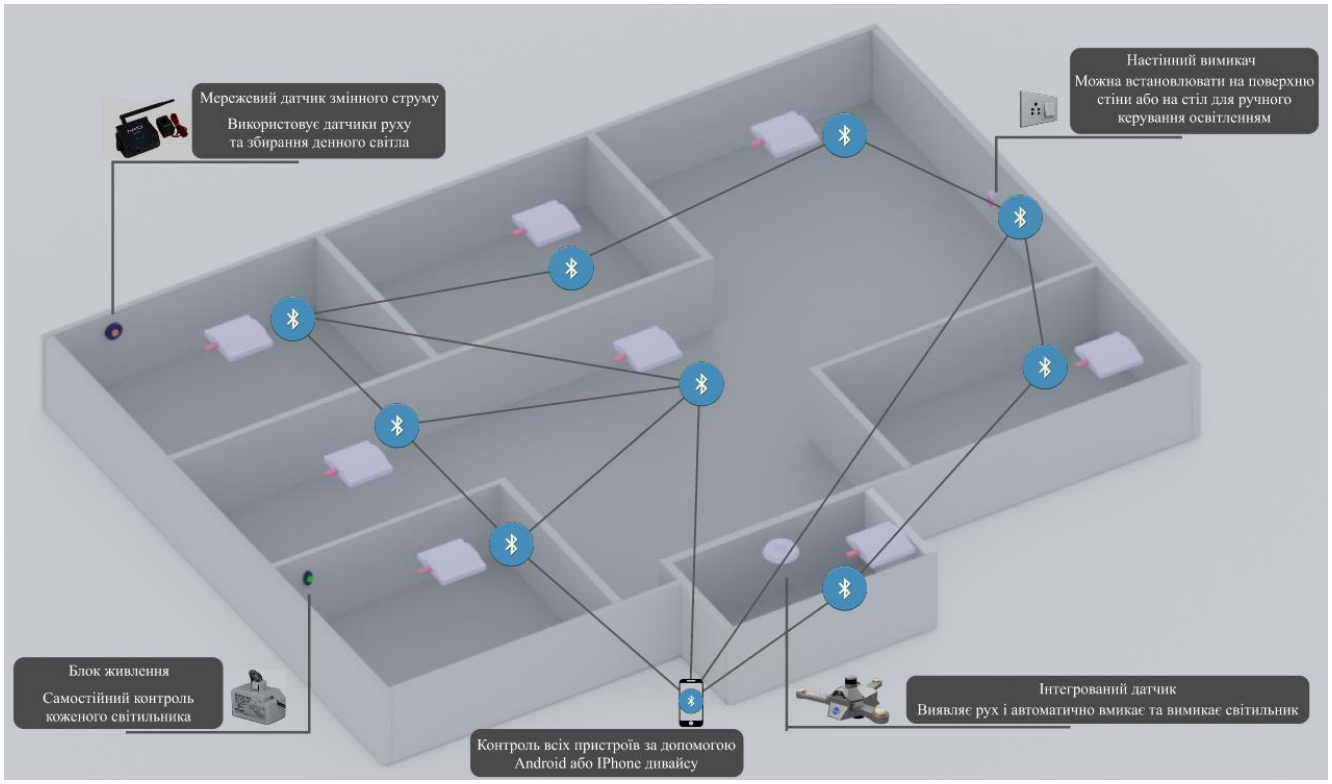

Рисунок 1.4 – Схематична побудова взаємодії пристроїв в бездротовій Mesh-мережі Smart Bluetooth. Адаптовано з роботи [7]

Недоліки Mesh-мереж:

Збільшення робочого навантаження для кожного вузла. Кожен вузол у сітчастій мережі має велике навантаження. Крім надсилання повідомлень, вузол також повинен працювати як маршрутизатор. Кожен вузол, доданий до сітчастої мережі, також робить систему складнішою.

Вузли повинні будуть відстежувати повідомлення від п’яти до 10 сусідніх вузлів, залежно від конфігурації. Кожне повідомлення, яке має передати вузол, експоненціально збільшує обсяг даних, які він також має обробити. Збільшення діапазону системи може додати різноманітні небажані ускладнення через відповідне збільшення навантаження на дані.

Мережі з низьким енергоспоживанням можуть мати проблеми із затримкою. Якщо ви використовуєте глобальну мережу з низьким енергоспоживанням (LPWAN) у вас можуть виникнути проблеми із затримкою. Затримка – це час, потрібний повідомленню для проходження від вузла до шлюзу. Більшість мереж LPWAN не мають процесуальної можливості для своєчасної обробки всіх необхідних передач даних.

Якщо затримка є проблемою, можливо, доведеться оновити всю сітчасту мережу. Більша пропускна здатність, пам'ять і потужність для кожного вузла можуть збільшити швидкість передачі повідомлень. Звичайно, ці оновлення також коштують більше грошей.

Mesh-мережі чудові, якщо у вас (або у вашої організації) є багато грошей і багато часу для налаштування. Але якщо кожен новий вузол завдає удар по бюджету, цей тип мережі може бути повільнішим і обмеженішим.

Початкове налаштування мережі може бути складним. Коли сітчаста мережа запущена та працює, додавати вузли досить просто. Але реалізація сітчастої мережі з нуля зазвичай набагато складніша та трудомісткіша, ніж створення чогось більш традиційного.

Проблеми із затримкою передачі визначатимуть, де вам потрібно розмістити вузли. Вам може знадобитися додати виділені вузли виключно для цілей пересилання повідомлень. Але це може перетворитися на логістичну проблему, бо можливо, вам доведеться додати обладнання по всій локації розташування мережі, щоб повідомлення могли правильно та швидко маршрутизуватися.

Збільшене енергоспоживання для кожного вузла. Коли кожному вузлу доручено діяти як кінцева точка і маршрут це збільшує робоче навантаження. Кожен вузол потребуватиме більше енергії, ніж зазвичай, щоб працювати правильно.

Ймовірно, це не є великою проблемою, якщо вузол великий і підключений безпосередньо до електричної системи. Але це може стати проблемою для маленьких вузлів, що живляться від батареї.

Як правило, системи безпеки та освітлення можуть спричиняти проблеми,

якщо їх неправильно налаштовано. Датчики для системи безпеки потребують достатньої потужності, щоб передавати дані з кімнати в кімнату і навіть між поверхами. Це більш складне завдання, ніж у традиційних системах, де датчику потрібна лише достатня потужність, щоб досягти панелі керування.

1.3. Протоколи для організації Mesh

Існує понад 70 конкуруючих схем для маршрутизації пакетів у Mesh-мережах. Наведемо деякі з них:

– Маршрутизація на основі асоціативності (ABR)

– AODV (Ad hoc On-Demand Distance Vector)

– B.A.T.M.A.N. (Better Approach to Mobile Ad-hoc Networking)

– Babel (протокол дистанційної маршрутизації для IPv6 і IPv4 із властивостями швидкої конвергенції)

– Динамічна NIx-векторна маршрутизація (DNVR)

– DSDV (Destination-Sequenced Distance-Vector Routing)

– DSR (Dynamic Source Routing)

– HSLS (Hazy-Sighted Link State)

– HWMP (Hybrid Wireless Mesh Protocol, обов’язковий протокол маршрутизації за замовчуванням IEEE 802.11s)

– Інфраструктурний протокол бездротової сітки (IWMP) для інфраструктурних сітчастих мереж GRECO UFPB-Бразилія[39]

– ODMRP (On-Demand Multicast Routing Protocol)

– OLSR (Optimized Link State Routing protocol)

– OORP (OrderOne Routing Protocol) (Протокол мережевої маршрутизації OrderOne)

– OSPF (Open Shortest Path First Routing)

– Протокол маршрутизації для мереж із низьким енергоспоживанням і мережами з втратами (протокол IETF ROLL RPL, RFC 6550)

– PWRP (Протокол прогнозованої бездротової маршрутизації)[40]
– TORA (Тимчасово впорядкований алгоритм маршрутизації)
– ZRP (протокол зонної маршрутизації)

У Mesh-мережах можна використовувати стандартні протоколи автоконфігурації, такі як DHCP або автоконфігурація без збереження стану IPv6 [8].

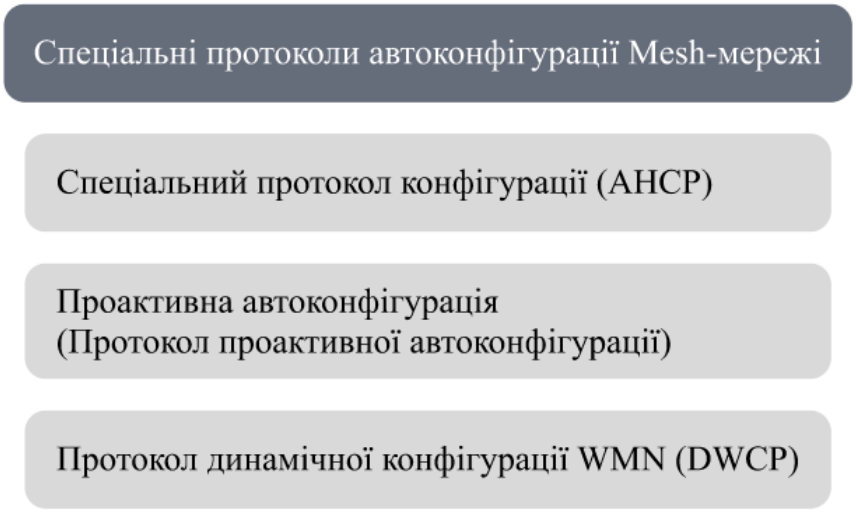

Рисунок 1.5 – Спеціальні протоколи автоконфігурації Mesh-мережі.

1.3.1. Протокол B.A.T.M.A.N.

B.A.T.M.A.N. – це проактивний протокол маршрутизації для бездротових спеціальних сітчастих мереж, включаючи мобільні спеціальні мережі (MANET), але не обмежуючись ними. Протокол проактивно зберігає інформацію про існування всіх вузлів у сітці, які доступні через канали зв’язку з одним або кількома переходами [8]. Стратегія B.A.T.M.A.N. полягає у визначенні для кожного пункту призначення в сітці одного сусіда до якого лише один перехід, який може бути використаний як найкращий шлюз для зв’язку з вузлом призначення. Для того, щоб виконати IP-маршрутизацію з кількома переходами, таблиця маршрутизації вузла повинна містити локальний шлюз для кожного хоста або мережевого маршруту.

Дізнаватися про найкращий наступний «стрибок» для кожного пункту призначення – про це і дбає алгоритм B.A.T.M.A.N. Немає необхідності з'ясовувати або розраховувати повний маршрут, що робить можливим дуже швидке та ефективне впровадження.

Бездротові Mesh-мережі мають особливі труднощі на відміну від дротових мереж: пакети даних можуть і будуть втрачатися в шумних місцях. B.A.T.M.A.N. вирішує проблеми шляхом статистичного аналізу втрати пакетів протоколу та швидкості розповсюдження та не залежить від інформації про стан чи топологію з інших вузлів. Метадані, що містяться в отриманому протокольному трафіку можуть бути запізнілі, застарілими або взагалі втрачені, тому рішення щодо маршрутизації базуються на знаннях про наявність або відсутність інформації. B.A.T.M.A.N . пакети протоколу містять лише дуже обмежену кількість інформації і тому дуже малі. Втрачені пакети протоколу через ненадійні з’єднання не компенсуються за допомогою резервування, але виявляються та використовуються для кращого рішення щодо маршрутизації. B.A.T.M.A.N. вибирає найнадійніший маршрут на основі рішення щодо маршрутизації наступного переходу окремих вузлів [8]. Цей підхід на практиці показав, що він є надійним і не створює зациклювань.

1.3.2. Протокол HWMP

HWMP є частиною IEEE 802.11s, є базовим протоколом маршрутизації для бездротової Mesh-мережі. Він заснований на AODV (RFC 3561) і маршрутизації на основі дерева. Він ґрунтується на протоколі Peer Link Management, за допомогою якого кожна точка мережі виявляє та відстежує сусідні вузли. [9] Якщо будь-яке з них підключено до дротового транспортного зв’язку, в HWMP немає потреби, бо він вибирає шляхи із вже зібраних шляхом компіляції всіх однорангових точок сітки в одну складену карту [10].

Протокол HWMP є гібридним, оскільки він складається з проактивного протоколу ієрархічної маршрутизації на основі дерева та логіки на вимогу,

заснованої на протоколі Ad-hoc On Demand Vector (AODV). На відміну від класичної маршрутизації на основі IP (рівень ISO 3), протокол HWMP базується на рівні ISO 2 (на основі MAC-адрес).

HWMP має на меті замінити пропрієтарні протоколи, що використовуються такими постачальниками, як Meraki, для тієї ж мети, дозволяючи однорангову участь через прошивку маршрутизатора з відкритим кодом. Реалізація 802.11s (open80211s) з відкритим кодом була інтегрована в ядро Linux компанією Cozybit Inc. FreeBSD підтримує HWMP, починаючи з FreeBSD 8.0 [11].

1.3.3. Протокол Optimized Link State Rout (OLSR)

Робоча група IETF MANET представила протокол **Optimized Link State Rout (OLSR)** для мобільних мереж Ad-Hoc. Протокол є оптимізацією алгоритму чистого стану посилань (Pure Link State). Ключовою концепцією, яка використовується в протоколі, є концепція багатоточкових реле (MPR). Набір MPR вибирається таким чином, щоб він охоплював усі вузли, розташовані за два переходи. Вузол знає про своїх сусідів і сусідів із двома переходами методом «HELLO»-повідомлень – це повідомлення, яке періодично генерує кожен вузол для отримання вузлів, які він чує. Вузол N, який вибрано сусідами як багатоточковий ретранслятор, періодично генерує повідомлення TC (Topology Control), повідомляючи інформацію про те, хто вибрав його як MPR. Окрім періодичної генерації TC, вузол MPR також може створювати повідомлення TC, як тільки він виявляє зміну топології в мережі. Повідомлення TC приймається та обробляється всіма сусідами N, але лише ті сусіди, які перебувають у наборі MPR N, повторно передають його. За допомогою цього механізму всі вузли інформуються про підмножину всіх зв’язків – зв’язки між MPR і селекторами MPR у мережі [12].

Отже, на відміну від класичного алгоритму стану посилань, замість усіх посилань оголошуються лише невеликі підмножини посилань. Для обчислення маршруту кожен вузол розраховує свою таблицю маршрутизації за допомогою алгоритму найкоротшого переходу на основі часткової топології мережі, яку він

вивчив. Вибір MPR є ключовим моментом у OLSR. Чим менший набір MPR, тим менше накладних витрат протокол вносить Запропонована евристика і в для вибору MPR полягає в ітераційному виборі сусіда з одним переходом, який досягає максимальної кількості непокритих сусідів з подвійним переходом як MPR. Якщо є зв'язок, той із вищим ступенем (більше сусідів) [12].

1.4. Приклади реалізації мереж

Прикладами реалізації меш мережі може слугувати дослідження «Implementation and Analysis of Routing Protocols for LoRa Wireless Mesh Networks». Топологія цієї мережі була розподілена вздовж вулиць міста Цюріх, Швейцарія.

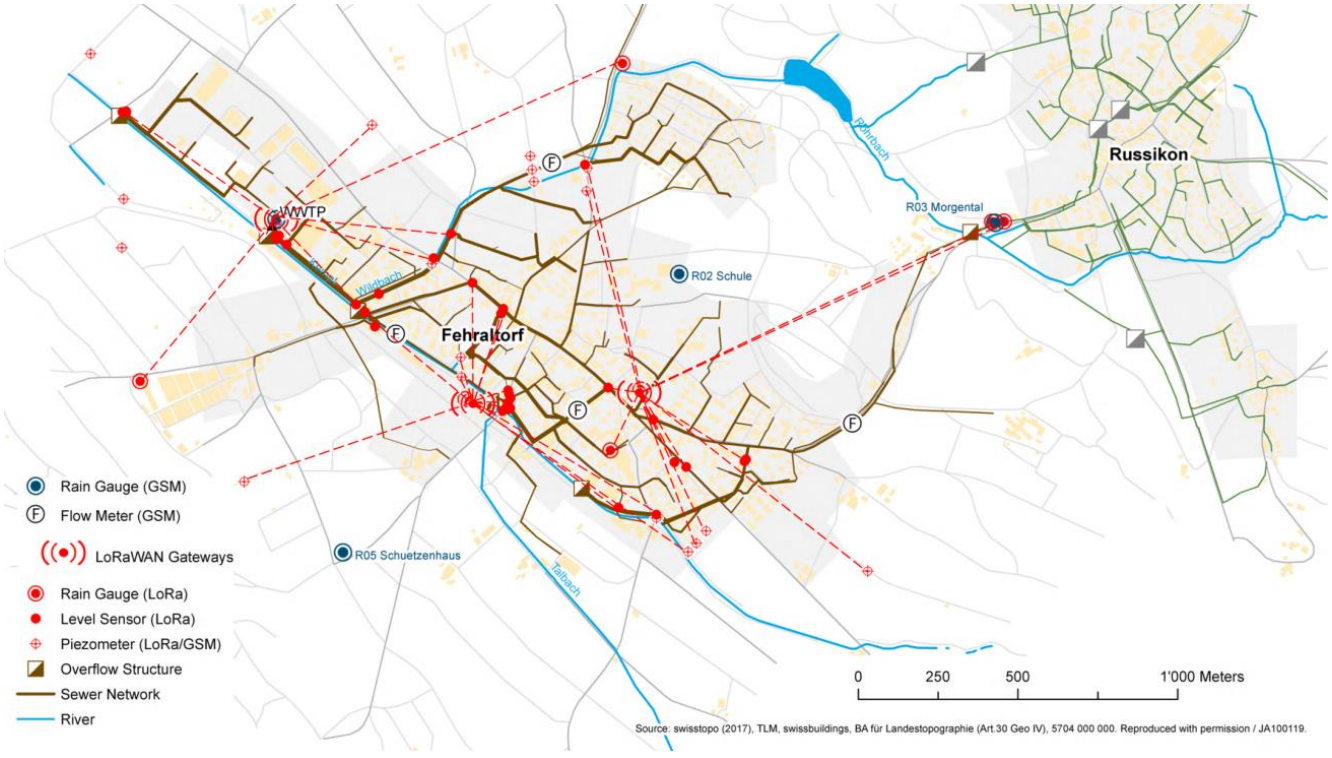

Рисунок 1.6 – Макет бездротової сенсорної мережі, розгорнутої у Феральторфі (Цюріх, Швейцарія), з використанням стандартної інфраструктури LoRaWAN; Червоні маркери вказують положення датчиків у каналізаційній мережі та за її межами; Червоні пунктирні лінії вказують на радіолінії від трьох центральних шлюзів до сенсорних вузлів.. Адаптовано з роботи [13]

Моделювання мало на меті оцінити продуктивність протоколів маршрутизації, коли вони представлені в реальному прикладному сценарії, в контексті розумних міст, у яких топологія мережі змінюється через мобільність вихідного вузла.

У розглянутому сценарії вузол-джерело, який періодично надсилає дані на шлюз, був мобільним і переміщався деякими маршрутами вулиць. Топологія мережі, що використовується, складається з 25 вузлів, і кожен вузол має радіус покриття приблизно 1,5 км із бездротовим радіозв’язком LoRa [13].

Наразі це тільки дослідження, що виконує конкретні завдання. Реалізація проекту не відбулася, вона була тільки змодельована.

Mesh-мережа в місті Нью-Йорк, США. В рамках проекту "LinkNYC" було встановлено мережу мереж, яка поєднує бездротові точки доступу та гнучку комунікаційну інфраструктуру. Ця мережа дозволяє містянам та відвідувачам міста отримувати безкоштовний інтернет, а також використовувати послуги, пов'язані з цифровою рекламою та інформаційними сервісами. Мережа побудована на принципах mesh-технології, що дозволяє забезпечити стабільний зв'язок у всьому місті.

Bluetooth SIG (Special Interest Group) є організацією, яка відповідає за розробку та стандартизацію технології Bluetooth, що використовує технологію BLE (Bluetooth Low Energy – Bluetooth з низьким енергоспоживанням). У рамках своєї діяльності, Bluetooth SIG розробила Bluetooth Mesh - спеціальну версію технології Bluetooth, яка дозволяє організовувати mesh-мережі.

Bluetooth Mesh - це мережевий протокол, який дозволяє створювати великі мережі, включаючи вузли, такі як смартфони, комп'ютери, освітлювальні пристрої, датчики та інші пристрої, що підтримують Bluetooth. Ця технологія дозволяє взаємодіяти між пристроями, які знаходяться у відносно великій відстані один від одного, шляхом передачі повідомлень через проміжні вузли в мережі.

Bluetooth Mesh має кілька ключових особливостей, які роблять його ефективним інструментом для створення мереж у різних сферах застосування. Деякі з цих особливостей включають:

Масштабованість. Bluetooth Mesh може підтримувати велику кількість пристроїв у мережі, що робить його ідеальним для розгортання у великих просторах, таких як офісні приміщення, фабрики або громадські приміщення.

Надійність. Завдяки проміжним вузлам у мережі, Bluetooth Mesh може автоматично виявляти та обходити перешкоди або несправні вузли, забезпечуючи надійний шлях комунікації між пристроями.

Гнучкість. Bluetooth Mesh дозволяє встановлювати різні типи вузлів у мережі, такі як вузли-контролери, вузли-пристрої або вузли-посередники, що надає гнучкість у використанні та розгортанні.

Одна з переваг нового протоколу – він покликаний об’єднати всі IoT мережі, які використовуються у розумних будинках.

Компанії, які використовують та розповсюджують технологію Bluetooth Mesh в своїх пристроях:

Signify (раніше відома як Philips Lighting): Signify використовує Bluetooth Mesh у своїх розумних освітлювальних системах, таких як Philips Hue. Це дозволяє користувачам керувати освітленням у своїх приміщеннях з використанням смартфонів або голосових асистентів.

Nordic Semiconductor: Nordic Semiconductor є провідним постачальником чіпів Bluetooth і використовує Bluetooth Mesh у своїх продуктах, що орієнтовані на IoT (Internet of Things) ринок. Їх чіпи забезпечують можливість побудови мереж з великою кількістю пристроїв.

Silicon Labs: Silicon Labs також надає рішення на основі Bluetooth Mesh для розумного освітлення, індустріального IoT, управління енергоефективністю та інших сфер. Вони пропонують чіпи, програмне забезпечення та інструменти для розробки мереж на основі Bluetooth Mesh.

Xiaomi: Xiaomi, китайський виробник електроніки та смартфонів, також використовує Bluetooth Mesh у своїх продуктах. їх розумні пристрої, такі як розумні лампи та вимикачі, підтримують зв'язок через мережу Bluetooth Mesh, що дозволяє користувачам керувати ними зі своїх смартфонів або голосових асистентів.

РОЗДІЛ 2.
ПРОТОКОЛИ ТА МАРШРУТИЗАЦІЯ В MESH-МЕРЕЖАХ

2.1. Моніторинг датчиків в підземній інфраструктурі за допомогою Mesh-мережі

Щоб розібрати варіативність реалізації Mesh-мережі, було розглянуто розроблений концепт збору точної інформації у реальному часі про продуктивність міських систем водовідведення із застосуванням Mesh-мереж.

Для виявлення, передбачення та керування критичними ситуаціями навантаження, такими як міські потокові повені та переповнення каналізації, необхідно мати точні дані. Хоча нові бездротові технології передачі даних забезпечують ефективний обмін інформації з високою продуктивністю на поверхні, важко досягти великого охоплення для підземних або внутрішніх застосувань через фізичні та топологічні обмеження, особливо в щільних міських районах. У цій роботі спочатку обговорюються обмеження дальності стандарту LoRaWAN на основі систематичної оцінки довготривалої роботи мережі датчиків, що моніторять динаміку процесів в каналізації [13].

Аналізи показують втрату даних у пакетах в середньому у п'ять разів більшу для вузлів підземних комунікацій, яка стабільно зростає зі збільшенням відстані до шлюзу. Далі пропонується новий концепт мережі LPWAN на основі технології LoRa, що покращує надійність, ефективність та гнучкість передачі в умовах обмеженої дальності за допомогою мережевої багатопередачної маршрутизації і забезпечує точну синхронізацію часу за допомогою опційного сигналу GPS або DCF77 довгих хвиль [13].

Демонструється корисність нової розробленої концепції шляхом оцінки продуктивності радіопередачі під час двох незалежних повномасштабних польових випробувань. Результати випробувань показують, що синхронна LoRa мережа множинних переходів відчутно перевершує стандартну техніку LoRaWAN за надійністю доставки пакетів при передачі з критичних точок. Отже, очікується, що

такий підхід загалом полегшить збір даних з місць, до яких важко отримати доступ, таких як підземні області.

Таблиця 1.1 – Налаштування параметрів LoRa для LoRaWAN (EU863-870) і синхронний зв'язок LoRa mesh.

Параметри		**LoRaWAN (EU863-870)**	**Синхронна сітка LoRa**
Коефіцієнт поширення	SF	7-12 (Варіативно)	9
Пропускна здатність	BW	125 kHz	125 kHz
Довжина преамбули	nPreambol Symbols	8	8
Потужність передачі	P_{Tx}	Варіативно	+14 dBm
Швидкість кодування	CR	4/5	4/5
Перевірка SRS		Водозбір і корисне навантаження	Водозбір і корисне навантаження

Розроблений концепт мережі LoRa з мережевою багатопередачною маршрутизацією та точною синхронізацією часу може ефективно вирішити проблеми передачі даних в умовах обмеженої дальності, особливо в підземних областях. Такий підхід може бути корисним для збору даних з важкодоступних місць і допомогти у виявленні та керуванні критичними ситуаціями в міському водовідведенні.

2.2. Bluetooth Mesh-мережа з низьким споживанням енергії

BE-Mesh (Bluetooth Low Energy-Meshed network) для BLE (Bluetooth Low Energy), дозволяє створювати мережеві з'єднання між бездротовими пристроями з

використанням мережі типу меш, як у режимі одного переходу, так і у режимі багатьох переходів.

Виходячи з класичної парадигми Master/Slave Bluetooth, було ідентифіковано два нових шари, що базуються на стеку BLE, які дозволяють кінцевому користувачеві швидко налаштовувати бажану топологію мережі, приховуючи складність та деталі низького рівня стеку BLE. Також було прототиповано відкриту бібліотеку для Android, яка реалізує запропоновану парадигму комунікації, а також додаток для Android, який дозволяє обмінюватися текстовими повідомленнями в мережі типу меш. Продемонстровано, як BE-Mesh дозволяє створювати спільний доступ до Інтернету з усієї мережі через один пристрій, підключений до Інтернету [14].

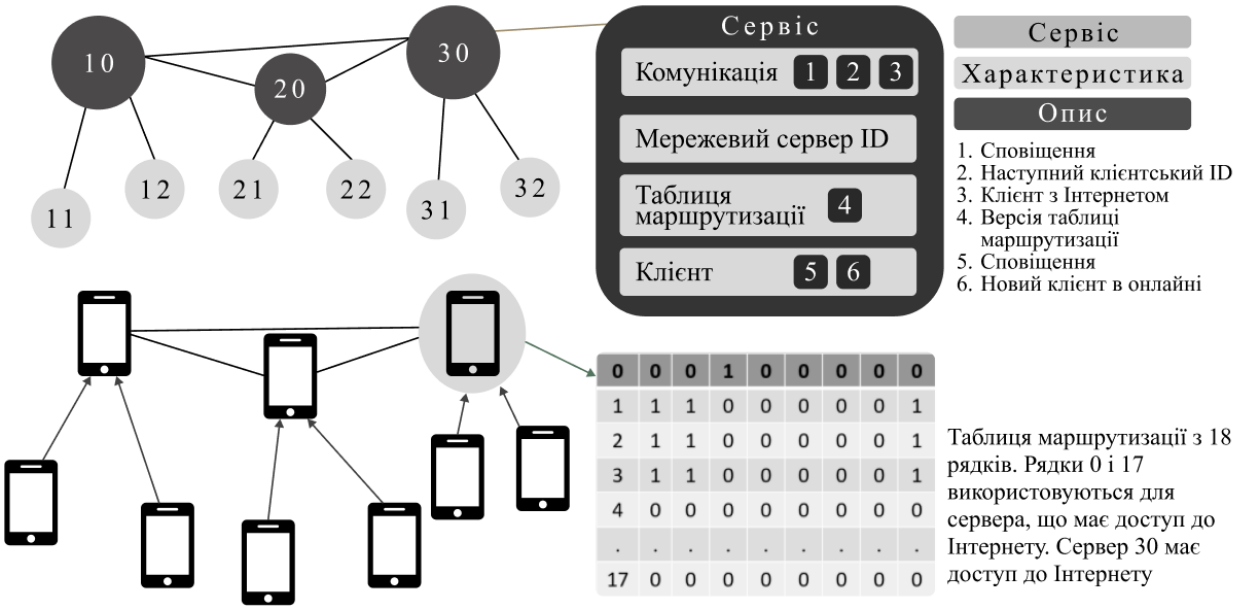

Рисунок 2.1 – Приклад топології з описом сервісу, сервера та таблицею маршрутизації клієнта BE-Mesh; Клієнт сірого кольору є унікальним, що має прямий доступ до Інтернету. Адаптовано з роботи [14]

Структура даних (Серверна сторона):

Зв'язок. Ця характеристика використовується для забезпечення обміну повідомленнями між клієнтами і сервером, а також для спілкування на рівні

сервера. Ця функціональність має три основні дескриптори:

– Дескриптор сповіщення: Цей дескриптор дозволяє клієнтам підписатися на характеристику, щоб отримувати сповіщення кожного разу, коли значення змінюється. Це дає можливість клієнтам отримувати актуальні повідомлення.

– Дескриптор наступного ідентифікатора: Цей дескриптор містить значення наступного ідентифікатора, який буде призначений новому клієнту. Він обмежений до максимум семи ідентифікаторів.

– Дескриптор клієнта з Інтернетом: Цей дескриптор дозволяє клієнтам вказати серверу свій власний ідентифікатор, щоб показати, що вони мають доступ до Інтернету. Це може бути корисною інформацією для сервера під час обробки запитів і забезпечення відповідної функціональності.

Таким чином, ця характеристика забезпечує механізми для ефективного обміну повідомленнями та комунікації між клієнтами і сервером на основі вказаних дескрипторів.

Таблиця маршрутизації. Цей механізм використовується для забезпечення актуальності та консистентності таблиці маршрутизації. За допомогою дескриптора версії сервери можуть визначити, чи потрібно оновлення таблиці маршрутизації, що дозволяє забезпечити правильну маршрутизацію даних в мережі. Ця характеристика дозволяє ефективно передавати та синхронізувати таблицю маршрутизації на стороні сервера, забезпечуючи актуальність та надійність процесу маршрутизації в мережі.

ID наступного сервера. Ця характеристика вказує на глобально синхронізоване значення, яке відображає ідентифікатор наступного сервера, доступного в мережі BE-Mesh. Цей параметр забезпечує унікальність ідентифікаторів серверів у мережі і дозволяє глобально визначати наступний доступний ідентифікатор. Це гарантує, що кожен сервер отримує унікальний ідентифікатор із синхронізованого набору значень, який використовується для ідентифікації серверів у мережі BE-Mesh.

Клієнт онлайн. Ця характеристика використовується сервером для збереження списку активних клієнтів і має два дескриптори, які виконують певні

функції.

– Дескриптор сповіщень: Цей дескриптор використовується сервером для отримання нового списку клієнтів, які знаходяться в режимі онлайн. Він дозволяє серверу отримувати сповіщення про зміни у списку активних клієнтів, що дозволяє серверу підтримувати актуальну інформацію про клієнтів, які знаходяться у мережі.

– Дескриптор нового клієнта онлайн: Цей дескриптор використовується серверами для зв'язку з новим клієнтом, щойно він з'являється в мережі. Використовуючи цей дескриптор, сервери можуть взаємодіяти з новими клієнтами та встановлювати з ними зв'язок, щоб забезпечити повноцінне функціонування мережі [14].

Парадигма BE-Mesh дозволяє створювати мережеві з'єднання з використанням BLE між бездротовими пристроями в мережі типу меш з різними топологіями. Прототипована Android бібліотека та додаток практично реалізують запропоновану парадигму комунікації.

2.3. Протоколи маршрутизації для LoRa бездротової Mesh-мережі

Реалізація та аналіз протоколів маршрутизації для бездротових мереж типу ad hoc. Розглядаються протоколи DVR (Distance Vector Routing), AODV (Ad Hoc On-Demand Distance Vector) і DSR (Dynamic Source Routing) за допомогою симулятора мереж Cupcarbon [15].

Використано бездротову технологію зв'язку LoRa, а топологія мережі розподілена вздовж вулиць міста Жоао-Пессоа, Бразилія [15]. Основними критеріями кількісного аналізу є відсоток доставки пакетів, середнє затримку від початку до кінця та пропускна здатність. Також отримано профілі споживання енергії кожним вузлом мережі для кожного протоколу. Результати показали, що протокол DSR має менше споживання енергії, а протокол AODV має кращу загальну продуктивність, але використовує більше енергії. Протокол DVR показав кращі результати щодо затримки.

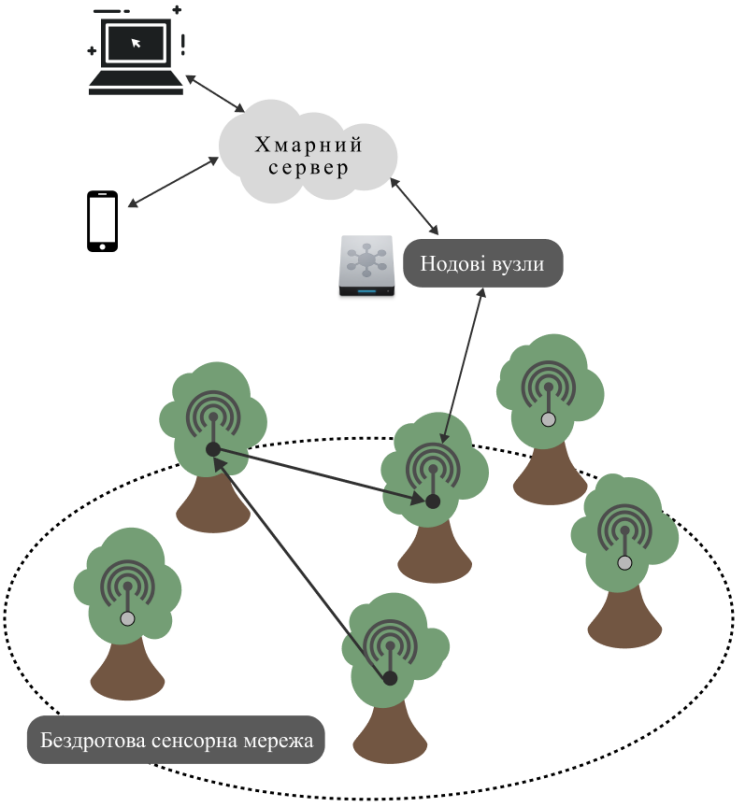

Рисунок 2.2 – Приклад бездротової сенсорної мережі. Адаптовано з роботи [15]

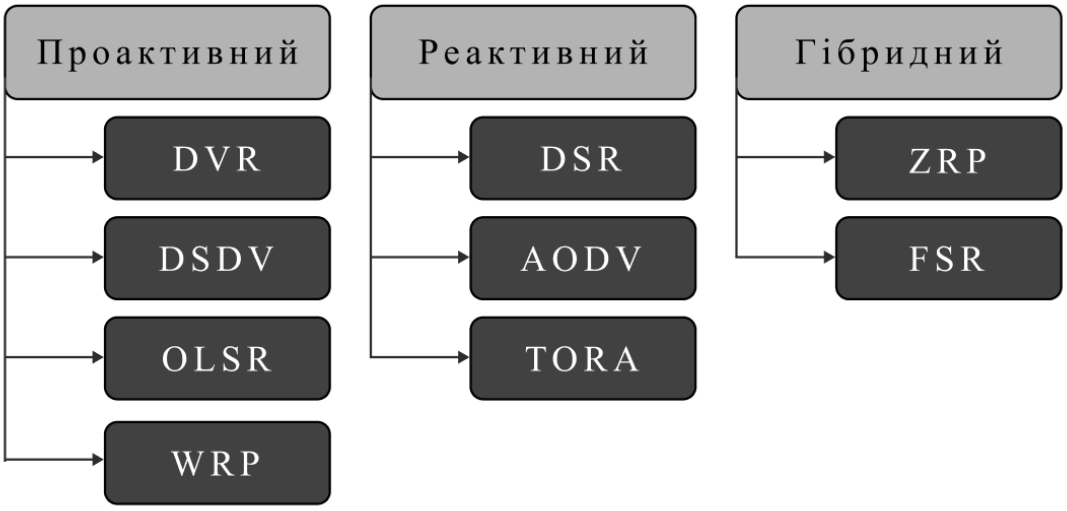

Рисунок 2.3 – Класифікація протоколів маршрутизації для ad hoc бездротових мереж. Адаптовано з роботи [15]

Протоколи маршрутизації мають різні характеристики щодо споживання енергії, продуктивності та затримки в мережах бездротових датчиків. Вибір конкретного протоколу залежить від конкретних вимог та обмежень мережі.

2.4. Метрика міжрівневої маршрутизації в бездротових Mesh-мережах

Для прогнозування якості з'єднання у бездротових мережах з маршрутизацією на основі мережі Mesh (WMNs) було розглянуто методику побудови метрики mesh-мережі [16]. Основна проблема в WMNs полягає у точному визначенні якості з'єднання, оскільки це впливає на ефективність протоколів мережі. Для вирішення цієї проблеми застосовуються техніки прогнозування машинного навчання, зокрема множинної лінійної регресії, векторної регресії та гаусової регресії [16].

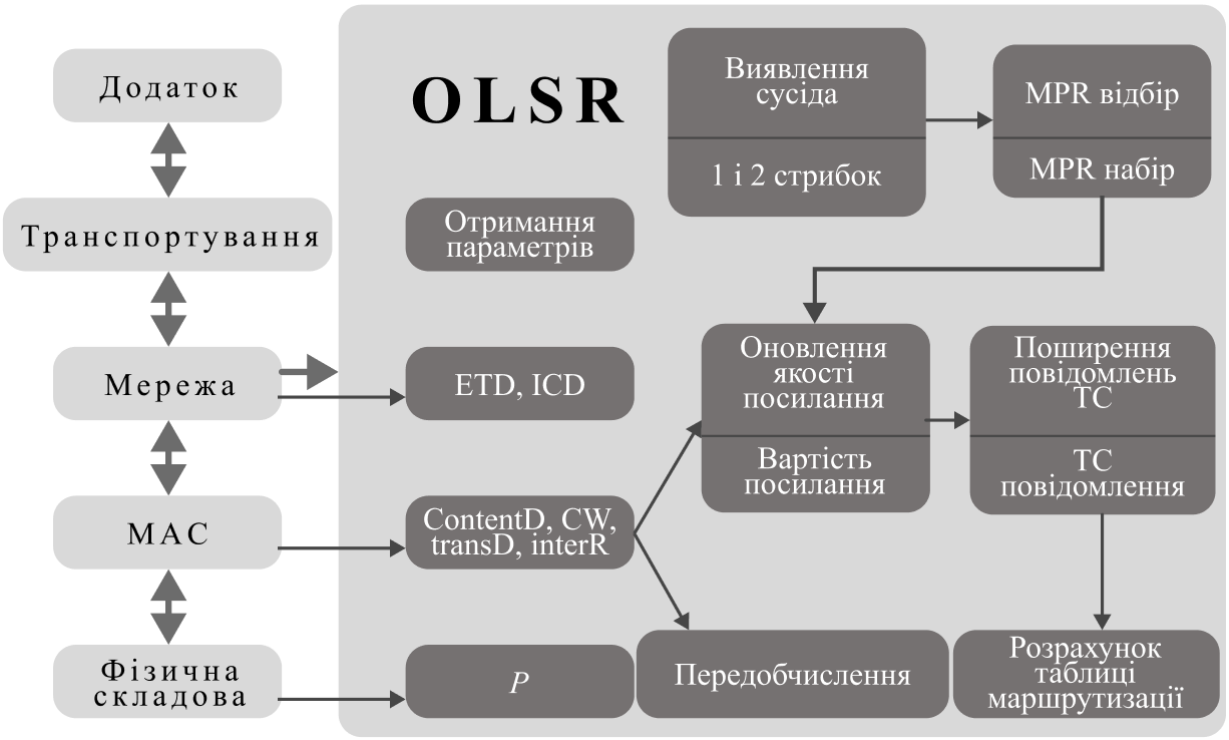

Рисунок 2.4 – Міжрівнева архітектура з використанням прогнозування вартості зв’язку. Адаптовано з роботи [16]

Експериментально досліджені різні сценарії з метою прогнозування якості з'єднання у бездротових мережах з використанням метрики маршрутизації PCL-IDA, яка вбудована у протокол маршрутизації OLSR. Результати експериментів з використанням симулятора NS-2 показують, що параметри якості з'єднання, такі як пропускна здатність, середнє затримання та втрата пакетів, є кращими, коли застосовується метод прогнозування множинної лінійної регресії [16].

Отже, застосування методів машинного навчання, зокрема множинної лінійної регресії, для прогнозування параметрів мережі в бездротових мережах з маршрутизацією на основі мережі Mesh може покращити якість з'єднання та ефективність мережевих протоколів.

РОЗДІЛ 3
КОМПОНУВАННЯ ТА ПРОГРАМУВАННЯ ВУЗЛА MESH МЕРЕЖІ

3.1. Мікроконтролерна платформа ESP8266

У даному проекті використовується контролер ESP8266, що є високоінтегрованим чіпом Wi-Fi модуля, який розроблений для забезпечення бездротового зв'язку та підключення до Інтернету. Цей чіп має вбудований мікроконтролер і відкриту архітектуру, що робить його дуже популярним серед розробників Інтернету речей (IoT) і різних електронних проектів.

Мікроконтролер ESP8266 має вбудований мікроконтролер з архітектурою RISC, який працює на швидкості до 80 МГц. Це дозволяє виконувати програми та керувати різними функціями пристрою.

Чіп ESP8266 підтримує **Бездротове підключення** до мережі Wi-Fi, що дозволяє пристрою підключатися до локальної мережі або безпосередньо до Інтернету. Він підтримує режими роботи станції (Station Mode) і точки доступу (Access Point Mode), що дозволяє використовувати його як клієнта або створювати власну Wi-Fi мережу.

ESP8266 має піни загального призначення **GPIO**, які можуть використовуватися для підключення до різних периферійних пристроїв, таких як сенсори, реле, дисплеї і т.д. Це дозволяє розширювати можливості пристрою і взаємодіяти з зовнішнім середовищем.

ESP8266 підтримує різні **мережеві протоколи**, такі як TCP/IP, UDP, HTTP, MQTT та інші. Це дає можливість обмінюватися даними з різними серверами та іншими пристроями через Інтернет.

ESP8266 може бути **запрограмований** з використанням Arduino IDE або мови програмування MicroPython. Це дозволяє розробникам легко створювати програми для управління пристроєм, обробки даних та взаємодії з Інтернетом.

Функціонал модуля ESP8266 може бути розширений за допомогою різних модулів, таких як сенсори, зовнішні пам'яті, індикатори, модулі комунікації і багато

інших. Це дозволяє створювати різноманітні проекти з використанням додаткових компонентів.

ESP8266 є потужним і універсальним чіпом для створення бездротових пристроїв та розширення можливостей електронних проектів. Використання цього чіпа дозволяє швидко підключати пристрої до Інтернету та забезпечувати комунікацію в межах меш-мережі, тому він і був обраний, як основа для даного проекту.

3.2. Розробка програмного забезпечення для побудови Mesh-мережі

Спочатку розберемося, що таке MAC-адреса і як протокол зв'язку ESPNOW звертається до палат по MAC-адресі. MAC-адреса (Media Access Control Address) нагляд за доступом до середовища або Hardware Address, (фізична адреса) – це унікальний 6-байтний ідентифікатор, прошитий при виготовленні пристрою. Унікальність MAC-адрес досягається тим, що кожен виробник отримує 16 мільйонів адрес і в міру вичерпання може запросити новий діапазон. За адресою ми можемо дізнатися виробника плати, що буде використовуватися. Ми використали плату NodeMCU.

```
// Якщо плата ESP32
#ifdef ESP32
  #include <WiFi.h>
// Якщо плата ESP8266
#else
  #include <ESP8266WiFi.h>
#endif
void setup(){
  Serial.begin(115200);
  Serial.println(WiFi.macAddress());    // Визначаємо MAC-адресу плати
}
void loop(){
  Serial.print("MAC-адреса плати:  ");
  Serial.println(WiFi.macAddress());
  delay(5000);
}
```

Розглянемо скетч передавача. Плата приймач може визначити кожного відправника за його MAC-адресою. Але це незручно. Тому ми кожну плату передавачів визначатимемо за ID, який самі і встановимо.

Скетч для плати ESP8266, що в коді позначається як ID 1, а в схемі позначається як transmitter 1:

```
/*  MAC адреса плати:  48:3F:DA:54:F5:31 Головна(Приймач) */
#include <ESP8266WiFi.h>
#include <espnow.h>
#define ILLUMINATION A0   //  Куди підключений Фоторезистор A0
#define HOLL 5         //  Куди підключений датчик ХОЛЛА

int i,h;
uint8_t broadcastAddress[] = {0x48, 0x3F, 0xDA, 0x54, 0xF5,
0x31};   // MAC-адреса отримувача
#define BOARD_ID 1              // Встановлюємо ID плати як 1
// структура для відправки даних
typedef struct struct_message {
    int id;
    int x;
    int y;
} struct_message;
struct_message myData; // Створено повідомлення з найменуванням
myData для зберігання відправлених змінних

unsigned long lastTime = 0; // Змінна для зберігання часу
unsigned long timerDelay = 10000;  // Затримка в 10 секунд між
відправкою пакетів даних
// Callback-функція при відправленні повідомлень
void OnDataSent(uint8_t *mac_addr, uint8_t sendStatus) {
  Serial.print("\r\nСтастус Відправки Останнього Пакету: ");
  if (sendStatus == 0){ Serial.println("Повідомлення
доставлено"); }
  else{ Serial.println("Повідомлення НЕ доставлено"); }
}
void setup() {
  Serial.begin(115200); // Запуск монітор порта на швидкості
115200 бод
  WiFi.mode(WIFI_STA);           // Режим роботи Клієнт
  WiFi.disconnect();
  // Ініціалізуємо ESP-NOW
  if (esp_now_init() != 0) {
    Serial.println("Помилка ініціалізації ESP-NOW");
    return;
  }
  // Роль плати в ESP-NOW
  esp_now_set_self_role(ESP_NOW_ROLE_CONTROLLER);
  esp_now_register_send_cb(OnDataSent);      // Отримуємо
повідомлення про відправку
  esp_now_add_peer(broadcastAddress, ESP_NOW_ROLE_SLAVE, 1,
NULL, 0);     // Реєструємо піри
}
void loop() {
  if ((millis() - lastTime) > timerDelay) {      // Якщо
пройшло більше 10 секунд
  i = analogRead(ILLUMINATION);   // Зчитуємо освітленість
```

```
      h = digitalRead(HOLL);          // Зчитуємо датчик Холла
        myData.id = BOARD_ID;
        myData.x = i;                 // Надсилаємо дані
        myData.y = h;                 // Надсилаємо дані
        esp_now_send(0, (uint8_t *) &myData, sizeof(myData));
// Надсилаємо повідомлення
        lastTime = millis();
      }
    //  Serial.print(F("Фоторезистор:\t"));
    //  Serial.print(i);
    //  if(i <= 700) Serial.print(" Темно");       else
Serial.print(" Светло");
    //  Serial.print(F("\tМагнит:\t"));
    //  if(h == 0) Serial.println("Спрацювало");   else
Serial.println("Піднесіть магніт");
    //  delay(2000);
    }
```

У цьому прикладі розглядається дві плати передавачів, тому ми їх і назвемо, як ID 1 і ID 2.

Скетч для плати ESP8266, що в коді позначається як ID 2, а в схемі позначається як transmitter 2:

```
    /* MAC адреса плати:  48:3F:DA:54:F5:31 Головна(Приймач) */
    #include <ESP8266WiFi.h>
    #include <espnow.h>
    #include <Adafruit_Sensor.h>
    #include "DHT.h"
    #define DHTPIN 5          //  Куди підключений датчик
      #define DHTTYPE DHT11       // DHT11 Версія датчика
    //#define DHTTYPE DHT22   // DHT 22  (AM2302), AM2321
    //#define DHTTYPE DHT21       // DHT 21 (AM2301)
      DHT dht(DHTPIN, DHTTYPE);
      float h,t;
    uint8_t broadcastAddress[] = {0x48, 0x3F, 0xDA, 0x54, 0xF5,
0x31};        // MAC-адреса отримувача
    #define BOARD_ID 2         // Встановлюємо ID плати як 2
    // структура для відправки даних
    typedef struct struct_message {
        int id;
        int x;
        int y;
    } struct_message;
    struct_message myData;     // Створено повідомлення з
найменуванням myData для зберігання відправлених змінних
    unsigned long lastTime = 0; // Змінна для зберігання часу
    unsigned long timerDelay = 10000; // Затримка в 10 секунд між
відправкою пакетів даних
    // Callback-функція при відправленні повідомлень
    void OnDataSent(uint8_t *mac_addr, uint8_t sendStatus) {
      Serial.print("\r\nСтастус Відправки Останнього Пакету: ");
      if (sendStatus == 0){ Serial.println("Повідомлення
```

```
доставлено"); }
       else{ Serial.println("Повідомлення НЕ доставлено"); }
     }
     void setup() {
       Serial.begin(115200);     // Запуск монітор порта на
швидкості 115200 бод
       dht.begin();
       WiFi.mode(WIFI_STA);      // Режим роботи Клієнт
       WiFi.disconnect();
       // Ініціалізуємо ESP-NOW
       if (esp_now_init() != 0) {
         Serial.println("Помилка ініціалізації ESP-NOW");
         return;
       }
       // Роль плати в ESP-NOW
       esp_now_set_self_role(ESP_NOW_ROLE_CONTROLLER);
       esp_now_register_send_cb(OnDataSent); // Отримуємо
повідомлення про відправку
       esp_now_add_peer(broadcastAddress,   ESP_NOW_ROLE_SLAVE,   1,
NULL, 0);     // Реєструємо піри
     }
     void loop() {
       if ((millis() - lastTime) > timerDelay) {     // Якщо пройшло
більше 10 секунд
       h = dht.readHumidity();       // Зчитуємо Вологість
       t = dht.readTemperature();    // Зчитуємо Температуру
         myData.id = BOARD_ID;
         myData.x = h;               // Надсилаємо дані
         myData.y = t;               // Надсилаємо дані
         esp_now_send(0, (uint8_t *) &myData, sizeof(myData));
// Надсилаємо повідомлення
         lastTime = millis();
       }
     //  Serial.print(F("Вологість:\t"));
     //  Serial.print(h);
     //  Serial.print(F("%\tТемпература:\t"));
     //  Serial.print(t);
     //  Serial.println(F("*C"));
     //  delay(2000);
     }
```

Для надсилання повідомлень ми будемо використовувати ID та змінні. Для другої це температура та вологість. Для першої це буде датчик освітленості та датчик холу. Датчиків на платі, як і самих плат, може бути велика кількість, що в свою чергу утворює бездротову mesh-мережу.

```
#include <ESP8266WiFi.h>
#include <espnow.h>
// Structure example to receive data
// Must match the sender structure
typedef struct struct_message {
    int id;
```

```
    int x;
    int y;
} struct_message;
struct_message board2;
void OnDataRecv(uint8_t * mac_addr, uint8_t *incomingData,
uint8_t len) {
  char macStr[18];
  memcpy(&myData, incomingData, sizeof(myData));
  boardsStruct[myData.id-1].x = myData.x;
  boardsStruct[myData.id-1].y = myData.y;
}
void setup() {
  // Initialize Serial Monitor
  Serial.begin(115200);

  // Set device as a Wi-Fi Station
  WiFi.mode(WIFI_STA);
  WiFi.disconnect();

  // Init ESP-NOW
  if (esp_now_init() != 0) {
    Serial.println("Error initializing ESP-NOW");
    return;
  }

  // Once ESPNow is successfully Init, we will register for
recv CB to
  // get recv packer info
  esp_now_set_self_role(ESP_NOW_ROLE_SLAVE);
  esp_now_register_recv_cb(OnDataRecv);
}
void loop(){
  // Передавач - 1
  int board1X = boardsStruct[0].x;
  int board1Y = boardsStruct[0].y;
  Serial.print("Фоторезистор: ");Serial.println(board1X);
  Serial.print("Магніт: ");Serial.print(board1Y);
  Serial.println();
  // Передавач - 2
  int board2X = boardsStruct[1].x;
  int board2Y = boardsStruct[1].y;
  Serial.print("Вологість: ");
Serial.print(board2X);
Serial.println("%");
  Serial.print("Температура: ");
Serial.print(board2Y);Serial.println("°C");
  Serial.println();
  delay(2000);
}
```

Після того, як ми все налаштували, потрібно увімкнути усі наші плати. Плата приймач повинна бути підключена до комп'ютера, інакше неможливо побачити

отримання даних.

Наразі виявлено один недолік цієї системи - це неможливість побачити отримані дані, створивши сервер і вивівши дані на веб-сторінку. Дані можна побачити лише на моніторі порту, але це не заважає керувати різними пристроями, підключеними до плати.

3.3. Побудова схеми для передачі даних в Mesh-мережі

Розробка мережі mesh на базі ESP8266 чіпу, для передачі даних про температуру, вологість, увімкнення реле і т.д. Для прототипування цього модуля використаємо середовище моделювання схем – EasyEDA.

Для створення повноцінної mesh-мережі нам потрібно пов’язати плати ESP. Попередньо потрібно передати данні з кількох плат ESP на іншу. ESP8266 буде отримувати дані, обробляти і керувати різними пристроями, виходячи з отриманих даних. Таким чином ми можемо вирішити проблему недостачі пінів на платі і зможемо розташувати датчики та сенсори в різних частинах приміщення або групи приміщень. Для роботи ми будемо використовувати протокол ESPNOW, і одну з його конфігурацій: «**багато до одного»**. Також є інші конфігурації, наприклад, «одна до багатьох», і кожна може працювати як приймач і передавач, що відправляє і отримує дані. Конфігурація, яку ми будемо використовувати – це один з найкращих способів бездротового зв'язку кількох плат та обробки даних одного з них.

Для під’єднання двох плат, ми будемо використовувати той самий протокол ESPNOW, що згадувався в підрозділі 3.2. Під час використання протоколу ESPNOW не існує таких понять, як приймач чи передавач. Кожна плата може працювати як приймач, і як передавач, а може поєднувати ці значення і бути, і приймачем, і передавачем одночасно.

В схемі на рисунку 3.1 два блоки transmitter 1 і transmitter 2 на макетній платі розміщено дві плати ESP8266. До першої підключені датчик освітленості BH1750 та датчик Холла A3144E. До другої плати підключено температурний датчик

DHT11. Також можна підключити будь-які датчики, дані з яких потрібно передати. Далі зображено звичайний регулятор напруги на 3,3 Вольти, тому що ESP8266 працюють від цієї напруги.

Як плати приймача ми використовуємо NodeMCU, що зображено на рисунку 3.2, плати ніяк не пов'язані одна з одною. Для перевірки отримання даних ми використовуємо монітор порту з підключеною платою NodeMCU. Наразі плата отримує дані з обох плат з інтервалом у 10 секунд. Але виведення значень було зроблено кожні дві секунди. У коді скетча було закоментовано виведення повідомлення про помилку передачі даних та успішність передачі. Також є інформація про кількість байтів, переданої інформації із зазначенням, з якої плати були отримані дані.

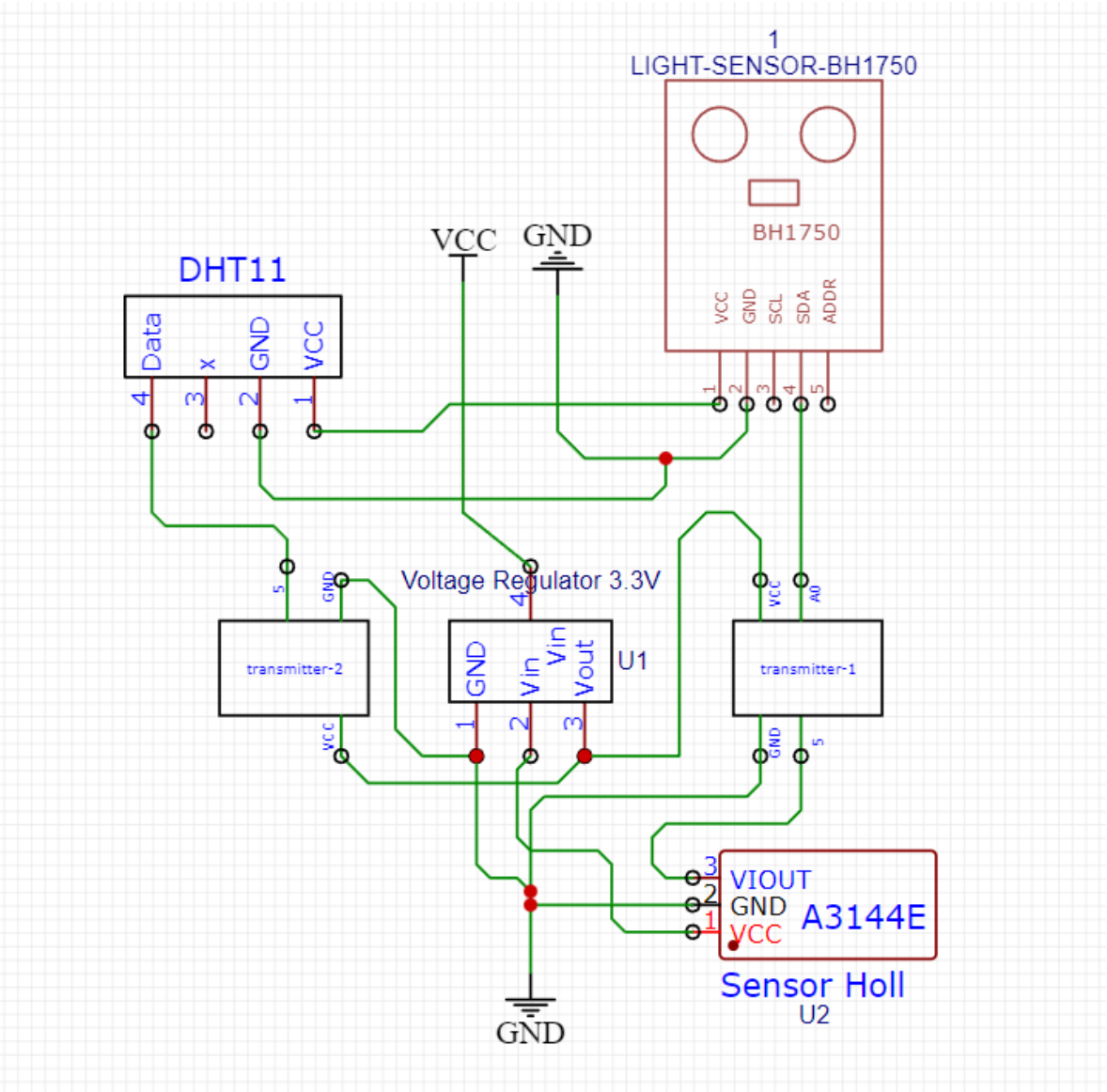

Рисунок 3.1 – Розроблена схема для побудови mesh-мережі

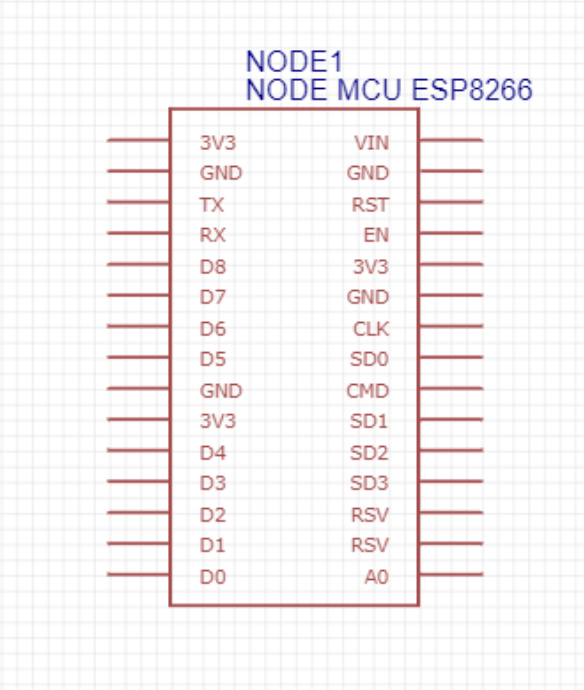

Рисунок 3.2 – Плата NodeMCU до якої під’єднаний монітор на яку передаються дані з ESP8266

Якщо піднести магніт до датчика холу, на платі передавачі, можемо побачити, що значення змінилися з одиниці на нуль. Це говорить про те, що дані передалися на приймач через 10 секунд після того, як прибрали магніт, а саме такий інтервал встановлено в скетчі.

```
20:25:03.341 -> Освітлюваність: 1022
20:25:03.375 -> Датчик хола: 0
20:25:03.409 -> Влажність: 29%
20:25:03.409 -> Температура: 20°C
20:25:03.443 ->
20:25:04.395 -> Освітлюваність: 1021
20:25:04.429 -> Датчик хола: 1
20:25:04.463 -> Влажність: 29%
20:25:04.463 -> Температура: 20°C
20:25:04.497 ->
```

Рисунок 3.3 – Знімок з монітору виводу даних з датчика, що передаються на плату NodeMCU та демонструє зміни даних при взаємодії з датчиком Холла

Дані повертаються у початкове значення. Якщо знову піднести магніт і перевірити роботу передачі, можемо побачити, що передача працює. Також

перевіряється, як передаватимуться дані з температурного датчика. Вологість змінюється дуже швидко, а ось температура набагато повільніша.

```
20:25:01.200 -> Освітлюваність: 1020
20:25:01.234 -> Датчик хола: 0
20:25:01.267 -> Влажність: 28%
20:25:01.301 -> Температура: 20°C
20:25:01.335 ->
20:25:02.253 -> Освітлюваність: 1020
20:25:02.321 -> Датчик хола: 0
20:25:02.321 -> Влажність: 28%
20:25:02.355 -> Температура: 21°C
20:25:02.389 ->
```

Рисунок 3.4 – Знімок з монітору виводу даних з датчика, що передаються на плату NodeMCU та демонструє зміни даних при взаємодії з датчиком температури

Ну і, нарешті, подивимося, як змінюються дані з датчика освітленості. Для кращого спрацьовування ми вмикаємо світло, тепер заслонивши датчик, ми будемо отримувати різні значення.

```
20:24:58.039 -> Освітлюваність: 1023
20:24:58.073 -> Датчик хола: 0
20:24:58.107 -> Влажність: 29%
20:24:58.107 -> Температура: 20°C
20:24:58.141 ->
20:24:59.092 -> Освітлюваність: 1021
20:24:59.126 -> Датчик хола: 1
20:24:59.160 -> Влажність: 28%
20:24:59.194 -> Температура: 21°C
20:24:59.228 ->
20:25:00.146 -> Освітлюваність: 1022
20:25:00.180 -> Датчик хола: 1
20:25:00.214 -> Влажність: 29%
20:25:00.248 -> Температура: 21°C
20:25:00.282 ->
```

Рисунок 3.5 – Знімок з монітору виводу даних з датчика, що передаються на плату NodeMCU та демонструє зміни даних при взаємодії з датчиком освітленості

Перевіримо, як передача працює, якщо змінити не одне, а два значення, а дані з температурного датчика змінюються самі.

Бачимо, що значення виводяться. Отже, передача працює.

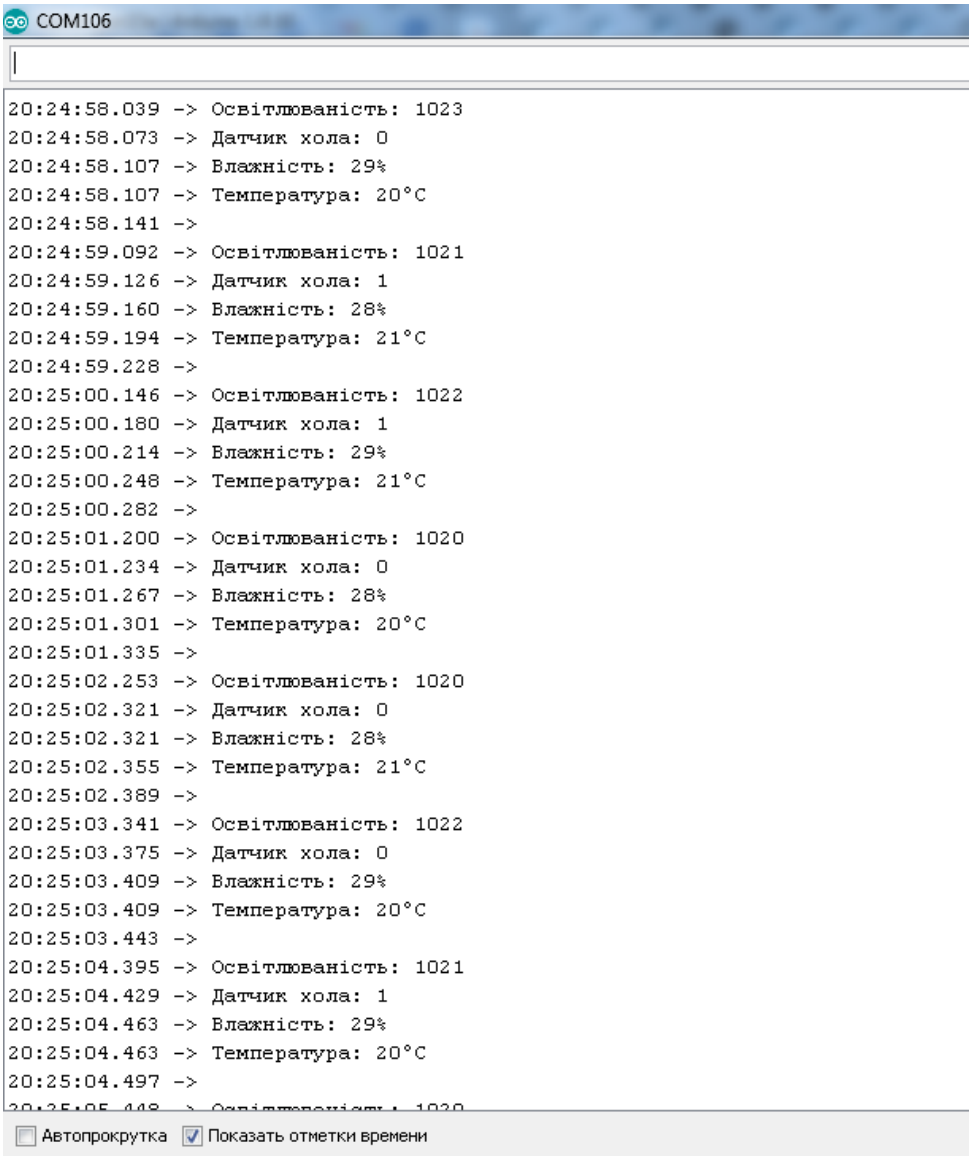

Рисунок 3.6 – Знімок з монітору виводу даних з датчика, що передаються на плату NodeMCU та демонструє зміну даних при взаємодії з різними датчиками одночасно

ВИСНОВКИ

1. Аналіз сучасних літературних джерел за тематикою Mesh-мереж показав, що вони мають широкий спектр застосування у різних галузях, включаючи бездротові комунікації, датчики моніторингу, розумний дім і т.д. Розглянуто базові визначення та принципи роботи розповсюджених мереж, таких як: Ethernet, AppleTalk, Token Ring, ArcNet. Проаналізовано основні протоколи та архітектуру Mesh-мереж.

2. Розглянуто протоколи маршрутизації, такі як DVR, AODV і DSR, у контексті LoRa бездротових Mesh-мереж. Зроблено оцінку їх продуктивність та ефективність в умовах реальних розподілених систем. Досліджено методики побудови метрики Mesh-мережі з використанням Mesh (WMNs), що дозволило зрозуміти користь використання машинного навчання для прогнозування параметрів мережі. Проаналізовано парадигму BE-Mesh, що дозволяє створювати мережеві з'єднання з використанням BLE між бездротовими пристроями в мережі типу Mesh з різними топологіями.

3. Розроблена схема та програмне забезпечення розподіленої mesh-мережі моніторингу оточуючого середовища на базі мікроконтролерної платформи ESP-8266. Під час виконання роботи було використано онлайн середовище проєктування EasyEDA з інтегрованими модулями платформи ESP-8266. Також був написаний код, що розроблявся мовою C++ в середовищі програмування Arduino IDE.

4. У результаті проведеного проєктування та розробки програмного забезпечення отримано Mesh-мережу моніторингу фізичних величин на основі мікроконтролерних модулів ESP з апаратною підтримкою Wi-Fi.

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. Шеховцов В. А. Операційні системи. Київ: BHV Publishing Group, 2005. 558 с.

2. Ethernet [Електронний ресурс]. – Режим доступу: https://www.cs.unc.edu/~jbs/resources/Internet/ethernet/ Дата доступу: 16.05.2023

3. Token Ring in Computer Networks [Електронний ресурс]. – Режим доступу: https://www.scaler.com/topics/token-ring-in-computer-networks/ Дата доступу: 16.05.2023

4. AppleTalk [Електронний ресурс]. – Режим доступу: https://wiki.cuspu.edu.ua/index.php/AppleTalk Дата доступу: 16.05.2023

5. W. Yao, L. Hou, K. Paul, Y. Ban, T. Zhao. ArcNet: Series AC Arc Fault Detection Based on Raw Current and Convolutional Neural Network. // IEEE Transactions on Industrial Informatics 18, no. 1 (2022): 77–86.

6. Yuan Chai, Xiao-Jun Zeng. The development of green wireless mesh network: A survey // J Smart Environ Green Comput 2021, 1:47-59,

7. Smart Bluetooth Wireless Mesh Networks [Електронний ресурс]. – Режим доступу: https://www.warehouse-lighting.com/blogs/lighting-blog/smart-bluetooth-wireless-mesh-networks Дата доступу: 16.05.2023

8. Chai-Keong Toh. Ad Hoc Mobile Wireless Networks: Protocols and Systems. New Jersey: Prentice Hall PTR, 2002. 302 с.

9. A. Neumann, C. Aichele, M. Lindner, S. Wunderlich. Better Approach To Mobile Ad-hoc Networking (B.A.T.M.A.N.), 2008. 24 с.

10. The Working Group for WLAN Standards of the Institute of Electrical and Electronics Engineers [Електронний ресурс]. – Режим доступу: https://mentor.ieee.org/mentor/bp/StartPage Дата доступу: 16.05.2023

11. 802.11s Wireless Mesh Networking [Електронний ресурс]. – Режим доступу: https://wiki.freebsd.org/WiFi/Mesh Дата доступу: 16.05.2023

12. S. Pierre, M.Barbeau, E Kranakis. Ad-Hoc, Mobile, and Wireless Network // ADHOC-NOW. Montreal: Second International Conference, 2003

13. C. Ebi, F. Schaltegger, A. Rüst, F. Blumensaat. Synchronous LoRa Mesh Network to Monitor Processes in Underground Infrastructure // Special Section On Advanced Sensor Technologies On Water Monitoring And Modeling. – V7, 2019

14. A. Lacava, G. Nero, P. Locatelli, F. Cuomo, T. Melodia. BE-Mesh: Bluetooth Low Energy Mesh Networking // IEEE INFOCOM Demo, 2019

15. D. Medeiros, M. Villarim, F. Carvalho, C. Souza. Implementation and Analysis of Routing Protocols for LoRa Wireless Mesh Networks // IEMCON 2020: Vancouver, British Columbia, Canada. – 2020

16. M. Naravani, D. Narayan, S. Shinde, M. Mulla. A Cross-Layer Routing Metric with Link Prediction in Wireless Mesh Networks // Procedia Computer Science. – V.171, 2020. –p. 2215–2224

17. G. Saldamli, S. Deshpande, K. Jawalekar, P.Gholap. Wildfire Detection using Wireless Mesh Network. // 2019 Fourth International Conference on Fog and Mobile Edge Computing (FMEC). – 2019

18. Пелих Р.К., Тищенко К.В. Принципи організації MESH-мереж: топології та протоколи / Матеріали Міжнародної науково-технічної конференції студентів та молодих вчених «Фізика, електроніка, електротехніка ФЕЕ-2023». – Суми: СумДУ, 2023. – С.56.