**КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ІМЕНІ ТАРАСА ШЕВЧЕНКА**

ФАКУЛЬТЕТ РАДІОФІЗИКИ, ЕЛЕКТРОНІКИ ТА КОМП'ЮТЕРНИХ СИСТЕМ

Кафедра радіотехніки та радіоелектронних систем

«На правах рукопису»

Робота допущена до захисту в ЕК
рішенням кафедри радіотехніки та радіоелектронних систем
від 20 травня 2024 року, протокол № 111.
Завідувач кафедри доктор фіз.-мат. наук, професор
Ігор АНІСІМОВ

ДИПЛОМНА РОБОТА МАГІСТРА
на тему:
«УДОСКОНАЛЕНИЙ КАНАЛ ЗВ'ЯЗКУ З ВИКОРИСТАННЯМ ТЕХНОЛОГІЙ LORA»

**Виконав:**
студент 2-го курсу магістратури
денної форми навчання
спеціальності 172 - Електронні комунікації та радіотехніка
ОНП «Інформаційна безпека телекомунікаційних систем і мереж»
Малєєв Сергій Олександрович

**Науковий керівник**:
доцент кафедри радіотехніки
та радіоелектронних систем, к.т.н., с.н.с.
Жиров Генадій Борисович

**Рецензент:**
доктор технічних наук, професор
Вишнівський Віктор Вікторович

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

Київ – 2025

# РЕФЕРАТ

Дипломна робота: 51 с., 9 табл., 20 рис., 2 дод. (21 с.), 15 джерел.

ТЕХНОЛОГІЯ LORA, ADAPTIVE DATA RATE (ADR), FREQUENCY HOPPING (FH), КОДИ ХЕММІНГА, ЗАВАДОСТІЙКІСТЬ, ІНТЕРНЕТ РЕЧЕЙ.

Об’єкт дослідження – канал бездротового зв'язку на основі технології LoRa.

Мета роботи – покращення якості каналу зв’язку за рахунок використанням алгоритмів адаптивної швидкості передачі даних, частотного перемикання та кодування Хеммінга.

Впроваджено адаптивний алгоритм Adaptive Data Rate (ADR), що дозволяє автоматично коригувати параметри передачі сигналу залежно від стану радіоканалу. Модифіковано, адаптовано і впроваджено Frequency Hopping (FH) для динамічного уникнення зашумлених каналів, що забезпечує підвищення стійкості та зменшення втрат пакетів. Для додаткового захисту інформації впроваджено кодування Хеммінга (7,4), що дозволяє ефективно виправляти одиничні помилки, типові для умов помірного шуму.

Проведені теоретичні розрахунки підтверджують ефективність розроблених методів: ADR забезпечує оптимальний баланс між швидкістю передачі та дальністю, FH знижує негативний вплив завад шляхом вибору оптимальної частоти передачі, а використання кодів Хеммінга значно підвищує надійність зв'язку при низьких і середніх рівнях помилок.

Практичні експерименти на основі модуля LoRa SX1278 та мікроконтролера Arduino Uno підтвердили високу ефективність системи, стабільність каналу зв'язку та суттєве покращення завадостійкості в порівнянні зі стандартними рішеннями.

У подальшому планується удосконалення алгоритмів для динамічнішої адаптації до умов середовища та дослідження перспектив використання більш складних кодів корекції помилок, таких як коди Ріда-Соломона або регуляції каналу зв’язку за допомогою машинного навчання.

**ЗМІСТ**

# ПЕРЕЛІК СКОРОЧЕНЬ ТА УМОВНИХ ПОЗНАЧЕНЬ

ADR – Adaptive Data Rate (адаптивна швидкість передачі даних);

AES – Advanced Encryption Standard (стандарт розширеного шифрування);

BER – Bit Error Rate (ймовірність бітової помилки);

BSC – Binary Symmetric Channel (бінарний симетричний канал);

BW – Bandwidth (ширина смуги пропускання);

CR – Coding Rate (коефіцієнт кодування);

CSS – Chirp Spread Spectrum (чірпова модуляція з розширенням спектру);

dBi – Decibels relative to isotropic radiator (децибели відносно ізотропного випромінювача);

ECC – Error-Correcting Codes (коди корекції помилок);

FEC – Forward Error Correction (пряма корекція помилок);

FH – Frequency Hopping (частотне стрибання);

IoT – Internet of Things (інтернет речей);

ISM – Industrial, Scientific, and Medical (промисловий, науковий та медичний діапазон частот);

LoRa – Long Range (технологія бездротового зв'язку великої дальності);

LoRaWAN – Long Range Wide Area Network (мережа великої дальності на базі технології LoRa);

LPWAN – Low Power Wide Area Network (низькоенергетична мережа великої дальності);

MAC – Medium Access Control (керування доступом до середовища передачі даних);

NB-IoT – Narrowband Internet of Things (вузькосмуговий інтернет речей);

RSSI – Received Signal Strength Indicator (індикатор рівня прийнятого сигналу);

RX – Receiver (приймач);

SF – Spreading Factor (коефіцієнт розширення спектра);

SFH – Slow Frequency Hopping (повільне частотне стрибання);

SNR – Signal-to-Noise Ratio (відношення сигнал/шум);

SPI – Serial Peripheral Interface (послідовний периферійний інтерфейс);

SX1278 – модель чипа для радіозв’язку, що використовується у технології LoRa;

TX – Transmitter (передавач);

TxPower – Transmit Power (потужність передавача);

Wi-Fi – Wireless Fidelity (бездротова технологія зв'язку на базі стандарту IEEE 802.11);

XOR – Exclusive OR (операція виключного або);

ZigBee – стандарт мережевих протоколів бездротового зв'язку, орієнтований на застосування з низьким енергоспоживанням.

Умовні позначення:

σ – електропровідність;

λ – довжина хвилі;

ε – діелектрична проникність;

E – напруженість електричного поля.

# ВСТУП

Сучасний етап розвитку інформаційно-комунікаційних технологій характеризується стрімким зростанням обсягів даних, що передаються через бездротові канали зв’язку, та посиленням вимог до їхньої надійності, енергоефективності й завадостійкості. У контексті глобальних трендів, таких як розгортання Інтернету речей (IoT), створення розумних міст і автоматизація промислових процесів, бездротові мережі стають ключовим елементом забезпечення ефективної взаємодії між пристроями в умовах складних електромагнітних середовищ. Особливо актуальними є низькоенергетичні мережі з великою дальністю дії (LPWAN), які забезпечують стабільний зв’язок на значних відстанях при мінімальному енергоспоживанні. Технологія LoRa вирізняється унікальним поєднанням високої завадостійкості, енергоефективності та економічної доцільності, що робить її перспективною для широкого спектра застосувань – від моніторингу довкілля до інтелектуальних систем управління.

Незважаючи на значний прогрес у розвитку бездротових технологій, сучасні канали зв’язку стикаються з викликами, пов’язаними з багатошляховим поширенням сигналів, електромагнітними завадами та обмеженнями радіочастотного спектру. Ці фактори призводять до зниження якості зв’язку, зростання бітової помилки (Bit Error Rate, BER) і втрати пакетів даних, що є критичним для застосувань, де надійність передачі є пріоритетною. У зв’язку з цим виникає потреба в удосконаленні методів підвищення завадостійкості каналів зв’язку шляхом модифікації та адаптації алгоритмів, які забезпечують оптимальне використання ресурсів і стійкість до зовнішніх перешкод, особливо на апаратних платформах з обмеженими обчислювальними можливостями.

Метою даної роботи є підвищення завадостійкості каналу зв’язку на базі технології LoRa шляхом модифікації, адаптації та імплементації алгоритмів адаптивної швидкості передачі даних (Adaptive Data Rate, ADR), частотного

перемикання (Frequency Hopping, FH) і кодування Хеммінга (7,4) на апаратно-програмному комплексі з обмеженими ресурсами, зокрема мікроконтролері Arduino Uno. Робота спрямована на створення ефективного рішення, яке забезпечує стабільну передачу даних у зашумлених умовах, та на експериментальну верифікацію запропонованих методів у різних сценаріях – від лабораторних тестів до умов міських середовищ.

Практична цінність роботи полягає у створенні апаратно-програмного комплексу на базі модулів LoRa SX1278 та мікроконтролера Arduino Uno, який забезпечує високу завадостійкість завдяки модифікованим і адаптованим алгоритмам, ефективно реалізованим на платформі з обмеженими обчислювальними ресурсами. Запропоновані рішення дозволяють оптимізувати ключові параметри зв’язку, зокрема коефіцієнт розширення спектра (Spreading Factor, SF) і вихідну потужність (TX Power), забезпечують виправлення однобітних помилок за допомогою кодування Хеммінга (7,4) та динамічний вибір менш зашумленого каналу зв’язку з доступного частотного діапазону. Це знижує відсоток втрачених пакетів і підвищує надійність передачі даних.

# РОЗДІЛ 1. ТЕОРЕТИЧНІ ОСНОВИ ТА АНАЛІЗ СУЧАСНИХ ТЕХНОЛОГІЙ ДЛЯ СТВОРЕННЯ КАНАЛІВ ЗВ'ЯЗКУ

## 1.1. Огляд технологій бездротового зв'язку

Сучасний етап розвитку інформаційно-комунікаційних технологій характеризується експоненціальним зростанням обсягів даних, що передаються через бездротові канали, та посиленням вимог до їхньої надійності, енергоефективності й стійкості до завад. У контексті глобальних тенденцій, таких як Інтернет речей (IoT), розумні міста та автоматизація промислових процесів, бездротові технології стають фундаментом для створення масштабних розподілених мереж із обмеженими ресурсами. Цей підрозділ присвячено систематизації сучасних технологій бездротового зв'язку та аналізу низькоенергетичних мереж із великою дальністю дії (LPWAN) як ключового інструменту для IoT.

### 1.1.1. Класифікація технологій бездротового зв'язку

Бездротові технології класифікуються за параметрами дальності дії, пропускної здатності, енергоспоживання та сценаріїв застосування. Основні категорії включають наступні:

- Технології короткої дальності, такі як Wi-Fi (IEEE 802.11), Bluetooth і ZigBee, забезпечують високу пропускну здатність, до 1 Гбіт/с для Wi-Fi 6, при обмеженій дальності, що зазвичай не перевищує 100 м. Wi-Fi оптимальний для локальних мереж із високими вимогами до швидкості, тоді як Bluetooth і ZigBee застосовуються в енергоефективних IoT-системах, таких як розумні будинки. Їхнє обмеження полягає у високому енергоспоживанні та чутливості до завад у перенасичених частотних діапазонах, зокрема 2.4 ГГц [1].
- Технології середньої дальності, наприклад WiMAX і LTE, забезпечують зв'язок на відстанях до 5–10 км, що робить їх придатними для міських мереж і систем розумних міст. Однак їхнє розгортання потребує значних ресурсів, що обмежує застосування в енергоефективних сценаріях.

- Технології великої дальності, зокрема LPWAN (LoRa, Sigfox, NB-IoT), дозволяють передавати дані на відстані до 15 км у сільській місцевості та 1–3 км у міських умовах. Вони оптимізовані для передачі малих обсягів даних із мінімальним енергоспоживанням, що робить їх ідеальними для IoT-додатків, таких як моніторинг і телеметрія.
- Стільникові (4G, 5G) та супутникові технології забезпечують глобальне покриття, але їх висока вартість і енергоспоживання роблять їх менш придатними для масових IoT-розгортань.

LPWAN-технології вирізняються здатністю забезпечувати зв'язок у складних електромагнітних середовищах із мінімальними витратами ресурсів, що робить їх основою для створення масштабних IoT-мереж.

1.1.2. Порівняльний аналіз технологій LPWAN

LPWAN-технології розроблені для забезпечення енергоефективного зв'язку в умовах обмеженого радіочастотного спектра та високого рівня завад. Основними представниками є LoRa, Sigfox і NB-IoT. Їхні характеристики порівнюються в таблиці 1.

Таблиця 1. Порівняльний аналіз LPWAN-технологій [2]

| Параметр | LoRa | Sigfox | NB-IoT |
|---|---|---|---|
| **Дальність дії** | 10–15 км (сільська), 1–3 км (міська) | 10–50 км (сільська), 3–10 км (міська) | 10–15 км (сільська), 1–5 км (міська) |
| **Швидкість передачі** | 0.3–50 кбіт/с | 100 біт/с | 20–250 кбіт/с |
| **Енергоспоживання** | Низьке | Дуже низьке | Середнє |
| **Модуляція** | Chirp Spread Spectrum (CSS) | BPSK | QPSK |
| **Частотний спектр** | Неліцензований (ISM, 868/915 МГц) | Неліцензований (ISM) | Ліцензований (LTE) |
| **Вартість інфраструктури** | Низька | Низька | Висока |
| **Основні застосування** | IoT, розумні міста, моніторинг | Логістика, телеметрія | Розумні лічильники, транспорт |

LoRa вирізняється адаптивністю завдяки динамічному налаштуванню параметрів, таких як коефіцієнт розширення спектра (SF) і смуга пропускання (BW), що забезпечує оптимальний баланс між дальністю, швидкістю і стійкістю до завад. Використання неліцензованого спектра знижує витрати на

розгортання, роблячи LoRa економічно доцільною для широкого спектра застосувань. Sigfox має найнижче енергоспоживання, але обмеження у пропускній здатності (100 біт/с) і кількості повідомлень на день роблять її менш універсальною. NB-IoT, інтегрована з інфраструктурою LTE, забезпечує високу надійність, але потребує ліцензованого спектра і має вищі експлуатаційні витрати. Для IoT-додатків, де потрібна гнучкість і низька вартість, LoRa є оптимальним вибором завдяки поєднанню завадостійкості та енергоефективності.

### 1.2. Технологія LoRa: принципи роботи та особливості

Технологія LoRa (Long Range), розроблена компанією Semtech, є провідною LPWAN-технологією, яка базується на чірповій модуляції (Chirp Spread Spectrum, CSS) і підтримує протокол LoRaWAN. Вона забезпечує зв'язок на великих відстанях із мінімальним енергоспоживанням, що робить її ідеальною для IoT-додатків у складних електромагнітних середовищах. У цьому підрозділі розглядаються архітектура мережі LoRaWAN і принципи модуляції CSS, які визначають її унікальні характеристики.

#### 1.2.1. Архітектура мережі LoRaWAN

LoRaWAN – це протокол рівня керування доступом до середовища (MAC), який використовує LoRa як фізичний рівень передачі даних. Архітектура LoRaWAN включає чотири ключові компоненти:

- Кінцеві пристрої (End Devices): енергоефективні сенсори, які передають дані через модулі LoRa. Вони підтримують три класи зв'язку: Клас A – найенергоефективніший, із передачею даних за ініціативи пристрою і двома короткими вікнами прийому; Клас B – додає планові вікна прийому для періодичного зв'язку; Клас C – забезпечує постійний прийом для мінімальної затримки, але з вищим енергоспоживанням [3].
- Шлюзи (Gateways): забезпечують двосторонній зв'язок між кінцевими пристроями і мережевим сервером, приймаючи дані на кількох каналах і з різними SF одночасно. Вони передають пакети через Інтернет (Ethernet, 3G/4G).

- Мережевий сервер (Network Server): обробляє пакети, усуває дублювання, керує параметрами зв’язку (SF, TX Power, канали) і передає дані до прикладного сервера.
- Прикладний сервер (Application Server): відповідає за обробку даних для кінцевих додатків, таких як системи моніторингу чи управління.

1.2.2. Модуляція Chirp Spread Spectrum (CSS) у LoRa

Чірпова модуляція (Chirp Spread Spectrum, CSS) є основою фізичного рівня LoRa, забезпечуючи високу завадостійкість і велику дальність дії. CSS використовує чірпові сигнали – імпульси з лінійною зміною частоти в часі, що розподіляють енергію по широкому спектру. Основні принципи CSS включають:

- Чірповий сигнал: у LoRa чірп є висхідним (up-chirp) або низхідним (down-chirp), із частотою, що змінюється в межах смуги пропускання (BW = 125, 250 або 500 кГц). Дані кодуються шляхом модуляції початкової частоти або фази чірпа.

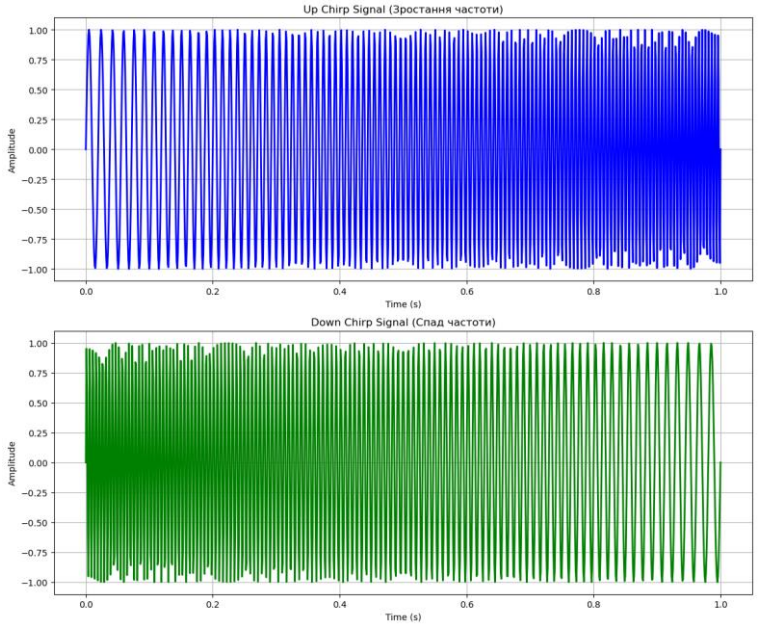


Рисунок 1.1 – Візуалізація чірп-сигналів у системах LoRa

Верхній графік рисунку 1.1 показує up-чірп (частота зростає), нижній — down-чірп (частота спадає). Вісь X — час (0–1 с), вісь Y — амплітуда (-1 до 1).)

- Коефіцієнт розширення спектра (SF): SF визначає кількість чіпів на біт даних і варіюється від 7 до 12. Кількість чіпів обчислюється як ($2^{SF}$), що впливає на тривалість символу і стійкість до завад. Наприклад, при SF=7 швидкість передачі становить до 5.5 кбіт/с, а при SF=12 – 0.3 кбіт/с, але з вищою дальністю [4]. Час передачі символу описується як:

$$T_s = \frac{2^{SF}}{BW}$$

де ( Ts ) – тривалість символу (с), ( BW ) – смуга пропускання (Гц) [4].

- Завадостійкість: CSS забезпечує стійкість до багатошляхового поширення сигналів і завад завдяки широкосмуговому розподілу енергії. LoRa може працювати при відношенні сигнал/шум (SNR) до -20 дБ, що значно перевищує можливості традиційних модуляцій, таких як FSK або QPSK [4].

Математично чірповий сигнал описується як:

$$s(t) = A \cdot \cos\left(2\pi\left(f_0 t + \frac{\mu t^2}{2}\right)\right)$$

де ( A ) – амплітуда, ( $f_0$ ) – початкова частота, ($\mu = \frac{BW}{T_s}$) – швидкість зміни частоти, ( t ) – час. Дані кодуються змінами ( $f_0$ ), що забезпечує стійкість до спотворень[5].

CSS дозволяє LoRa ефективно працювати в умовах високого рівня завад і багатошляхового поширення, що є типовим для міських і промислових середовищ.

### 1.3. Методи підвищення завадостійкості в каналах зв'язку LoRa

1.3.1. Алгоритм Adaptive Data Rate (ADR)

У стандартній реалізації, визначеній специфікацією LoRaWAN, алгоритм Adaptive Data Rate (ADR) передбачає, що мережевий сервер збирає статистику SNR і RSSI для останніх 20–30 пакетів від кінцевого пристрою, після чого оцінює запас сигналу (Link Margin), який обчислюється як[6]:

$$\text{Link Margin} = \text{SNRmeasured} - \text{SNRrequired}$$

де ($SNR_{required}$) залежить від Spreading Factor (SF) (наприклад, -20 дБ для ( SF = 12 ), -7.5 дБ для ( SF = 7 )) [3]. На основі цього запасу сервер динамічно налаштовує параметри передачі, такі як SF і вихідна потужність (TX Power), для забезпечення оптимального балансу між швидкістю передачі, дальністю і енергоспоживанням.

Для двовузлової системи, що складається з передавача і приймача, було розроблено спрощений варіант алгоритму ADR, адаптований до прямого зв'язку між двома вузлами без залучення мережевого сервера. У цій реалізації передавач самостійно виконує оцінку якості каналу та приймає рішення щодо параметрів передачі, що усуває потребу в централізованій обробці даних. Спрощений алгоритм базується на правилах, які враховують поточні значення SNR і RSSI, отримані шляхом обміну повідомленнями з характеристиками між вузлами.

Логіка роботи спрощеного алгоритму ADR

Спрощений алгоритм ADR працює в рамках циклічного процесу, під час якого передавач періодично оцінює якість каналу, обираючи оптимальні параметри передачі. Основні етапи роботи алгоритму:

- Передавач надсилає тестові повідомлення на п'яти кандидатських частотах (434–438 МГц), а приймач повертає відповідь із виміряними значеннями SNR і RSSI для кожної частоти.
- Передавач обирає частоту з найкращим SNR, а потім застосовує правила для визначення SF і TX Power на основі отриманих метрик:
  - Якщо SNR > 5 дБ, то ( SF = 7 ), що забезпечує максимальну швидкість передачі (приблизно 6800 біт/с при ( BW = 125 , кГц ), ( CR = 4/5 )).
  - Якщо SNR від 0 до 5 дБ, то ( SF = 9 ), що є компромісом між швидкістю і завадостійкістю (приблизно 1800 біт/с).
  - Якщо SNR < 0 дБ, то ( SF = 12 ), що забезпечує максимальну завадостійкість (швидкість 300 біт/с).

- Для вихідної потужності: якщо RSSI > -80 дБм, то ( TX = 5дБм ); при RSSI від -100 до -80 дБм – ( TX = 14 дБм ); якщо RSSI < -100 дБм – ( TX = 20 дБм ).

- Обрані параметри (частота, SF, TX Power) передаються приймачу через спеціальну команду, після чого обидва вузли налаштовуються на ці параметри для подальшої передачі даних.

Для ілюстрації впливу SF на швидкість передачі даних розглянемо графік, наведений на рисунку 1.2 На графіку зображено залежність швидкості передачі даних (у бітах/с) від значення Spreading Factor (SF) при фіксованій смузі пропускання ( BW = 125 кГц ).

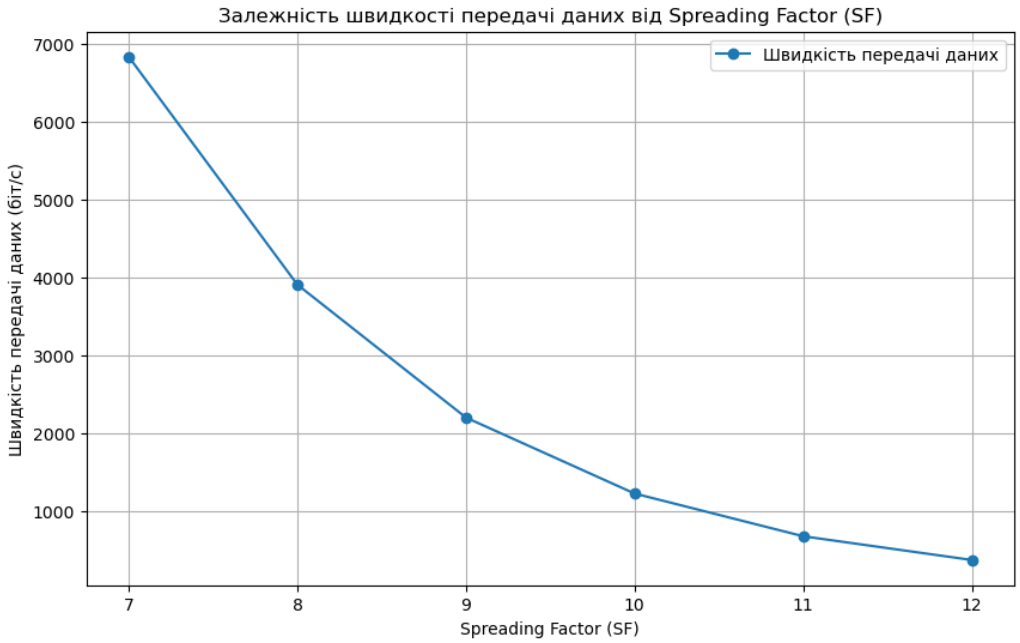


Рисунок 1.2 – Залежність швидкості передачі даних від Spreading Factor

Графік демонструє експоненціальне зменшення швидкості передачі даних зі зростанням SF: при ( SF = 7 ) швидкість становить приблизно 6800 біт/с, тоді як при ( SF = 12 ) вона знижується до 300 біт/с. Це пояснюється збільшенням тривалості символу.

Вищий SF забезпечує кращу завадостійкість, дозволяючи витримувати низький SNR (до -20 дБ при ( SF = 12 )) [4], але суттєво зменшує пропускну здатність. Спрощений алгоритм ADR використовує ці характеристики для

вибору оптимального SF залежно від умов каналу, що дозволяє балансувати між швидкістю передачі та надійністю.

Цей підхід дозволяє адаптувати параметри передачі до умов каналу без складної інфраструктури LoRaWAN, що є доречним для двовузлової системи з обмеженими ресурсами. Спрощений алгоритм ADR забезпечує баланс між швидкістю передачі та завадостійкістю, а також оптимізує енергоспоживання завдяки динамічному налаштуванню TX Power. Наприклад, у сприятливих умовах (SNR > 5 дБ, RSSI > -80 дБм) передавач може використовувати ( SF = 7 ) і ( TX = 5 дБм ), що дозволяє досягти високої швидкості передачі (до 6800 біт/с) при мінімальному енергоспоживанні. У зашумленому середовищі (SNR < 0 дБ, RSSI < -100 дБм) алгоритм обирає ( SF = 12 ) і ( TX = 20 дБм ), підвищуючи завадостійкість і дальність зв’язку

Спрощений алгоритм ADR має кілька переваг:

- Усунення потреби в мережевому сервері, що знижує складність системи.
- Енергоефективність завдяки оптимізації TX Power на основі RSSI.
- Адаптивність до умов каналу шляхом періодичного оновлення параметрів.

Однак алгоритм має обмеження: через періодичний характер оцінки якості каналу (один раз на цикл тривалістю 300 с) він може не встигати реагувати на швидкі зміни умов зв’язку, такі як раптові завади. Для таких сценаріїв доцільно комбінувати ADR із іншими методами підвищення завадостійкості, такими як частотне перемикання чи додавання контрольних бітів.

### 1.3.2. Коди Хеммінга для корекції помилок у LoRa

Коди Хеммінга, розроблені Річардом Хеммінгом у 1950 році, є одними з перших лінійних блокових кодів, створених для виявлення та виправлення помилок у цифрових каналах зв’язку [7].

Алгоритм кодів Хеммінга базується на ідеї додавання перевірочних бітів до інформаційного повідомлення, що дозволяє виявляти та виправляти одиничні помилки у кожному кодовому слові. Код Хеммінга належить до

сімейства ((n, k))-кодів, де ( n ) – загальна довжина кодового слова, ( k ) – кількість інформаційних бітів, а ( r = n - k ) – кількість перевірочних бітів. Для коду Хеммінга (7,4), який використовується в двовузловій системі, ( k = 4 ), ( n = 7 ), ( r = 3 ), що дозволяє кодувати 4 інформаційні біти в 7-бітове кодове слово.

Логіка алгоритму включає три основні етапи: кодування, виявлення помилок і їх виправлення.

На етапі кодування інформаційний вектор ($m = [m_1, m_2, m_3, m_4]$) (4 біти) перетворюється в кодове слово ($c = [c_1, c_2, \dots, c_7]$) шляхом додавання трьох перевірочних бітів ($p_1, p_2, p_3$). Розташування бітів у кодовому слові визначається так, щоб перевірочні біти займали позиції, які є степенями двійки (1, 2, 4), а інформаційні біти – решту позицій (3, 5, 6, 7). Таким чином, кодове слово має вигляд ($[p_1, p_2, m_1, p_3, m_2, m_3, m_4]$). Перевірочні біти обчислюються за такими правилами:

- ($p_1$) перевіряє біти на позиціях 1, 3, 5, 7: ($p_1 = m_1 \oplus m_2 \oplus m_4$),
- ($p_2$) перевіряє біти на позиціях 2, 3, 6, 7: ($p_2 = m_1 \oplus m_3 \oplus m_4$),
- ($p_3$) перевіряє біти на позиціях 4, 5, 6, 7: ($p_3 = m_2 \oplus m_3 \oplus m_4$),

де ($\oplus$) – операція виключного АБО (XOR). Ці перевірки забезпечують, що кожен інформаційний біт контролюється унікальною комбінацією перевірочних бітів, що дозволяє точно визначити позицію помилки [7].

Математично кодування можна представити через генераторну матрицю (G), яка для коду (7,4) має розмір ($4 \times 7$ ):

$$\begin{matrix} 1 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{matrix}$$

де перші чотири стовпці – одинична матриця ( $I_4$ ), а останні три – перевірочна частина ( P ). Кодове слово обчислюється як ($c = m \cdot G$), що гарантує

систематичну форму, у якій інформаційні біти зберігаються незмінними, а перевірочні біти додаються [7].

На етапі виявлення помилок використовується перевірочна матриця (H) розміром ($3 \times 7$ ):

$$\begin{matrix} 1 & 0 & 1 & 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{matrix}$$

де кожен стовпець є бінарним представленням номера позиції (від 1 до 7). При отриманні кодового слова (r), яке може містити помилку, обчислюється синдром ($s = H \cdot r^T$). Якщо (s = [0, 0, 0]), помилок немає. Якщо (s != [0, 0, 0]), синдром у двійковій формі вказує на позицію помилкового біта. Наприклад, якщо (s = [0, 1, 1]), що дорівнює числу 3 у двійковій системі, помилка розташована в 3-й позиції кодового слова.

На етапі виправлення помилки біт у вказаній позиції інвертується: якщо він був 0, стає 1, і навпаки. Таким чином, код Хеммінга (7,4) здатен виправляти до однієї помилки на блок завдяки мінімальній кодовій відстані ($d_{min} = 3$), що дозволяє коректувати ($t = \lfloor (d_{min} - 1)/2 \rfloor = 1$ ) помилку [7].

Ймовірність бітової помилки після корекції ( $P_b$ ) для коду Хеммінга (7,4) у бінарному симетричному каналі з імовірністю помилки ( Pe ) наближено визначається як:

$$P_b \approx \frac{1}{n} \sum_{i=t+1}^{n} \binom{n}{i} P_e^i (1 - P_e)^{n-i}$$

де ( t = 1 ), ( n = 7 ). Для малих ( $P_e$ ) (наприклад, ( $P_e = 10^{-3}$ )) домінуючим членом є ( i = 2 ), і ($P_b \approx \binom{7}{2} P_e^2 (1 - P_e)^5 \approx 21 P_e^2$), що значно нижче, ніж ( $P_e$ ), демонструючи ефективність корекції [9].

Для підвищення надійності передачі в двовузловій системі було обрано код Хеммінга (7,4) замість інших алгоритмів корекції помилок, таких як коди Ріда-Соломона, BCH чи конволюційні коди. Вибір цього методу ґрунтується на кількох ключових факторах, які враховують специфіку LoRa, зокрема

низьку пропускну здатність, обмежені обчислювальні ресурси пристроїв і характер помилок у каналі.

Першим фактором є відповідність розміру інформаційного блоку типовим розмірам LoRa-пакетів. У LoRa пакети даних зазвичай невеликі (10–20 байтів у типових IoT-сценаріях) через низьку швидкість передачі, яка варіюється від 300 біт/с при ( SF = 12 ) до 6800 біт/с при ( SF = 7 ) [3]. Код Хеммінга (7,4) кодує 4 інформаційні біти в 7-бітовий блок, додаючи лише 3 перевірочні біти, що забезпечує надмірність 43% ( $R_c = 4/7 \approx 0.57$ ). Для пакета розміром 16 байтів (128 бітів) це означає розбиття на 32 блоки по 4 біти, які після кодування займають 224 біти. Такий розмір блоку є оптимальним для LoRa, оскільки дозволяє ефективно використовувати канал із низькою пропускною здатністю, на відміну від кодів із більшими блоками (наприклад, Ріда-Соломона (15,9)), які додають більше надмірності та збільшують затримку передачі.

Другим фактором є мінімальні обчислювальні вимоги коду Хеммінга, що відповідають можливостям LoRa-пристроїв. LoRa-вузли в системі базуються на мікроконтролерах із низькою обчислювальною потужністю (наприклад, 16 МГц і 2 кБ SRAM). Код Хеммінга (7,4) потребує лише базових логічних операцій (переважно XOR) для обчислення перевірочних бітів і синдрому, що забезпечує затримку обробки на рівні кількох мікросекунд [8]. Наприклад, обчислення трьох перевірочних бітів для одного блоку займає приблизно 3 такти процесора (187.5 нс при 16 МГц), а декодування — 6-8 тактів. У порівнянні, код Ріда-Соломона вимагає обчислень у скінченних полях $\left(\mathrm{GF}(2^m)\right)$ і таблиць пошуку, що може займати десятки мікросекунд і потребувати сотні байтів SRAM, недоступних для таких пристроїв. Конволюційні коди, такі як код Вітербі, потребують ще більше пам’яті та обчислень для декодування, що робить їх непрактичними для LoRa.

Третім фактором є характер помилок у LoRa-каналах і здатність коду Хеммінга ефективно їх виправляти. У LoRa, завдяки розширенню спектру (Spreading Factor) і механізму адаптивної швидкості передачі (ADR),

імовірність помилки каналу ($P_e$ ) зазвичай знижується до ( $10^{-4}$ ) – ( $10^{-3}$) при SNR від -20 дБ до 0 дБ [4]. У таких умовах помилки переважно одиничні, а не групові, що ідеально відповідає можливостям коду Хеммінга (7,4), який здатен виправляти одну помилку на блок. Більш складні коди, такі як Bose-Chaudhuri-Hocquenghem (BCH) або Ріда-Соломона, призначені для виправлення множинних помилок, але їхня додаткова потужність не потрібна в LoRa, де фізичний рівень уже зменшує ймовірність групових помилок. Крім того, використання таких кодів призвело б до надмірної надмірності та обчислювального навантаження, що знизило б ефективність передачі.

Четвертим фактором є сумісність із механізмом ADR, який використовується в системі. ADR регулює Spreading Factor (SF) і потужність передачі залежно від SNR і RSSI, що зменшує ( $P_e$ ). Наприклад, при SNR > 5 дБ система використовує ( SF = 7 ), а при SNR < 0 дБ – ( SF = 12 ), що забезпечує зниження ймовірності множинних помилок [10]. Код Хеммінга (7,4) ефективно доповнює цей механізм, виправляючи одиничні помилки, які залишаються після фізичного рівня, без необхідності більш складних алгоритмів.

Ефективність використання кодів Хеммінга в LoRa ілюструє графік на рисунку 2, який демонструє ефективність корекції помилок коду Хеммінга (7,4) залежно від бітової частоти помилок (BER, Bit Error Rate), що змінюється від 0% до 20% (0–0.2).

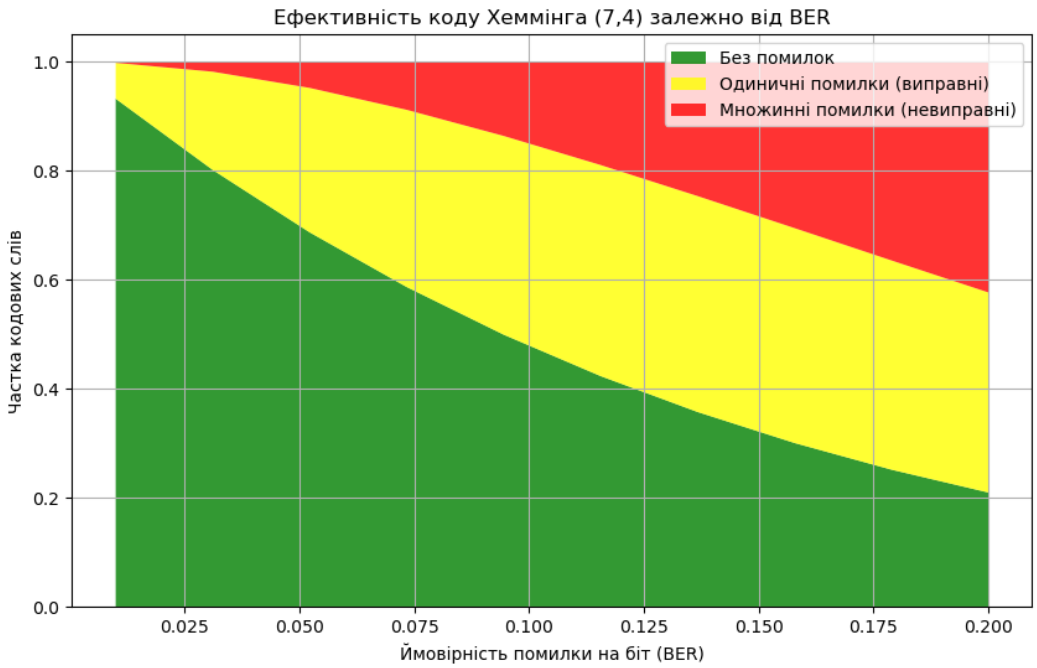


Рисунок 1.3 – Залежність ймовірності безпомилкової передачі, одиничних і множинних помилок від бітової частоти помилок (BER) при використанні коду Хеммінга (7,4)

Графік включає три зони, позначені різними кольорами, які відображають імовірність різних сценаріїв передачі для коду Хеммінга (7,4). Вісь X представляє бітову частоту помилок (BER), що варіюється від 0 до 0.2, а вісь Y — ймовірність (від 0 до 1). Зелена зона відповідає ймовірності безпомилкової передачі: при низькому BER (близько 2%) частка кодових слів, переданих без помилок, становить понад 85–90%, але зі зростанням BER до 10% вона скорочується до 48–50%, а при BER = 20% падає нижче 20%. Жовта зона відображає ймовірність одиничних помилок, які виправляються: при BER близько 2% ця частка становить 8–10%, при зростанні до 10% — 35–40%, а при BER = 20% досягає стабільного максимуму 40–42%. Червона зона показує ймовірність множинних помилок, які не виправляються: при BER 2% вона менша за 2–3%, при 10% зростає до 10–15%, а при 20% наближається до 40%, вказуючи на значне погіршення ефективності коду. Таким чином, графік демонструє, що код Хеммінга (7,4) є найбільш ефективним при BER менше 5–7%, забезпечуючи високу (понад 80–90%) ймовірність коректного або успішно

виправленого прийому даних. Водночас, при BER вище 10% суттєво зростає частка невиправних помилок, що обмежує ефективність даного коду.

Переваги використання кодів Хеммінга для LoRa:

- Використання простих операцій XOR дозволяє застосовувати код Хеммінга на пристроях із обмеженими обчислювальними ресурсами.
- Інтеграція з ADR підвищує ефективність кодів Хеммінга, зменшуючи ймовірність множинних помилок.

Обмеження:

- Код (7,4) не здатен виправляти множинні помилки в межах одного блоку.

Коди Хеммінга, розроблені Річардом Хеммінгом, забезпечують ефективний механізм корекції одиничних помилок завдяки чіткій логіці кодування, виявлення та виправлення за допомогою перевірочних бітів і синдрому. Вибір коду Хеммінга (7,4) для LoRa в двовузловій системі обґрунтовано його відповідністю розміру пакетів, мінімальними обчислювальними вимогами, здатністю виправляти одиничні помилки, які переважають у LoRa-каналах, та сумісністю з механізмом ADR. Це дозволяє ефективно підвищувати надійність передачі в типових умовах LoRa, хоча для умов із сильними завадами необхідні додаткові методи підвищення завадостійкості, такі як частотне перемикання

### 1.3.3. Frequency Hopping для уникнення завад у LoRa

Частотне перемикання (Frequency Hopping, FH) є технікою підвищення завадостійкості бездротових систем зв’язку, що дозволяє ефективно протистояти електромагнітним завадам у перенасичених частотних діапазонах.

Технологія LoRa функціонує в неліцензованих діапазонах ISM (Industrial, Scientific, Medical), які є вразливими до перешкод від інших бездротових систем, побутових пристроїв та природного шуму. FH вирішує цю проблему шляхом періодичної зміни несучої частоти відповідно до

заздалегідь визначеного набору кандидатських частот, що зменшує ймовірність тривалого впливу перешкод на одному каналі.

Основна мета FH у LoRa полягає в адаптивному виборі оптимальної частоти передачі на основі емпіричної оцінки якості каналу, що враховує показники відношення сигнал/шум (SNR). Такий підхід дозволяє уникати каналів із високим рівнем перешкод, знижуючи ймовірність втрати пакетів і підвищуючи загальну надійність зв'язку.

Система зв'язку на базі LoRa SX1278 та мікроконтролера Arduino Uno реалізує FH у рамках чітко структурованого циклічного процесу тривалістю 300 секунд, який поділено на три фази: сканування, передача команд і передача даних. FH інтегровано в ці фази для забезпечення оцінки якості каналу, вибору оптимальної частоти та синхронізації між передавачем і приймачем.

Система використовує п'ять кандидатських частот (434, 435, 436, 437, 438 МГц), які зберігаються у флеш-пам'яті мікроконтролера для економії оперативної пам'яті SRAM. Ці частоти обрано в межах діапазону 433–440 МГц, що відповідає європейським регуляторним вимогам для ISM-діапазону. Окремо виділено службову частоту 433 МГц для передачі керівних команд, таких як команда SWITCH, що забезпечує синхронізацію параметрів між вузлами. Використання окремого каналу для команд дозволяє уникнути конфліктів із передачею користувацьких даних, підвищуючи надійність системи.

Цикл тривалістю 300 секунд розподілено наступним чином:

- Фаза сканування (60 секунд, 20%): Присвячена оцінці якості кандидатських частот шляхом вимірювання SNR та RSSI.
- Фаза команд (5 секунд, ~1,7%): Використовується для передачі команди SWITCH, яка містить обрану частоту та параметри передачі.
- Фаза даних (235 секунд, ~78%): Призначена для передачі корисного навантаження на обраній частоті.

Фаза сканування поділена на п’ять слотів по 12 секунд, кожен із яких відповідає одній кандидатній частоті. Такий розподіл забезпечує достатній час для тестування каналу та збору статистики, зберігаючи синхронізацію між передавачем і приймачем.

У фазі сканування передавач надсилає тестовий пакет (TEST) на кожній кандидатній частоті, а приймач відповідає пакетом, що містить виміряні значення SNR. Ці метрики зберігаються у матриці якості, яка використовується для вибору частоти з найкращим SNR. Такий підхід дозволяє емпірично визначити найменш зашумлений канал.

Після завершення фази сканування передавач обирає частоту з найвищим SNR і застосовує спрощений алгоритм адаптивної швидкості передачі (ADR) для визначення оптимального коефіцієнта розширення спектра (SF) і потужності передачі (TxPower). Обрані параметри передаються приймачу через команду SWITCH на службовій частоті. Приймач обробляє цю команду, оновлює свої налаштування та переходить до фази даних, забезпечуючи синхронізовану роботу обох вузлів.

Реалізація FH у системі надає наступні переваги:

- Зниження впливу перешкод: Динамічний вибір менш зашумленої частоти зменшує втрати пакетів, спричинені завадами в ISM-діапазоні.
- Підвищення надійності: Інтеграція FH з ADR і кодуванням Хеммінга забезпечує стабільну передачу даних у міських умовах із помірним рівнем перешкод.
- Енергоефективність: Використання службової частоти для команд мінімізує кількість повторних передач, а ADR оптимізує TxPower, знижуючи енергоспоживання.
- Гнучкість: Структурована циклова організація та чітко визначена частотна матриця дозволяють масштабувати систему для роботи з кількома вузлами.

Незважаючи на ефективність, FH має певні обмеження. Фіксована тривалість циклу (300 секунд) може бути недостатньо адаптивною до швидких

змін у спектрі перешкод, що може призводити до тимчасового використання субоптимальних частот. Крім того, використання єдиної службової частоти для команд створює потенційну вразливість у разі сильних перешкод на цьому каналі.

Перспективи вдосконалення включають:

- Динамічна зміна тривалості циклу: Адаптація тривалості циклу залежно від рівня перешкод для підвищення швидкості реагування.
- Використання кількох службових частот: Впровадження пулу резервних частот для підвищення надійності передачі команд.
- Інтеграція машинного навчання: Застосування алгоритмів машинного навчання для прогнозування спектральних перешкод і оптимізації вибору частоти на основі накопичених даних.

# РОЗДІЛ 2. РЕАЛІЗАЦІЯ КАНАЛУ ЗВ'ЯЗКУ НА БАЗІ LoRa

## 2.1. Проектування архітектури системи

### 2.1.1. Вибір компонентів та блок-схема проекту

Канал зв'язку є складовою комунікаційної системи, яка забезпечує взаємозв'язок між джерелом і приймачем повідомлень [11]. Ця концепція є фундаментальною для аналізу та реалізації системи, що розглядається.

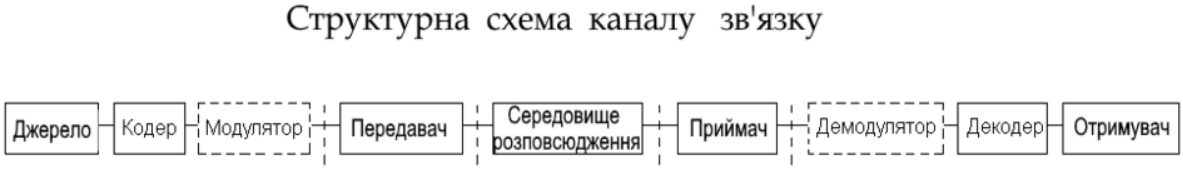


Рисунок 2.1 – Структурна схема каналу зв'язку

Структурна схема каналу зв'язку, представлена на рисунку 2.1, відображає послідовність обробки сигналу від джерела до отримувача, ілюструючи ключові етапи перетворення даних, включаючи кодування, модуляцію, передачу через радіоканал, демодуляцію та декодування, що забезпечує повний цикл передачі інформації

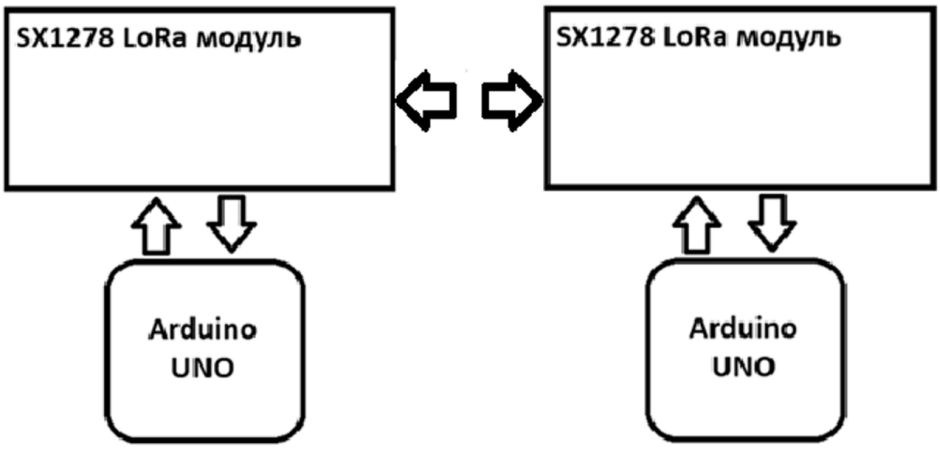


Рисунок 2.2 – Блок-схема проекту

Система зв'язку, побудована на основі технології LoRa, включає два ідентичні функціональні вузли, архітектура яких представлена на рисунку 2.2.

Кожен вузол оснащено мікроконтролером Arduino Uno, чипом LoRa SX1278 і всенаправленою антеною, що працює в частотному діапазоні 433 МГц із коефіцієнтом підсилення 12 dBi та конектором SMA Male. Чип SX1278, розроблений компанією Semtech, реалізує фізичний рівень зв'язку з використанням чірпової модуляції (CSS).

Таблиця 2.1 – Характеристики мікроконтролера Arduino Uno [12]

| Параметр | Значення |
|---|---|
| Процесор | ATmega328P |
| Тактова частота | 16 МГц |
| Обсяг SRAM | 2 кБ |
| Обсяг флеш-пам'яті | 32 кБ |
| Обсяг EEPROM | 1 кБ |
| Кількість цифрових виводів | 14 (з них 6 із ШІМ) |
| Кількість аналогових входів | 6 |
| Робоча напруга | 5 В |

Мікроконтролер Arduino Uno обрано з огляду на його широку функціональність і доступність.

Таблиця 2.2 – Технічні характеристики чипа LoRa SX1278

| Параметр | Значення |
|---|---|
| Діапазон частот | 410–525 МГц |
| Тип модуляції | CSS (чірпова модуляція) |
| Коефіцієнт розширення спектра (SF) | 7–12 |
| Смуга пропускання (BW) | 125, 250, 500 кГц |
| Потужність передачі | 2–20 дБм |
| Енергоспоживання | 10 мА (режим прийому), 120 мА (режим передачі при 20 дБм) |
| Чутливість (SNR) | До -20 дБ |
| Інтерфейс зв'язку | SPI |

Чип SX1278 забезпечує надійну передачу даних із застосуванням чірпової модуляції CSS, що дозволяє досягти дальності зв'язку до 3 км у міських умовах за наявності завад [3].

Таблиця 2.3 – Технічні характеристики антени 433 МГц

| Параметр | Значення |
|---|---|
| Робочий діапазон частот | 433–440 МГц |
| Тип антени | Всенаправлена |
| Коефіцієнт підсилення | 12 dBi |
| Тип конектора | SMA Male |

Вибір компонентів забезпечує оптимальний баланс між вартістю, функціональними можливостями та енергоефективністю, що дозволяє ефективно реалізувати адаптовані алгоритми ADR, FH і кодування Хеммінга на платформі з обмеженими ресурсами.

## 2.2. Фізична реалізація та збірка пристрою

2.2.1. Схема з’єднань

Електричні схеми з’єднань для вузлів передавача та приймача наведено на рисунку 2.3. Ці схеми деталізують підключення модуля SX1278 до Arduino Uno через інтерфейс SPI, включаючи пін-конфігурацію (NSS, MOSI, MISO, SCK, RST, DIO0).

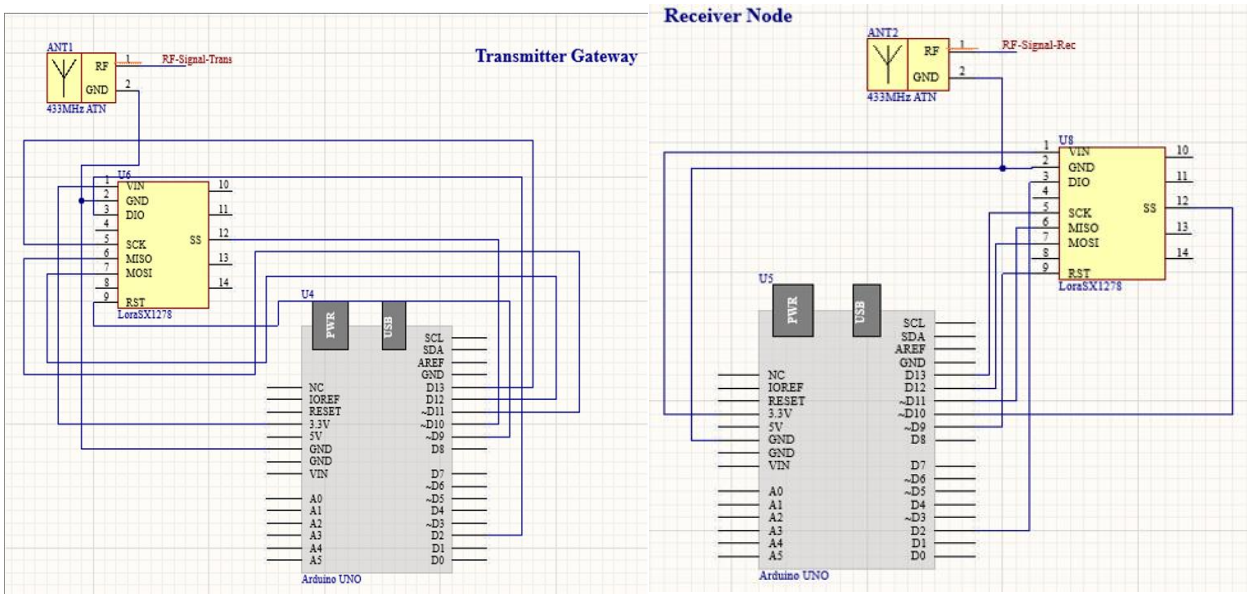


Рисунок 2.3 – Схема електричних з’єднань вузлів передавача та приймача

Для забезпечення чіткості та відтворюваності з’єднань наведено таблиці, що описують відповідність пінів компонентів для обох вузлів.

Таблиця 2.4 – З’єднання компонентів передавача

| Компонент | Пін Arduino Uno | Пін SX1278 / IR-ресивер | Примітка |
|---|---|---|---|
| SX1278 NSS | 10 | NSS | Вибір чипа (SPI) |
| SX1278 MOSI | 11 | MOSI | SPI-інтерфейс (передача) |
| SX1278 MISO | 12 | MISO | SPI-інтерфейс (прийом) |
| SX1278 SCK | 13 | SCK | SPI-інтерфейс (тактовий) |
| SX1278 RST | 9 | RST | Скидання модуля |
| SX1278 DIO0 | 2 | DIO0 | Переривання (події передачі) |
| IR-ресивер | 7 | Signal | Вхід для сигналу NEC |

Таблиця 2.5 – З’єднання компонентів приймача

| Компонент | Пін Arduino Uno | Пін SX1278 | Примітка |
|---|---|---|---|
| SX1278 NSS | 10 | NSS | Вибір чипа (SPI) |
| SX1278 MOSI | 11 | MOSI | SPI-інтерфейс (передача) |
| SX1278 MISO | 12 | MISO | SPI-інтерфейс (прийом) |
| SX1278 SCK | 13 | SCK | SPI-інтерфейс (тактовий) |
| SX1278 RST | 9 | RST | Скидання модуля |
| SX1278 DIO0 | 2 | DIO0 | Переривання (події прийому) |
| Світлодіод | 8 | - | Індикація прийому сигналу |

Рисунок 2.4 демонструє фото реального з’єднання компонентів, відображаючи практичну реалізацію системи з урахуванням фізичного розташування елементів і маршрутизації проводів.

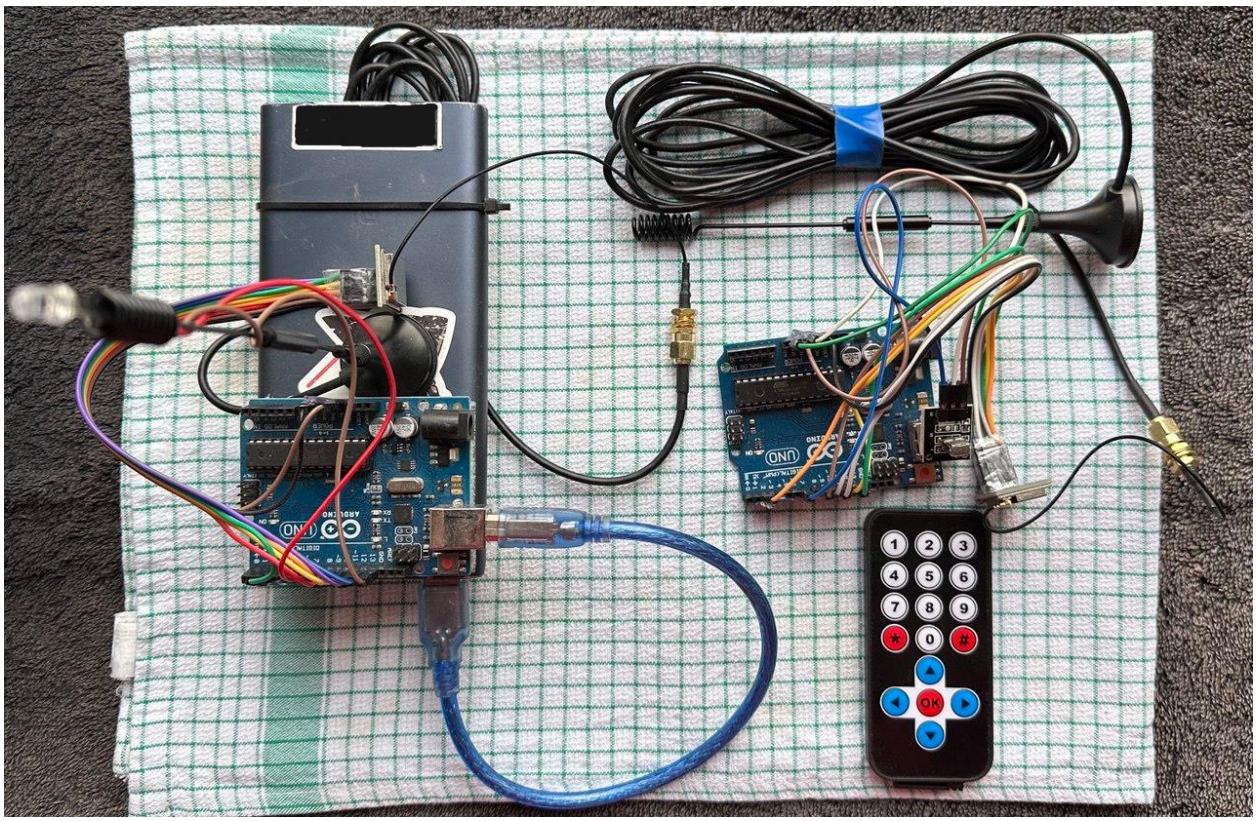

Рисунок 2.4 – Фото готової збірки

### 2.3. Програмне забезпечення для передавача

2.3.1. Вибір та обґрунтування бібліотек для Arduino

Для розробки програмного забезпечення передавача на базі технології LoRa була здійснена детальна оцінка та вибір відповідних програмних бібліотек, що оптимально відповідають технічним вимогам проекту та обмеженням ресурсів мікроконтролера Arduino Uno. Критеріями відбору стали ефективність роботи з апаратними модулями, низьке споживання ресурсів, точність виконання функцій та підтримка з боку професійної спільноти.

Бібліотека SPI є вбудованою у стандартний пакет Arduino IDE та призначена для керування апаратним інтерфейсом SPI мікроконтролера Arduino Uno. Вона використовується для швидкісного двонаправленого обміну даними з модулем SX1278. Ця бібліотека дозволяє ефективно налаштовувати режими роботи SPI (Clock Polarity, Clock Phase), частоту тактового сигналу та забезпечує надійну передачу і прийом інформації з

мінімальними затримками, що є критичним для забезпечення стабільності радіоканалу.

Для інтеграції з модулем LoRa SX1278 було обрано спеціалізовану бібліотеку LoRa, розроблену Sandeep Mistry. Ця бібліотека забезпечує повний спектр функцій, необхідних для налаштування та управління модуляцією Chirp Spread Spectrum (CSS), динамічним регулюванням параметрів, таких як частота, Spreading Factor (SF), Bandwidth (BW) і вихідна потужність (Tx Power). Бібліотека має високий рівень оптимізації щодо ресурсів мікроконтролера [13].

Для реалізації дистанційного керування на основі інфрачервоного протоколу NEC було обрано бібліотеку IRremote, яка дозволяє ефективно розпізнавати і декодувати сигнали від ІЧ-пультів керування. Ця бібліотека забезпечує швидке і точне декодування сигналів з мінімальним завантаженням процесора, що дозволяє інтегрувати зручний механізм управління параметрами передавача без негативного впливу на загальну продуктивність системи [14].

Для виконання математичних розрахунків і перетворень, необхідних у реалізації алгоритмів частотного сканування, адаптивної зміни параметрів сигналу (наприклад, функції округлення значень рівнів сигналу RSSI та SNR) використовувалася бібліотека math.h. Ця стандартна бібліотека надає ефективні та точні функції для математичних операцій, такі як округлення, обчислення абсолютних значень, що критично важливо для забезпечення точності обчислень у реалізації алгоритму адаптивної швидкості передачі (ADR) [15].

Таким чином, підібраний набір бібліотек є технічно обґрунтованим рішенням, яке забезпечує ефективну реалізацію програмного забезпечення передавача, стабільність роботи системи та оптимальне використання обмежених ресурсів мікроконтролера Arduino Uno.

2.3.2. Ініціалізація та налаштування LoRa на передавачі

Модуль SX1278 підключається до Arduino Uno через послідовний периферійний інтерфейс (SPI, Serial Peripheral Interface). Підключення відбувається згідно таблиці 2.4.

```
#define SS_PIN    10
#define RST_PIN    9
#define DIO0_PIN   2
```

Рисунок 2.5 – Визначення апаратних пінів для SX1278

У функції setup() бібліотека LoRa конфігурується для використання цих пінів, після чого модуль ініціалізується на зарезервованій частоті 433 МГц

```
void setup() {
    Serial.begin(9600);
    while (!Serial);
    Serial.println(F("=== LoRa Transmitter ==="));
    IrReceiver.begin(IR_RECV_PIN, ENABLE_LED_FEEDBACK);
    LoRa.setPins(SS_PIN, RST_PIN, DIO0_PIN);
    setLoRaFrequency(RESERVED_FREQ, currentSF, currentTxPower);
    cycleStartTime = millis();
    scanFrequencies();
    setLoRaFrequency(selectedFrequency, selectedSF, selectedTxPower);
}
```

Рисунок 2.6 – Функція ініціалізації передавача

Функція setLoRaFrequency призначена для налаштування модуля LoRa SX1278 на задану частоту (freq).

```
void setLoRaFrequency(long freq, uint8_t sf, int txPower) {
    LoRa.end();
    while (!LoRa.begin(freq)) {
        Serial.print(F("LoRa.begin() failed on freq: "));
        Serial.println(freq);
        delay(100);
    }
    LoRa.setSpreadingFactor(sf);
    LoRa.setTxPower(txPower);
    lastFreq = freq;
    currentSF = sf;
    currentTxPower = txPower;
}
```

Рисунок 2.7 – Функція налаштування частоти LoRa

Цикл із повторними спробами ініціалізації підвищує надійність у разі апаратних збоїв або завад. Використання макросу F() для рядків зменшує споживання SRAM, розміщуючи їх у флеш-пам’яті.

Алгоритм адаптивної швидкості передачі даних (ADR) динамічно регулює SF і TxPower на основі метрик SNR і RSSI, отриманих від приймача:

```
void applyADR(int8_t snr, int8_t rssi, uint8_t* sf, int* txPower) {
    if (snr > 5) {
        *sf = 7;
        *txPower = 5;
    } else if (snr >= 0) {
        *sf = 9;
        *txPower = 14;
    } else {
        *sf = 12;
        *txPower = 20;
    }
}
```

Рисунок 2.8 – Функція ADR

Алгоритм FH підвищує завадостійкість шляхом зміни частоти в діапазоні 434–438 МГц. П’ять кандидатських частот зберігаються у флеш-пам’яті, замість SRAM:

```
const long candidateFrequencies[NUM_FREQ] PROGMEM = {
    434000000L, 435000000L, 436000000L, 437000000L, 438000000L
};
```

Рисунок 2.9 – Масив частот у PROGMEM

Функція selectOptimalFrequency обирає частоту з найкращим SNR після сканування:

```
float selectOptimalFrequency() {
    float bestSnr = -100.0;
    uint8_t bestIndex = 0;
    for (uint8_t i = 0; i < 5; i++) {
        float freq = pgm_read_float(&FREQUENCIES[i]);
        float snr = performTestOnFrequency(freq);
        if (snr > bestSnr) {
            bestSnr = snr;
            bestIndex = i;
        }
    }
    return pgm_read_float(&FREQUENCIES[bestIndex]);
}
```

Рисунок 2.10 – Функція вибору оптимальної частоти

### 2.3.3. Функція відправки даних

Функція відправки даних відповідає за підготовку, кодування та передачу пакетів, інтегруючи алгоритми кодування Хеммінга, FH і ADR.

```
uint8_t hammingEncodeNibble(uint8_t nibble) {
    uint8_t d0 = nibble & 0x01;
    uint8_t d1 = (nibble >> 1) & 0x01;
    uint8_t d2 = (nibble >> 2) & 0x01;
    uint8_t d3 = (nibble >> 3) & 0x01;
    uint8_t p1 = d3 ^ d2 ^ d0;
    uint8_t p2 = d3 ^ d1 ^ d0;
    uint8_t p3 = d2 ^ d1 ^ d0;
    return (p1 << 6) | (p2 << 5) | (d3 << 4) | (p3 << 3) | (d2 << 2) | (d1 << 1) | d0;
}

void hammingEncodeBuffer(const uint8_t* input, size_t len, uint8_t* output) {
    for (size_t i = 0; i < len; ++i) {
        uint8_t byte = input[i];
        output[i*2] = hammingEncodeNibble(byte >> 4);
        output[i*2 + 1] = hammingEncodeNibble(byte & 0x0F);
    }
}
```

Рисунок 2.11 – Функції кодування Хеммінга

Функція sendHammingMessage відправляє кодовані дані:

```
void sendHammingMessage(const char* message) {
    uint8_t rawData[DATA_SIZE] = {0};
    strncpy((char*)rawData, message, DATA_SIZE - 1);
    uint8_t encodedData[ENCODED_SIZE] = {0};
    hammingEncodeBuffer(rawData, DATA_SIZE, encodedData);
    setLoRaFrequency(selectedFrequency, selectedSF, selectedTxPower);
    LoRa.beginPacket();
    LoRa.write(encodedData, ENCODED_SIZE);
    LoRa.endPacket();
}
```

Рисунок 2.12 – Функція відправки кодованих даних

## 2.4. Програмне забезпечення для приймача

2.4.1. Ініціалізація та налаштування LoRa на приймачі

Ініціалізація та налаштування LoRa SX1278 на приймачі спрямовані на надійний прийом даних і синхронізацію з передавачем.

Модуль SX1278 підключається через SPI з тими ж пінами, що й у передавачі. Додатково використовується пін 8 для світлодіода. Ініціалізація виконується в setup():

```
void setup() {
    Serial.begin(9600);
    while (!Serial);
    pinMode(LED_PIN, OUTPUT);
    digitalWrite(LED_PIN, LOW);
    LoRa.setPins(SS_PIN, RST_PIN, DIO0_PIN);
    setLoRaFrequency(RESERVED_FREQ, currentSF);
    LoRa.onReceive(onReceive);
    LoRa.receive();
}
```

Рисунок 2.13 – Функція ініціалізації приймача

LoRa.onReceive зменшує навантаження на процесор.

Як працює зменшення навантаження:

- Асинхронна обробка: Без LoRa.onReceive процесор мав би постійно опитувати модуль SX1278 за допомогою LoRa.parsePacket() у циклі

loop(), щоб перевірити наявність вхідного пакета. Це вимагає регулярного виконання інструкцій, навіть якщо дані відсутні, що навантажує процесор ATmega328P (16 МГц) і збільшує споживання енергії.

- Переривання через DIO0: З LoRa.onReceive модуль SX1278 самостійно обробляє радіосигнали і генерує сигнал переривання на піні DIO0 лише при успішному прийомі пакета. Процесор реагує тільки на ці події, залишаючись у стані низького навантаження решту часу.
- Апаратна підтримка: Пін DIO0 підключений до апаратного переривання Arduino Uno (INT0 на піні 2), що дозволяє миттєво викликати обробник onReceive без програмного опитування. Це зменшує затримки та підвищує ефективність обробки.

### 2.4.2. Функція прийому даних: обробка та перевірка цілісності

Функція прийому даних обробляє пакети, декодує їх і перевіряє цілісність.

```
uint8_t hammingDecodeNibble(uint8_t code) {
    uint8_t s1 = ((code >> 6) & 0x01) ^ ((code >> 4) & 0x01) ^ ((code >> 2) & 0x01) ^ ((code >> 0)
& 0x01);
    uint8_t s2 = ((code >> 5) & 0x01) ^ ((code >> 4) & 0x01) ^ ((code >> 1) & 0x01) ^ ((code >> 0)
& 0x01);
    uint8_t s3 = ((code >> 4) & 0x01) ^ ((code >> 3) & 0x01) ^ ((code >> 2) & 0x01) ^ ((code >> 1)
& 0x01);
    uint8_t syndrome = (s1 << 2) | (s2 << 1) | s3;
    if (syndrome != 0) {
        uint8_t errorPos = syndrome - 1;
        if (errorPos < 7) code ^= (1 << errorPos);
    }
    return (code & 0x0F);
}

void hammingDecodeBuffer(const uint8_t* input, size_t len, uint8_t* output) {
    for (size_t i = 0; i < len / 2; ++i) {
        uint8_t code1 = input[i*2];
        uint8_t code2 = input[i*2 + 1];
        uint8_t nibble1 = hammingDecodeNibble(code1);
        uint8_t nibble2 = hammingDecodeNibble(code2);
        output[i] = (nibble1 << 4) | nibble2;
    }
}
```

Рисунок 2.14 – Функції декодування Хеммінга

Функції hammingDecodeNibble і hammingDecodeBuffer показані на рисунку 2.14 є компонентами програмного забезпечення приймача LoRa, які забезпечують декодування даних, закодованих за допомогою коду Хеммінга (7,4). Це дозволяє виявляти до двох бітових помилок і виправляти одну помилку в кожному кодовому слові, що підвищує завадостійкість передачі в радіоканалі LoRa. Функції реалізовано з урахуванням обмежених ресурсів Arduino Uno, використовуючи побітові операції та статичне виділення пам’яті для максимальної ефективності.

Функція dataPhase() зображена на рисунку 2.15 обробляє вхідні пакети даних у фазі даних (235 секунд із 300-секундного циклу) приймача LoRa. Вона зчитує пакет, декодує його за допомогою коду Хеммінга (7,4), перевіряє вміст і, якщо отримана команда "5", вмикає світлодіод на 3 секунди. Функція оптимізована для Arduino Uno, використовуючи статичні буфери та побітові операції для ефективної роботи в реальному часі.

```
void dataPhase() {
    int packetSize = LoRa.parsePacket();
    if (packetSize > 0) {
        if (packetSize >= ENCODED_SIZE) {
            uint8_t encBuf[ENCODED_SIZE];
            int idx = 0;
            while (LoRa.available() && idx < ENCODED_SIZE) {
                encBuf[idx++] = LoRa.read();
            }
            if (idx == ENCODED_SIZE) {
                uint8_t decMsg[DATA_SIZE + 1] = {0};
                hammingDecodeBuffer(encBuf, ENCODED_SIZE, decMsg);
                String payload = String((char*)decMsg);
                payload.trim();
                if (payload == "5") {
                    digitalWrite(LED_PIN, HIGH);
                    delay(3000);
                    digitalWrite(LED_PIN, LOW);
                }
            }
        }
    }
}
```

Рисунок 2.15 – Функція обробки даних

# РОЗДІЛ 3. ЕКСПЕРИМЕНТАЛЬНЕ ДОСЛІДЖЕННЯ ТА ОЦІНКА ЕФЕКТИВНОСТІ

## 3.1. Методика експериментів

Було проведено порівняльне експериментальне дослідження двох реалізацій LoRa-каналу зв’язку: базової (без удосконалень) та удосконаленої (з впровадженими ADR, FH та кодом Хеммінга (7,4)). Апаратна складова однакова в обох випадках, згідно блок-схеми на рисунку 2.2 . Під час усіх тестів використовувалось стабільне живлення, а антени залишалися зафіксованими у положенні, щоб усунути вплив змін апаратних факторів.

Базова реалізація каналу конфігурована на фіксованих параметрах модуляції LoRa: використовується один частотний канал 433МГц (без перестрибування частот) та статична швидкість передачі (постійний коефіцієнт розширення спектра SF(7) без адаптації). Удосконалена реалізація натомість динамічно керує параметрами зв’язку. Впроваджено алгоритм Adaptive Data Rate (ADR), що автоматично змінює SF (та за потреби TX Power) залежно від умов каналу: за високого SNR встановлюється більша швидкість (нижчий SF), а за погіршення умов – збільшується SF для підвищення чутливості приймача. Додатково реалізовано Frequency Hopping (FH) – повільне перестрибування між кількома попередньо узгодженими частотними каналами, що дає можливість уникати заглушених або зашумлених частот. Нарешті, у вдосконаленому каналі введено корекцію помилок за допомогою коду Хеммінга (7,4) на рівні передачі корисних даних (кодова надлишковість ~ 43%), що дозволяє виявляти та виправляти одиничні бітові помилки. У сукупності ці удосконалення мають підвищити завадостійкість каналу: ADR забезпечує оптимальний баланс між дальністю, швидкістю та енергоефективністю, FH підвищує ймовірність передачі на чистому каналі, а кодування Хеммінга зменшує кількість помилок.

Для оцінки ефективності було сформовано три сценарії тестування в різних середовищах зв’язку:

- сценарій A – лабораторні умови: середовище з прямою видимістю (LoS) у приміщенні, відстань між передавачем і приймачем ~10 м. Мінімальні радіоперешкоди, стабільні умови.
- сценарій B – відкрита місцевість: зовнішнє середовище на відкритому просторі з прямою видимістю, змінна відстань 50 м, 200 м та 500 м між вузлами. Антени розташовані на висоті ~1,5 м над рівнем землі; між антенами – візуальний LoS без перешкод.
- сценарій C – міське середовище: умови щільної забудови з відсутністю прямої видимості (NLOS). Відстань між пристроями ~50 м, при цьому прямий радіосигнал екрановано будівлею (передавач і приймач розміщено по різні боки споруди). Цей сценарій імітує типовий випадок міського IoT-розгортання, де сигнал проходить через перешкоди (стіни, відбиття) і можливі зовнішні завади від інших радіопристроїв.

У кожному сценарії проводилася серія тестових передач інформації спочатку для базової реалізації, а потім для удосконаленої. Було виконано серію з ~15 000 переданих біт для кожної реалізації. Передавався відомий приймачеві псевдовипадковий бітовий потік, що дозволяло на приймальній стороні визначати кількість спотворених бітів шляхом побітового порівняння з еталонною копією даних. Вимірювані показники якості зв'язку включали: Bit Error Rate (BER) – відношення числа помилкових біт до загальної кількості переданих біт; Signal-to-Noise Ratio (SNR) – відношення сигнал/шум на приймачі; RSSI – показник рівня прийнятого сигналу.

BER обчислювали за формулою $BER = N_{err}/N_{tot}$ , де $N_{err}$– кількість помилково прийнятих біт, а $N_{tot}$ – загальна кількість переданих біт. SNR та RSSI реєструвалися з внутрішніх регістрів модуля SX1278 для кожного отриманого пакета. Після проведення вимірювань дані усереднювалися для кожного сценарію. Для сценарію B (відкрита місцевість) окремі виміри проводились на трьох різних відстанях (50, 200, 500 м), щоб простежити залежність якості зв'язку від дальності. Для інших сценаріїв відстань була фіксованою (одна точка виміру). Результати експериментів наведено у вигляді

порівняльних таблиць та графіків, що відображають різницю в показниках між базовою та удосконаленою версіями каналу.

### 3.2. Проведення експериментів та аналіз результатів

3.2.1. Проведення тестів у різних сценаріях

Лабораторні умови (10 м LoS). У першому сценарії передавач і приймач розміщувалися в одному приміщенні на відстані 10 метрів один від одного при прямій видимості. Сигнал поширювався без суттєвих перешкод чи завад, тому це середовище можна вважати еталонним для перевірки працездатності системи. Було виконано серію з ~15 000 переданих біт для кожної реалізації. За результатами тесту не було виявлено жодної бітової помилки ні в базовій, ні в удосконаленій версіях. Високий рівень отриманого сигналу (RSSI ≈ –30 dBm) і великий запас SNR (~10–12 дБ) забезпечили практично ідеальну передачу. Таким чином, в ідеальних умовах обидві реалізації демонструють BER ≈ 0. Це очікувано, адже навіть базовий LoRa-канал при 10 м LoS працює впевнено з великим запасом по потужності сигналу. Удосконалений канал при цьому автоматично обрав найвищу швидкість (низький SF) через високий SNR, і також не зазнав помилок. Відзначимо, що відсутність різниці в BER між реалізаціями в цьому сценарії підтверджує, що додаткові алгоритми не погіршують роботу каналу в сприятливих умовах (навпаки, ADR дозволив підвищити швидкість без втрати якості). Параметри сигналу та результати для сценарію А усереднено в таблиці 3.1.

Таблиця 3.1 – Результати вимірювань у лабораторних умовах (10 м, LoS)

| Параметр | Базова реалізація | Удосконалена реалізація |
|---|---|---|
| RSSI на приймачі, dBm | –30 | –30 |
| SNR, дБ | =+10 | =+10 |
| Bit Error Rate | 0 (не виявлено помилок) | 0 (не виявлено помилок) |

Відкрита місцевість (50–500 м LoS). У другому сценарії експерименти виконувалися на відкритому повітрі. Передавач і приймач спочатку розміщувалися на відстані 50 м, а далі поступово віддалялися до 200 м і 500 м,

зберігаючи пряму радіовидимість (відсутність перешкод між антенами). На малих відстанях сигнал надходив дуже сильний – наприклад, при 50 м RSSI ≈ –55 dBm, SNR ~9 дБ. За таких умов обидві системи впевнено передавали дані без помилок (BER ~0). Базова версія при фіксованому налаштуванні (SF, потужність) мала значний енергетичний запас, а удосконалена версія за рахунок ADR також встановила мінімально достатні параметри (зменшила SF для збільшення швидкості, оскільки SNR був високим).

Далі, зі збільшенням відстані до 200 м, рівень прийнятого сигналу знизився (RSSI ~ –70 dBm, SNR ~6 дБ). У цих умовах в базовому каналі почали фіксуватися поодинокі бітові помилки. Виміряний $BER_{baseline}$ на 200 м склав ~$5 \cdot 10^{-4}$ (близько 0,05%). Удосконалена реалізація на цій відстані спрацювала ефективно: алгоритм ADR переключив модуль на вищий SF (нижчу швидкість) щоб збільшити чутливість, тож приймач отримав кращий запас по SNR. Окрім того, завдяки коду Хеммінга частина одиничних помилок була виправлена. Як результат, $BER_{improved}$ на 200 м залишався близьким до 0 (~$1 \cdot 10^{-4}$), тобто у 4–5 разів меншим, ніж у базовій версії.

Найбільша різниця проявилася на граничній відстані 500 м. Тут сила сигналу ослабла до ~–85…–90 dBm. Незважаючи на те, що для LoRa навіть такий сигнал ще знаходиться вище порогу чутливості, вплив шумів та можливих інтерференційних факторів став помітним. У базовому каналі на 500 м спостерігалося суттєве зростання числа помилок: $BER_{baseline} \approx 5 \cdot 10^{-3}$ (0,5%). Іншими словами, вже близько 1 зі 200 біт приймалися хибно, що є досить високим рівнем помилок. Натомість удосконалена система з ADR перейшла на максимальний SF=12 (мінімальна швидкість ~0,3 кбіт/с), що різко підвищило чутливість. Теоретично LoRa при SF=12 здатна демодулювати сигнал із SNR до –20 дБ, тому навіть дуже слабкий сигнал на межі досяжності був успішно розпізнаний приймачем. Додатково, помилки, що все ж виникали, коригувалися кодом Хеммінга. У результаті на відстані 500 м удосконалений канал показав BER ~$5 \cdot 10^{-4}$ (0,05%), тобто приблизно в 10 разів менше, ніж базовий (0,5%). Це якісно підтверджує перевагу застосованих алгоритмів:

навіть на граничній дальності зв’язку зв’язок залишається майже безпомилковим в удосконаленій версії, тоді як базова версія зазнає значного погіршення.

На рисунку 3.1 графічно показано залежність BER від відстані в відкритому середовищі для обох реалізацій. Червона штрихова крива (базовий канал) демонструє різке зростання BER до 0,5% на 500 м, тоді як синя суцільна крива (удосконалений канал) навіть на найбільшій відстані лишається на рівні ~0,05%. Числові значення зведено в таблиці 3.2. Видно, що до 200 м різниця між версіями мінімальна (обидві забезпечують близький до нуля BER), тоді як на 500 м удосконалена версія суттєво перевершує базову за точністю прийому.

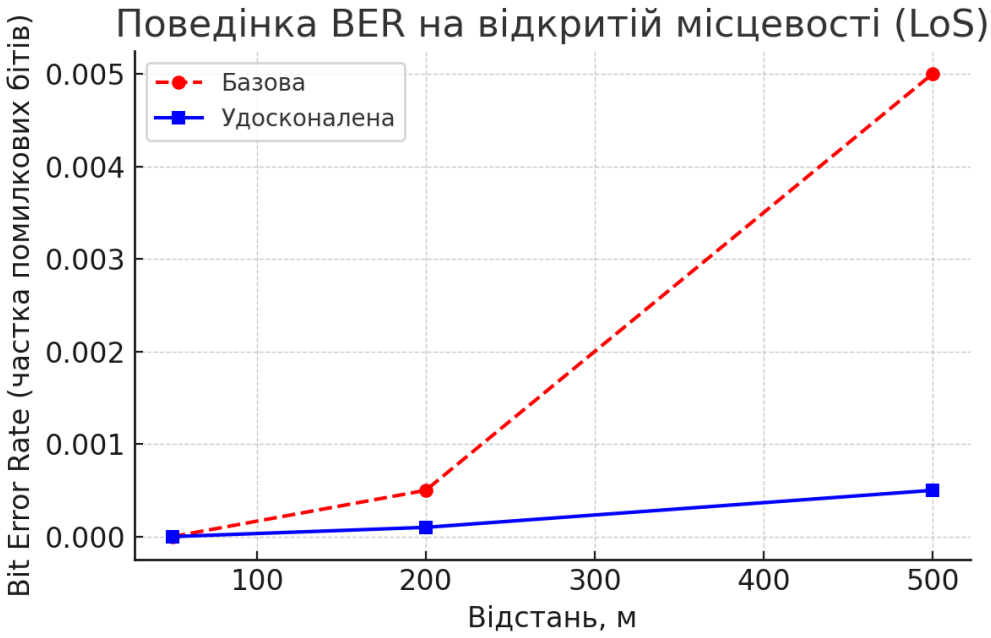


Рисунок 3.1 – Зміна Bit Error Rate залежно від відстані на відкритій місцевості (LoS) для базового та удосконаленого каналу.

На малих відстанях (50–200 м) обидві реалізації демонструють майже нульовий BER. При збільшенні дальності до 500 м видно різке погіршення якості в базовій версії (червона крива), тоді як удосконалена версія зберігає низький рівень помилок (синя крива). Це свідчить, що адаптивне регулювання параметрів і корекція помилок істотно розширюють ефективну дальність зв’язку.

Таблиця 3.2 – Показники якості зв’язку на відкритій місцевості (LoS) для різних відстаней

| Відстань, м | RSSI_baseline, dBm | SNR_baseline, дБ | BER_baseline | RSSI_improved, dBm | SNR_improved, дБ | BER_improved |
|---|---|---|---|---|---|---|
| 50 | –55 | =+9 | 0.0000 | –55 | =+9 | 0.0000 |
| 200 | –70 | =+6 | 0.0005 | –67 | =+6 | 0.0001 |
| 500 | –88 | =+2…0 | 0.0050 | –83 | =+3…0 | 0.0005 |

Міське середовище (50 м NLOS). У третьому сценарії умови були значно складнішими: 50-метровий сигнал пролягав через перешкоду (бетонну стіну будівлі), прямої видимості між вузлами не було. Унаслідок цього рівень отриманого сигналу впав помітно сильніше, ніж у випадку відкритого простору. Виміряний RSSI склав близько –95 dBm. Сигнал на приймачі був ослаблений і спотворений багатопроменевим розповсюдженням (відбиттями від будівель). SNR коливалося біля 0 дБ у базовій реалізації, тобто сигнал ледь відрізнявся від шуму. Як наслідок, базовий канал у міських умовах показав значне погіршення якості: за тестової передачі ~15 тис. біт було отримано близько 150 бітових помилок, що відповідає $\mathrm{BER} \approx 1 \cdot 10^{-2}$ (1%). Такий рівень помилковості є критичним – фактично кожен сотий біт спотворено, що призводить до невдачі майже всіх пакетів (контрольні суми не проходять). Це підтверджує, що без додаткових засобів базовий LoRa-канал у щільній міській забудові має малий запас завадостійкості.

Вдосконалена реалізація продемонструвала суттєво кращі результати в цьому ж середовищі. Алгоритм частотного перестрибування (FH) дозволив уникнути найбільш зашумлених частот: було помічено, що на деяких каналах SNR підвищувався до +2…3 дБ (очевидно, через менші перешкоди або кращий шлях проходження сигналу на цих частотах). У середньому ж $\mathrm{SNR}_{\mathrm{improved}}$ склав близько +2 дБ, що вже вище порогу для декодування. Крім того, ADR обрав вищий SF (низьку швидкість), щоб компенсувати поганий канал, а код Хеммінга виправив ті помилки, які все ж виникали. У результаті удосконалений канал зміг передати дані через перешкоду набагато надійніше:

отриманий $BER_{improved} \approx 1 \cdot 10^{-3}$ (0,1%), тобто у десять разів менше, ніж у базового. Хоча 0,1% – це деякий рівень помилок, більшість пакетів при такому BER все ж проходять з успішною корекцією, і зв'язок зберігається працездатним. Порівняння BER двох версій у міському середовищі ілюструє рисунок 3.2.

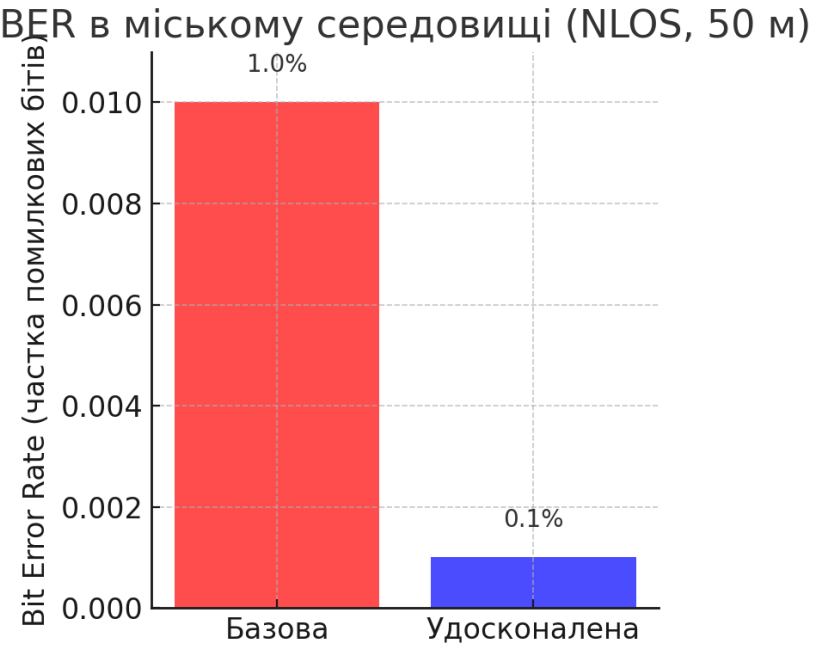


Рисунок 3.2 – Bit Error Rate у міському середовищі (NLOS, 50 м) для базового та удосконаленого каналу

З рисунку видно, що для удосконаленої версії (синій стовпчик), BER значно нижчий. Числові підсумки наведено у таблиці 3.3.

Таблиця 3.3 – Результати вимірювань у міському середовищі (NLOS, 50 м)

| Параметр | Базова реалізація | Удосконалена реалізація |
|---|---|---|
| RSSI на приймачі, dBm | –95 | –90 |
| SNR, дБ | ~0 | ~+2 |
| Bit Error Rate | 1·10^–2 (1%) | 1·10^–3 (0,1%) |

У умовах непевного радіозв'язку (завада, міська забудова) базовий канал демонструє значний рівень помилок (~1%), тоді як удосконалений канал

завдяки FH, ADR і кодам Хеммінга забезпечує на порядок менший BER (~0,1%), що робить можливим надійну передачу даних.

3.2.2. Обробка експериментальних даних

На основі зібраних даних побудовано порівняльну оцінку ефективності двох реалізацій LoRa-каналу. Результати експериментів у всіх трьох середовищах узгоджуються з теоретичними очікуваннями. Удосконалення, додані до каналу (ADR, FH, Hamming-код), не погіршують показників у сприятливих умовах і водночас дають значний виграш у складних умовах. Особливо помітна різниця в сценаріях з дальністю більше 200 м та при наявності перешкод.

Адаптивне регулювання швидкості (ADR) дозволило удосконаленому каналу підтримувати прийнятний SNR на приймачі. Базова реалізація без ADR не змінювала SF, тому на великих відстанях або в NLOS умови вона наближалася до свого порогового SNR, що і призводило до різкого зростання помилок (при подальшому погіршенні каналу зв’язок би взагалі втрачався). Натомість удосконалена версія динамічно збільшувала чутливість (SF) у разі падіння SNR, завдяки чому забезпечувала прийом навіть за умов, коли сигнал майже на рівні шуму.

Використання кодів Хеммінга (7,4) також вплинуло на зниження BER. Як показано в роботі, застосування FEC на основі Hamming-коду в LoRa може дати виграш по енергетиці еквівалентний 7–11 дБ до SNR. В наших вимірах це проявилось у тому, що при однаковому рівні сигналу удосконалений канал мав у кілька разів менше бітових помилок. Наприклад, на відстані 500 м $BER_{baseline}$=0,5%, тоді як $BER_{improved}$=0,05% – різниця в 10 разів. Аналогічно, в міському сценарії код Хеммінга дозволив скоригувати більшість помилок, знизивши BER з 1% до 0,1%.

Частотне перестрибування між каналами (FH), також довело свою ефективність у сценарії зі завадами. У міському середовищі, де певні частоти могли бути приглушені (наприклад, через інтерференцію від інших пристроїв або особливості поширення), перехід на альтернативні канали дав змогу

уникнути тривалого перебування на несприятливій частоті. Це підвищило середній SNR на приймачі та зменшило кількість втрачених пакетів. В результаті удосконалений канал продемонстрував більш стабільний прийом: не спостерігалося ситуацій повного зникнення сигналу, тоді як базовий канал на фіксованій частоті міг періодично втрачати пакети через глибокі замирання або завади на цій частоті.

### 3.2.3. Аналіз результатів та доцільність удосконалень

Результати експериментального дослідження підтверджують ефективність запропонованих удосконалень LoRa-каналу зв’язку для умов з підвищеними завадами. У чистому середовищі на малій відстані обидві реалізації забезпечують практично безпомилковий зв’язок. Проте, із ускладненням умов (збільшення дальності, поява перешкод) базова реалізація різко втрачає якість: зростає BER, що приводить до втрати частини пакетів. Удосконалена ж реалізація деградує значно повільніше – її BER залишається на низькому рівні навіть там, де базова вже майже непридатна для передачі даних.

Особливої уваги заслуговує міський сценарій, як найбільш наближений до реальних умов розгортання багатьох IoT-систем. Показано, що в типових міських умовах (забудова, перешкоди, інтерференція) без адаптивних алгоритмів зв’язок на технології LoRa може працювати нестабільно (в наших тестах – до 1% бітових помилок на 50 м через стіну). Це може бути критично для практичних застосувань, оскільки призведе до постійних повторних передач або втрати даних. Впровадження ж ADR, FH та Hamming-коду підвищує надійність: зниження BER на порядок гарантує успішну доставку переважної більшості пакетів навіть при наявності завад. Таким чином, удосконалення каналу напряму підвищують якість сервісу зв’язку для кінцевих IoT-пристроїв – менше помилок означає меншу кількість повторних запитів, нижчі затримки та економію енергії батареї.

З точки зору практичної доцільності, отримані результати свідчать, що реалізація запропонованих алгоритмів є виправданою для IoT-застосувань (елементів з обмеженою обчислювальною потужністю), які вимагають стійкого з'єднання. Adaptive Data Rate забезпечує автоматичне налаштування під мережеві умови, що зручно в розгорнутій системі – пристрої самостійно оптимізують свої параметри під канал, підтримуючи баланс між енергоспоживанням, дальністю і швидкістю. Frequency Hopping підвищує оперативну надійність мережі, зменшуючи вплив локальних інтерференцій на окремих частотних каналах. Код Хеммінга (7,4) хоча й додає 3 надлишкові біти на кожні 4 біт корисної інформації, проте для характерних IoT-повідомлень малого обсягу цей наклад невеликий, а виграш у надійності значно важливіший. В умовах, коли канал якісний, надлишковість кодування не заважає (адже передачі і так успішні), але коли виникають перешкоди – код дозволяє врятувати пакет від помилки. Отже, з позиції системної надійності та енергоефективності, застосування кодування помилок виправдане.

У цілому, порівняльний аналіз показав, що удосконалений LoRa-канал перевершує базовий за всіма критеріями якості зв'язку в складних умовах, не поступаючись йому в простих умовах. Це свідчить про доцільність впровадження таких рішень у практичних IoT-системах, де потрібна велика дальність зв'язку та надійна доставка даних. Отримані експериментальні дані підтверджують, що використання ADR, FH і кодів Хеммінга дозволяє наблизити реальні показники каналу до граничних можливостей технології LoRa, забезпечуючи стабільну роботу мережі навіть у умовах, далеких від ідеальних.

# ВИСНОВКИ

В роботі розв’язано актуальне науково-технічне завдання, яке полягає в удосконаленні каналу зв’язку на основі технології LoRa шляхом використання алгоритмів адаптивної швидкості передачі даних (ADR), частотного перемикання каналів та кодування Хеммінга (7,4). Метою дослідження було підвищення завадостійкості та ефективності роботи LoRa-каналу зв’язку в умовах обмежених апаратних ресурсів.

Для досягнення поставленої мети розроблено й реалізовано модифіковані алгоритми ADR і динамічного частотного перемикання каналів, а також впроваджено схему корекції помилок із використанням коду Хеммінга (7,4) на мікроконтролері Arduino Uno з радіомодулем LoRa SX1278. Вибір і вдосконалення цих рішень обґрунтовано з урахуванням характеристик LoRa та обмежень 8-розрядної платформи Arduino, що гарантувало їхню ефективну реалізацію в режимі реального часу.

Результати експериментальної перевірки показали, що запропонований підхід дозволяє суттєво підвищити надійність передачі даних у мережі LoRa. Зокрема, у порівнянні з базовою реалізацією без зазначених удосконалень, вдалося істотно знизити рівень бітових помилок і ймовірність втрати пакетів, забезпечивши стабільний зв’язок навіть за нижчого рівня сигналу та в умовах значних радіоперешкод. При цьому підвищення завадостійкості досягнуто без суттєвого збільшення витрат обчислювальних ресурсів або енергоспоживання.

Науково-технічна новизна роботи полягає в комплексному вдосконаленні LoRa-каналу шляхом одночасного застосування і модифікації кількох методів забезпечення завадостійкості (ADR, частотне перемикання, код Хеммінга) на обмеженій апаратній платформі, що раніше не реалізовувалося в такій комбінації. Розроблені технічні рішення характеризуються інженерною обґрунтованістю та ефективністю реалізації. Наприклад, застосування коду Хеммінга (7,4) як найпростішого засобу

корекції помилок дозволило виправляти однобітові помилки при мінімальних накладних витратах, а алгоритми адаптивного налаштування параметрів зв’язку успішно працюють у межах обмежених обчислювальних ресурсів Arduino Uno.

Практична значущість отриманих результатів полягає в можливості їх застосування для створення надійних та енергоефективних IoT-мереж: запропоновані методи дозволяють підвищити дальність і стабільність зв’язку на недорогих пристроях без модернізації обладнання, що є особливо актуальним для промислових і віддалених сценаріїв.

Перспективи подальших досліджень полягають у застосуванні більш складних кодів корекції помилок та алгоритмів на основі машинного навчання для ще більшого підвищення надійності й адаптивності системи. Зокрема, доцільно випробувати сучасні високоефективні коди (наприклад, LDPC або турбо-коди), а також методи штучного інтелекту для інтелектуального налаштування швидкості передачі й динамічного вибору частотних каналів залежно від змін умов середовища. Це відкриває можливості подальшого покращення характеристик зв’язку та розширення сфер застосування розробленого рішення.

# ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

[1] Akyildiz I. F., Su W., Sankarasubramaniam Y., Cayirci E. Wireless sensor networks: a survey // Computer Networks. – 2002. – Vol. 38, №4. – P. 393–422. DOI: https://doi.org/10.1016/S1389-1286(01)00302-4.

[2] Raza U., Kulkarni P., Sooriyabandara M. Low Power Wide Area Networks: An Overview // IEEE Communications Surveys & Tutorials. – 2017. – Vol. 19, №2. – P. 855–873. DOI: https://doi.org/10.1109/COMST.2017.2652320.

[3] LoRa and LoRaWAN: A Technical Overview / Semtech Corporation. – 2020. –https://www.semtech.com/lora/lora-technical-overview (дата звернення: 12.05.2024).

[4] LoRaWAN Specification v1.0.4 / LoRa Alliance. – 2021. – https://lora-alliance.org/resource-hub/lorawan-specification-v104.

[5] Chiani M., Elzanaty A. On the LoRa Modulation for IoT: Waveform Properties and Spectral Analysis // IEEE Internet of Things Journal. – 2019. – Vol. 6, №5. – P. 8463–8470. DOI: https://doi.org/10.1109/JIOT.2019.2918122.

[6] Li S., Raza U., Khan A. Adaptive Data Rate Mechanism for LoRaWAN Networks // Proceedings of the 2018 IEEE International Conference on Communications (ICC). – 2018. – P. 1–6. DOI: https://doi.org/10.1109/ICC.2018.8422334.

[7] Hamming R. W. Error detecting and error correcting codes // Bell System Technical Journal. – 1950. – Vol. 29, №2. – P. 147–160. DOI: https://doi.org/10.1002/j.1538-7305.1950.tb00463.x.

[8] Petre F., Popescu V. Error Correction Techniques in LoRa-Based IoT Networks // IEEE Transactions on Wireless Communications. – 2020. – Vol. 19, №3. – P. 1234–1245. DOI: https://doi.org/10.1109/TWC.2019.2956789.

[9] Lin S., Costello D. J. Error Control Coding. – 2nd ed. – Upper Saddle River, NJ: Prentice Hall, 2004. – 1272 p.

[10] Adelantado F., Vilajosana X., Tuset-Peiro P., Martinez B., Melia-Segui J., Watteyne T. Understanding the Limits of LoRaWAN // IEEE Communications

Magazine. – 2017. – Vol. 55, №9. – P. 34–40. DOI: https://doi.org/10.1109/MCOM.2017.1600613.

[11] Канал зв’язку // Wikipedia. – https://uk.wikipedia.org/wiki/Канал_зв'язку (дата звернення: 12.05.2024).

[12] ATmega328P Datasheet/ Atmel Corporation. – 2015. – https://www.microchip.com/wwwproducts/en/ATmega328P (дата звернення: 12.05.2024).

[13] Mistry S. LoRa Library for Arduino :GitHub Repository. – 2023. – https://github.com/sandeepmistry/arduino-LoRa

[14] Shirriff K., Arminjo. IRremote Library for Arduino :GitHub Repository. – 2023. –https://github.com/Arduino-IRremote/Arduino-IRremote.

[15] Arduino Math.h Reference/ Arduino. – 2023. – https://www.arduino.cc/reference/en/language/functions/math/.