МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
"КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ІМ. ІГОРЯ СІКОРСЬКОГО "
Факультет електроніки
Кафедра електронної інженерії

До захисту допущено
Завідувач кафедри
_____ В. І. Тимофєєв
"___"_____ 20___р.

Дипломна робота

освітнього рівня «бакалавр»
за спеціальністю 153 мікро- та наносистемна техніка

на тему: Вимірювання радіочастотних характеристик ZigBee пристроїв у несигнальному режимі

Виконав студент 4 курсу, групи ДМ-62
Макарчук Сергій Іванович _____
(прізвище, ім'я, по батькові) (підпис)

Керівник проф. каф. ЕІ, доц., д.т.н. Прокопенко Ю.В. _____
(посада, вчене звання, науковий ступінь, прізвище та ініціали) (підпис)

Рецензент зав. каф. МЕ, доц., к.т.н. Орлов А.Т. _____
(посада, вчене звання, науковий ступінь, прізвище та ініціали) (підпис)

Засвідчую, що у цій дипломній роботі немає запозичень з праць інших авторів без відповідних посилань.
Студент _____
(підпис)

Київ - 2020 року

Національний технічний університет України
“Київський політехнічний інститут імені Ігоря Сікорського”

Факультет електроніки
Кафедра електронної інженерії
Освітній рівень «бакалавр»
за спеціальністю 153 мікро- та наносистемна техніка

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ В. І. Тимофєєв
“___”_____ 20___ р.

З А В Д А Н Н Я
НА ДИПЛОМНУ РОБОТУ СТУДЕНТУ

_____Макарчуку Сергію Івановичу____ __
(прізвище, ім’я, по батькові)

1. Тема роботи «Вимірювання радіочастотних характеристик ZigBee пристроїв у несигнальному режимі»

керівник роботи проф. каф. ЕІ, доц., д.т.н. Прокопенко Ю.В._____
(прізвище, ім’я, по батькові, науковий ступінь, вчене звання)
затверджені наказом по університету від “25” травня 2020 року № 1196-с
2. Строк подання студентом роботи 12 червня 2020 року.
3. Вихідні дані до роботи: специфікація IEEE 802.15.4, специфікація ZigBee, тестер MTP300A, посібник користувача MTP300A, бібліотека SCPI-команд, бібліотека керування DUT.
4. Зміст дипломної роботи (перелік питань, які потрібно розробити): розглянути основні можливості стандарту ZigBee; розглянути архітектуру стеку ZigBee; описати функції основних рівнів стеку; порівняти сигнальний та несигнальний методи тестування: розробити додаток з графічним інтерфейсом користувача для вимірювання радіочастотних характеристик пристроїв ZigBee; провести моделювання пристроїв ZigBee та виміряти їх характеристики.
5. Перелік графічного (ілюстративного) матеріалу (із зазначенням обов’язкових креслень, плакатів, презентацій тощо): рисунки в тексті до пояснювальної записки, ілюстрації для захисту.
6. Дата видачі завдання: 03 лютого 2020 року.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів виконання дипломної роботи	Строк виконання етапів роботи	Примітка
1	Огляд літератури про стандарт ZigBee	лютий – березень 2020 року	
2	Розробка програми для роботи з тестером MTP300A	лютий – березень 2020 року	
3	Моделювання пристроїв ZigBee та вимірювання їх радіочастотних характеристик	лютий – березень 2020 року	
4	Дослідження архітектури протоколу ZigBee	березень – квітень 2020 року	
5	Оформлення пояснювальної записки	травень – червень 2020 року	
7	Підготовка дипломної роботи до захисту	червень 2020 року	

Студент _____ Макарчук С.І.
(підпис) (прізвище та ініціали)

Керівник роботи _____ Прокопенко Ю.В.
(підпис) (прізвище та ініціали)

РЕФЕРАТ

Дипломна робота: 84 ст., 3 розділи, 17 рис., 3 табл. та 16 джерел.

ZIGBEE, IEEE 802.15.4, MESH-МЕРЕЖІ, НЕСИГНАЛЬНЕ ТЕСТУВАННЯ, OQPSK-МОДУЛЯЦІЯ

Об'єктом дослідження даної роботи є пристрої ZigBee. Предметом дослідження – радіочастотні характеристики цих пристроїв. Метою даною роботи є розробка несигнального методу тестування пристроїв ZigBee.

У першому розділі дипломної роботи було здійснено огляд основних особливостей мереж протоколу ZigBee. Розглянуті частотні діапазони, у яких працюють пристрої ZigBee. Проведено порівняння основних можливостей ZigBee з двома найпоширенішими безпровідними протоколами зв'язку: Bluetooth та WLAN (Wi-Fi). Описані топології мереж ZigBee та ролі пристроїв у цих мережах.

У другому розділі роботи надано короткий огляд архітектури протоколу ZigBee. Описано ролі основних рівнів стеку протоколу (фізичного рівня PHY, рівня доступу до середовища MAC, мережевого рівня NWK та прикладного рівня APL). Показано як узгоджена взаємодія цих рівнів (модуляція та демодуляція радіосигналів, взаємодія двох окремо взятих приймача та передавача, маршрутизація повідомлень) забезпечує повноцінне функціонування мережі ZigBee.

У третьому розділі наведено результати моделювання роботи передавача ZigBee у частотному діапазоні 2,4 ГГц та вимірювання залежностей, що використовуються при несигнальному тестуванні: спектральної характеристики, часової залежності потужності сигналу та сигнального сузір'я. Описані переваги та недоліки несигнального тестування в порівнянні з сигнальним режимом. Розглянутий підхід до дискретизації сигналів за допомогою квадратурних I/Q складових. Обчислено основний показник якості модуляції сигналу – величину вектора похибок (EVM).

ABSTRACT

Diploma work: 84 p., 3 sections, 17 figures, 3 tables and 16 references.

ZIGBEE, IEEE 802.15.4, MESH NETWORKS, NON-SIGNALING TESTING, OQPSK MODULATION

The object of the study are ZigBee devices. The subject are radio characteristics of these devices. The goal of this work is a development of ZigBee devices non-signaling testing method.

The first section of the diploma work reviews the main features of ZigBee protocol networks. ZigBee devices operating frequency bands was considered. The main possibilities of ZigBee were compared with the two most common wireless communication protocols: Bluetooth and WLAN (Wi-Fi). The topologies of ZigBee networks and the roles of devices in these networks were described.

The second sections provides brief review of ZigBee protocol architecture. Roles of main protocol stack layers (PHY layer, MAC layer, NWK layer, APL layer) were described. It is shown how the coordinated interaction of these levels (modulation and demodulation of radio signals, interaction of two separate receiver and transmitter, routing of messages) ensures the full functioning of the ZigBee network.

The third section describes results of the ZigBee transmitter simulation in the 2.4 GHz frequency band and measurements of the dependences used in non-signaling testing: spectral characteristics, time dependence of signal power and signal constellation plot. The advantages and disadvantages of non-signaling testing were compared with the signaling mode. The approach to signal sampling using quadrature I/Q components was considered. The OQPSK-modulated signal was recovered from the measured I/Q samples. The main indicator of signal modulation quality is calculated - the error vector magnitude (EVM).

ЗМІСТ

СКОРОЧЕННЯ ТА УМОВНІ ПОЗНАКИ

APL – Application Layer (прикладний рівень)

BER – Bit Error Rate (коефіцієнт бітових помилок)

BPSK – Binary Phase Shift Keying (двопозиційна фазова маніпуляція)

DSSS – Direct Sequence Spread Spectrum (розширення спектру методом прямої послідовності)

DUT – Device under test (пристрій, що тестується)

ED – Energy Detection (визначення рівня сигналу)

EPID – Extended PAN ID (розширений ідентифікатор PAN)

EVM – Error Vector Magnitude (величина вектора похибки)

FFD – Full-Function Device (повнофункціональний пристрій)

HR-WPAN – high-rate WPAN (високошвидкісна WPAN)

LQI – Link Quality Indicator (показник якості зв'язку)

LR-WPAN – low-rate WPAN (низькошвидкісна WPAN)

LSB – найменш значущий біт

MAC – Media Access Control Layer (рівень керування доступом до середовища)

MR-WPAN – medium-rate WPAN (середньошвидкісна WPAN)

NWK – Network layer (мережевий рівень)

O-QPSK – Offset Quadrature Phase Shift Keying (чотирьохпозиційна фазова маніпуляція)

PAN – Personal Area Network (персональна локальна мережа)

PAN ID – PAN Inditificator (ідентифікатор PAN)

PER – Packet Error Rate (коефіцієнт помилок пакетів)

PHR – PHY Header (заголовок PHY)

PHY – Physical Layer (фізичний рівень)

PIB – PAN Information Base

PPDU – PHY Protocol Data Unit (блок даних протоколу PHY)

RFD – Reduced-Function Device (пристрій з обмеженою функціональністю)

RSSI – Receive Signal Strength Indication

SFD – Synchronization Delimiter (роздільник початку фрейму)

SHR – Synchronization Header (заголовок синхронізації)

WLAN – Wireless Local Area Network (безпровідна локальна мережа)

WPAN – Wireless Personal Area Network (безпровідна персональна мережа)

ZC – ZigBee Coordinator (координатор ZigBee)

ZCL – ZigBee Cluster Library (бібліотека кластерів ZigBee)

ZDO – ZigBee Device Object (об’єкт пристрою ZigBee)

ZDP – ZigBee Device Profile (профіль пристроїв ZigBee)

ZED – ZigBee End Device (кінцевий пристрій ZigBee)

ZR – ZigBee Router (маршрутизатор ZigBee)

ВСТУП

З кожним роком розвиток безпровідних комунікацій відбувається все активніше. Найбільш яскраво це демонструється переходом до 5G покоління мобільного зв'язку [1]. Така трансформація обумовлена перш за все невпинним ростом трафіка мереж операторів мобільного зв'язку та неможливістю покрити витрати, що виникають унаслідок цього, за рахунок традиційних послуг. При чому основний ріст трафіка відбувається у секторі IoT, а не у секторі людей.

Мережі 5G вирішують перелічені проблеми за рахунок підвищення швидкості передачі даних та значного розширення можливостей. Фактично 5G підтримує усі послуги пов'язані з "інтернетом речей": починаючи від засобів промислової автоматизації до "розумного міста".

Відомою серед такого роду концепцій є ідея "розумного будинку". Одним з виробників пристроїв, що дозволяють дистанційно керувати своїм домом є китайська компанія Xiaomi, яка широким масам відома своїми смартфонами. У комплекти їх датчиків входять сенсори руху, відкриття вікон та дверей, радіо, бездротові комутатори тощо [2]. Окрім відомого усім протоколу Wi-Fi, який забезпечує безпровідний доступ до мережі Internet, пристрої Xiaomi підтримують стандарт ZigBee.

ZigBee використовується для мереж, у яких енергоспоживання є важливим параметром, але висока швидкість передачі даних не є необхідною. Саме такими і є безпровідні сенсорні мережі, такі як мережі пристроїв "розумного дому". Також ZigBee забезпечує криптографічний захист даних та можливість створення самовідновлюваних мереж і, як наслідок, високу надійність.

Проте застосування протоколу ZigBee не обмежується контролем за побутовими приладами. Безпровідні сенсорні мережі ZigBee використовуються для автоматизації великих будівель, керуванням виробничими потужностями, моніторингу стану пацієнтів… Даний стандарт можна використовувати для V2X

комунікації (англ. Vehicle-to-everything), що застосовується у сфері automotive, зокрема, для зменшення кількості проводів та кабелів в автомобільних системах.

Разом з розвитком сфери безпровідної комунікації стає актуальною і тема тестування цієї безпровідних пристроїв. Виробники зацікавлені у прискоренні тестування без втрати його якості, оскільки це зменшує собівартість кінцевого продукту. У цьому, зокрема, допомагає несигнальний метод тестування, що не зобов'язує підтримувати повноцінне протокольне з'єднання і, як наслідок, зменшує тривалість тестування.

Об'єктом дослідження даної роботи є пристрої ZigBee. Предметом дослідження – радіочастотні характеристики цих пристроїв.

Метою даною роботи є розробка несигнального методу тестування пристроїв ZigBee. Для досягнення поставленої мети буде вирішено наступні завдання:

1. Дослідити основні можливості стандарту ZigBee.
2. Вивчити архітектуру стеку ZigBee.
3. Проаналізувати функції основних рівнів стеку та яким чином їх взаємодія забезпечує ефективну комунікацію пристроїв у мережі ZigBee.
4. Порівняти сигнальний та несигнальний методи тестування.
5. Розробити додаток з графічним інтерфейсом користувача для вимірювання радіочастотних характеристик пристроїв ZigBee за допомогою тестера MTP300A.
6. Провести моделювання пристроїв ZigBee за допомогою тестера MTP300A та виміряти його характеристики (спектр сигналу, часову залежність потужності сигналу, величину вектора похибки EVM).

1 ОГЛЯД СТАНДАРТУ ZIGBEE

1.1 Загальна характеристика та застосування

ZigBee – це стандарт, який визначає набір протоколів зв'язку для бездротових мереж з низькою швидкістю передачі даних (англ. Low rate wireless personal area network, LR-WPAN). Також стандарт ZigBee спеціально розроблений для того, щоб задовольнити потребу в дуже низькій вартості впровадження мереж WPAN (англ. Wireless personal area network) та із наднизьким споживанням енергії. Бездротові пристрої на основі ZigBee працюють у діапазонах частот 868 МГц, 915 МГц та 2,4 ГГц. Максимальна швидкість передачі даних – 250 кбіт/с [4]. ZigBee орієнтований головним чином на пристрої, що мають акумуляторне живлення, для яких основні вимоги – низька швидкість передачі даних, низька вартість та тривалий час роботи акумулятора. У багатьох застосуваннях ZigBee загальний час, коли пристрій з бездротовим зв'язком зайнятий виконанням своїх основних функцій, дуже обмежений; пристрій проводить більшу частину свого часу в сплячому режимі (тобто у режимі енергозбереження). Завдяки цьому пристрої ZigBee здатні працювати кілька років, перш ніж їх батареї потрібно буде замінити.

Стандарт ZigBee розроблений Альянсом ZigBee [3], який налічує сотні компаній-членів, від напівпровідникової індустрії та розробників програмного забезпечення до виробників обладнання. Альянс ZigBee був утворений у 2002 році як некомерційна організація, відкрита для всіх, хто хоче приєднатися. Фізичний рівень (англ. Physical layer, PHY) та рівень керування доступом до середовища (англ. Media access control, MAC) ZigBee керуються стандартом IEEE 802.15.4.

Одне із застосувань ZigBee – це спостереження за пацієнтами вдома [4]. Наприклад, кров'яний тиск і серцебиття пацієнта можна виміряти за допомогою переносних пристроїв. Пацієнт носить пристрій ZigBee, який взаємодіє з датчиком, який періодично збирає показники стану здоров'я, наприклад, артеріальний тиск. Потім дані бездротовим шляхом передаються на локальний сервер, наприклад, персональний комп’ютер всередині будинку пацієнта, де проводиться попередня

обробка інформації. Нарешті, дані надсилаються до медсестри або лікаря пацієнта через Інтернет для подальшого аналізу.

Ще один приклад використання приладів, що працюють за стандартом ZigBee – моніторинг стану конструкцій великих будівель [5]. У цьому застосуванні декілька бездротових датчиків із підтримкою ZigBee (наприклад, акселерометри) можуть бути встановлені в будівлі, і всі ці датчики можуть утворювати єдину бездротову мережу для збору інформації, яка буде використана для оцінки стану конструкції та виявлення ознак будівлі можливих пошкоджень. Наприклад, після землетрусу, будівлі може знадобитися перевірка, перш ніж туди зможуть повернутися люди. Дані, зібрані датчиками, можуть допомогти прискорити та зменшити витрати на перевірку.

Стандарт ZigBee допомагає знизити вартість реалізації мережі шляхом спрощення протоколів зв'язку та зменшення швидкості передачі даних. Мінімальні вимоги згідно специфікаціям ZigBee та IEEE 802.15.4 є відносно прості у дотриманні порівняно з іншими стандартами, такими як IEEE 802.11, що зменшує складність та витрати на реалізацію сумісних з ZigBee приймачів.

1.2 Класи бездротових мереж короткого діапазону

Бездротових мережі короткого діапазону поділяються на дві основні категорії: бездротові локальні мережі (англ. Wireless Local Area Network, WLAN) та бездротові персональні мережі (WPAN) [5].

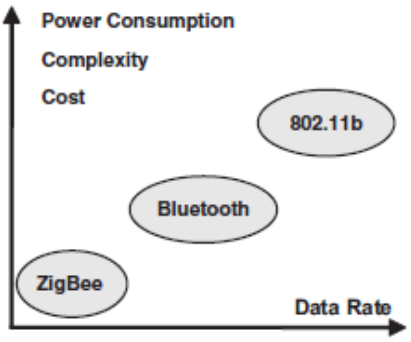
WLAN – це заміна або розширення для провідних локальних мереж (LAN), таких як Ethernet (IEEE 802.3). Пристрій WLAN може бути інтегровано до дротової мережі LAN, і як тільки пристрій WLAN стане частиною мережі, мережа обробляє бездротовий пристрій так само, як і будь-який інший дротовий пристрій у мережі. Мета WLAN - максимізувати діапазон та швидкість передачі даних.

Навпаки, WPAN не розроблені для заміни існуючих дротових локальних мереж. WPAN створені для забезпечення засобів для енергоефективного бездротового зв'язку в особистому операційному просторі (POS) без необхідності будь-якої інфраструктури.

WPAN поділяються на три класи: високошвидкісні (HR) WPAN, середньої швидкості (MR) WPAN та низькошвидкісні (LR) WPAN. Прикладом HR-WPAN є IEEE 802.15.3 зі швидкістю передачі даних від 11 до 55 Мбіт/с. Така висока швидкість передачі даних допомагає, наприклад, при бездротовій передачі відео в режимі реального часу від камери до телевізора, що знаходиться поблизу. Bluetooth, який забезпечує швидкість передачі даних від 1 до 3 Мбіт/с, є прикладом MR-WPAN і може використовуватися для передачі голосу високої якості в бездротових гарнітурах.

1.3 Порівняння ZigBee з Bluetooth та IEEE 802.11

Розглянемо найбільш поширені стандарти бездротових мереж короткого діапазону, включаючи стандарт бездротових локальних мереж IEEE 802.11 WLAN (також відомий як Wi-Fi) та Bluetooth. Кожен із цих стандартів має свої переваги, зокрема в застосуванні. На рисунку 1.1 узагальнено основні характеристики цих трьох стандартів [5].



	Data Rate	Typical Range	Application Examples
ZigBee	20 to 250 Kbps	10–100 m	Wireless Sensor Networks
Bluetooth	1 to 3 Mbps	2–10 m	Wireless Headset Wireless Mouse
IEEE 802.11b	1 to 11 Mbps	30–100 m	Wireless Internet Connection

Рисунок 1.1 – Порівняння ZigBee, Bluetooth та IEEE 802.11b

IEEE 802.11 – сімейство стандартів. Для порівняння вибрано IEEE 802.11b, оскільки саме цей стандарт працює в діапазоні 2,4 ГГц, що є спільним для Bluetooth та ZigBee. IEEE 802.11b має високу швидкість передачі даних (до 11 Мбіт/с), а забезпечення безпровідного підключення до мережі Інтернет є одним із його типових застосувань. Дальність IEEE 802.11b у приміщенні зазвичай становить від 30 до 100 метрів. З іншого боку, Bluetooth має нижчу швидкість передачі даних (менше 3 Мб/с), і його внутрішній діапазон, як правило, становить від двох до десяти метрів.

Одне з популярних застосування Bluetooth – бездротові гарнітури та навушники. ZigBee має найнижчу швидкість передачі даних і складність серед цих трьох стандартів і забезпечує значно довший термін служби акумулятора.

Дуже низька швидкість передачі даних ZigBee означає, що це не найкращий вибір для здійснення бездротового підключення до Інтернету або безпровідної гарнітури, де бажана швидкість передачі більше 1 Мбіт/с. Однак якщо метою бездротового зв’язку є передача та отримання простих команд і збирання інформації отримуваної від датчиків, таких як датчики температури чи вологості, ZigBee забезпечує найбільшу потужність та найвигідніше рішення у порівнянні з Bluetooth та IEEE 802.11b.

1.4 Зв'язок між стандартами ZigBee та IEEE 802.15.4

Одним із поширених способів налагодження мережі зв'язку (дротової чи бездротової) є використання концепції рівнів мережі. Кожен рівень відповідає за певні функції в мережі. Рівні зазвичай передають дані та команди тільки рівням безпосередньо над та під ними.

Рівні протоколів ZigBee засновані на базовій еталонній моделі OSI (англ. Open System Interconnect). Поділ протоколу на рівні має ряд переваг. Наприклад, якщо протокол змінюється з часом, легше замінити або змінити рівень, на який

впливає зміна, а не замінити весь протокол. Також при розробці програми нижчі рівні протоколу не залежать від програми і можуть бути отримані від третіх сторін, тому все, що потрібно зробити – це внести зміни в прикладний рівень протоколу.

Два нижні мережеві рівні визначаються стандартом IEEE 802.15.4. Цей стандарт розроблений комітетом зі стандартів IEEE 802 і був спочатку випущений у 2003 році. IEEE 802.15.4 визначає специфікації для рівнів PHY та MAC бездротових мереж, але він не визначає жодних вимог до вищих рівнів.

Стандарт ZigBee визначає лише мережевий (англ. Network layer, NWK), прикладний та захисний рівні протоколу та приймає рівні IEEE 802.15.4 PHY та MAC як частину мережевого протоколу ZigBee. Тому будь-який пристрій, сумісний із ZigBee, також відповідає IEEE 802.15.4.

IEEE 802.15.4 був розроблений незалежно від стандарту ZigBee, і на ньому можна будувати бездротові мережі LR-WPAN, а не тільки реалізовувати специфічні для ZigBee рівні. У цьому випадку користувачі розробляють власний протокол мережевого та прикладного рівня поверх IEEE 802.15.4 PHY та MAC. Ці власні рівні зазвичай простіші, ніж рівні протоколів ZigBee, і орієнтовані на конкретні застосування.

Однією з переваг користувацьких мережевого/прикладного рівнів є менший розмір пам’яті, необхідний для реалізації протоколу, що може призвести до зниження витрат. Однак реалізація повного протоколу ZigBee забезпечує сумісність з бездротовими рішеннями інших постачальників та додаткову надійність завдяки можливостям mesh-мереж, підтримуваних ZigBee. Рішення про те, застосовувати весь протокол ZigBee чи просто IEEE 802.15.4 PHY та MAC рівні, залежить від особливостей застосування.

Характеристики фізичного рівня мережі визначаються специфікацією рівня PHY; отже, такі параметри, як частота роботи, швидкість передачі даних, вимоги щодо чутливості приймача та типи пристроїв, визначені у стандарті IEEE 802.15.4.

1.5 Частота роботи та швидкість передачі даних

Протокол IEEE 802.15.4 підтримує три діапазони частот [6]:

– 868–868,6 МГц (діапазон 868 МГц)

– 902–928 МГц (діапазон 915 МГц)

– 2400–2483,5 МГц (діапазон 2,4 ГГц)

Діапазон 868 МГц використовується в Європі для багатьох застосувань, включаючи бездротові мережі короткого діапазону. Інші два діапазони (915 МГц і 2,4 ГГц) є частиною ISM-діапазону (від англ. industrial, scientific and medical). Діапазон частот 915 МГц використовується в основному в Північній Америці, тоді як діапазон 2,4 ГГц використовується в усьому світі.

У специфікації зазначається, що якщо приймач підтримує діапазон 868 МГц, то він також повинен підтримувати смугу 915 МГц, і навпаки. Тому ці два діапазони завжди поєднуються у діапазон частот 868/915 МГц.

Передавач 2,4 ГГц може підтримувати діапазони 868/915 МГц, але IEEE 802.15.4 цього не вимагає. У діапазоні 868 МГц є лише один канал. Діапазон 915 МГц має 10 каналів (без врахування додаткових каналів). У діапазоні 2,4 ГГц міститься загалом 16 каналів.

Діапазон ISM 2,4 ГГц прийнятий у всьому світі та має максимальну швидкість передачі даних та кількість каналів. З цієї причини розробка приймачів для діапазону 2,4 ГГц є популярним вибором для багатьох виробників. Однак IEEE 802.11b працює у цій же смузі частот, і співіснування може бути проблемою в деяких застосування. Також, чим нижча смуга частот, тим краще сигнали проникають через стіни та різні об'єкти. Тому деякі користувачі можуть вважати діапазон 868/915 МГц кращим вибором для своїх програм.

Протокол ZigBee використовує два типи модуляції визначені IEEE 802.15.4, а саме: двопозиційна фазова маніпуляція (англ., Binary phase shift keying, BPSK), та чотирьохпозиційна фазова маніпуляція зі зсувом квадратур (O-QPSK).

1.6 Сумісність пристроїв

ZigBee має широкий спектр застосувань; тому різні виробники надають рішення, засновані на використанні ZigBee. Важливо, щоб ці пристрої мали можливість взаємодіяти один з одним незалежно від виробника. Іншими словами, пристрої повинні бути сумісними. Саме сумісність одна з ключових переваг стеку протоколів ZigBee. Пристрої на основі ZigBee є сумісними навіть тоді, коли повідомлення шифруються з міркувань безпеки [4].

1.7 Типи пристроїв

У бездротовій мережі IEEE 802.15.4 є два типи пристроїв: пристрої з повною функціональністю (англ. Full-Function Device, FFD) та пристрої зі зниженою функціональністю (англ. Reduced-Function Device, RDF) [7].

FFD здатний виконувати всі обов'язки, описані в стандарті IEEE 802.15.4, і може приймати будь-яку роль у мережі. RFD, з іншого боку, має обмежені можливості. Наприклад, FFD може спілкуватися з будь-яким іншим пристроєм у мережі, але RFD може спілкуватися лише з пристроєм FFD. Пристрої RFD призначені для дуже простих застосувань, таких як вмикання або вимикання комутатора. Потужність обробки та об'єм пам'яті пристроїв RFD, як правило, менше, ніж у пристроїв FFD.

1.8 Ролі пристроїв згідно з IEEE 802.15.4 та ZigBee

Згідно специфікації IEEE 802.15.4 пристрій FFD може виконувати три різні ролі: координатор, координатор PAN та пристрій [6].

Координатор – це пристрій FFD, здатний ретранслювати повідомлення. Якщо координатор є також головним контролером персональної локальної мережі (англ. Personal area network, PAN), він називається координатором PAN. Якщо пристрій не виконує функції координатора, його просто називають пристроєм.

Стандарт ZigBee використовує дещо іншу термінологію. Для позначення ролей вводяться три типи пристроїв [4]:

– координатор (ZigBee coordinator, ZC);

– маршрутизатор (ZigBee router, ZR);

– кінцевий пристрій (ZigBee end device, ZED).

Координатор ZigBee по суті являється координатором IEEE 802.15.4 PAN, він створює та керує мережею. Також координатор є довірчим центром мережі (зберігає ключі безпеки).

Маршрутизатор ZigBee – це пристрій, який може виконувати функції координатора IEEE 802.15.4. Цей тип пристроїв відповідальний за ретрансляцію повідомлень до інших пристроїв мережі.

Кінцевий пристрій ZigBee – це пристрій, який не є ні координатором, ні маршрутизатором. Кінцевий пристрій ZigBee має найменший об'єм пам'яті та найменше можливостей та можливостей обробки. Кінцевий пристрій, як правило, є найменш дорогим пристроєм у мережі.

Більш детально ролі пристроїв ZigBee описані у розділі 2 (пункт 2.2.1).

1.9 Мережеві топології ZigBee

Мережі ZigBee можуть мати одну з трьох різних топологій, вибір якої впливає на те, як буде відбуватися процес маршрутизації повідомлень та які пристрої спілкуються з якими іншими пристроями. ZigBee підтримує наступні топології мереж [4]:

– топологія "зірка";

– топологія "дерево";

– mesh-топологія.

Схематичне зображення перечислених топологій зображено на рисунку 1.2.

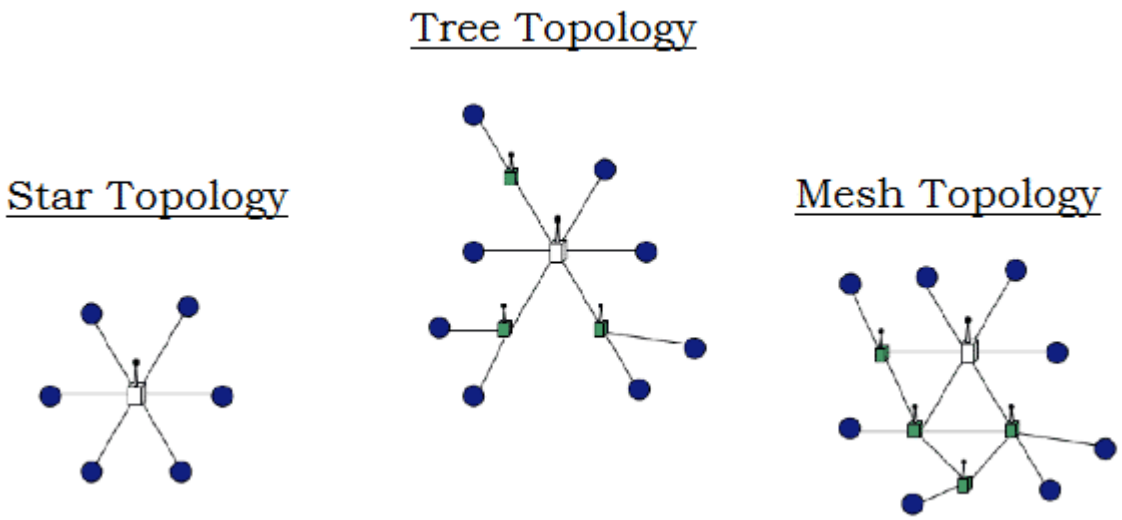


Рисунок 1.2 – Мережеві топології ZigBee [8]

Топологія "зірка" – найпростіша і найбільш обмежена топологія, доступна пристроям ZigBee. Усі пристрої підключаються до одного вузла (координатора), і будь-який зв'язок відбувається через нього. Варто зазначити, що ця топологія насправді визначається специфікацією IEEE 802.15.4, на якій будується ZigBee.

Дочірні вузли також можуть бути маршрутизаторами, хоча в цій топології вони не виконують жодної маршрутизації і по суті діють як кінцевий пристрій.

При використанні топології "зірка" пропускна здатність мережі обмежена можливостями координатора, і якщо координатор виходить з ладу, то і вся мережа виходить з ладу. Просторовий діапазон мережі також обмежений дальністю зв'язку самого координатора.

У топології "дерева" координатор формує кореневий вузол дерева дочірніх вузлів. Проміжними вузлами є маршрутизатори, а кінцевими – кінцеві пристрої (хоча маршрутизатор також може бути кінцевим вузлом, якщо до нього ще не приєдналися). Пряма комунікація може відбуватися лише між дочірнім вузлом та батьківським, але всі вузли можуть спілкуватися один з одним шляхом, що проходить по дереву до загального "предка", а потім вниз до цільового вузла.

У цій топології маршрутизатори можуть розширити діапазон мережі на відстані більші чим дальність одного пристрою. Однак, якщо маршрутизатор виходить з ладу, альтернативного маршруту немає, і частина мережі неодмінно відключиться.

Топологія mesh – одна з найбільш гнучких топологій ZigBee. Вона схожа з топологією "дерева", але без дотримання її жорсткої структури. Головною відмінністю від "дерева" є те, що маршрутизатор може спілкуватися безпосередньо з будь-яким іншим маршрутизатором або координатором, якщо він знаходиться в межах досяжності. Це означає, що може бути багато різних маршрутів через мережу до заданого вузла, і ZigBee має функцію виявлення маршруту для пошуку найкращого маршруту до даного вузла, і тому мережі ZigBee є самовідновлюваними.

1.10 Висновки до розділу 1

Було розглянуто загальні особливості протоколу ZigBee та його застосування.

З'ясовано, що мережі ZigBee відносять до класу LR-WPAN, які призначені для енергоефективного бездротового зв’язку з низькими швидкостями передачі даних чим відрізняється від поширений стандартів Bluetooth та IEEE 802.11b.

Стандарт ZigBee визначає три типи пристроїв: координатор, маршрутизатор та кінцевий пристрій.

Також особливістю ZigBee є можливість, окрім типових топологій типу "зірка" та "дерево", реалізовувати mesh-мережі, які і можуть забезпечити гнучкість мережі та самовідновлюваність.

Яким чином відбувається з'єднання та комунікація пристроїв у мережі буде розглянута у наступному розділі.

2 СТЕК ПРОТОКОЛУ ZIGBEE

2.1 Огляд архітектури

Архітектура стека ZigBee складається з набору блоків (див. рис. 2.1), які називають рівнями. Кожен рівень виконує певний набір послуг для рівня вище. Суб'єкт даних надає послугу передачі даних, а суб'єкт управління надає всі інші послуги. Кожна організація обслуговування виставляє інтерфейс до верхнього рівня через службову точку доступу (англ. Service access point, SAP), і кожен SAP підтримує ряд службових примітив для досягнення необхідної функціональності.

Стандарт IEEE 802.15.4 визначає два нижні рівні: фізичний (PHY) рівень і підрівень управління доступом до середовища (MAC). Альянс ZigBee будується на основі цих двох рівнів, забезпечуючи мережевий рівень (NWK) та фреймворк для прикладного рівня. Фреймворк прикладного рівня складається з підрівня підтримки програм (APS) та об'єктів пристрою ZigBee (ZDO). Об'єкти програм, визначені виробником, використовують цей фреймворк та ділять послуги APS та служби безпеки із ZDO.

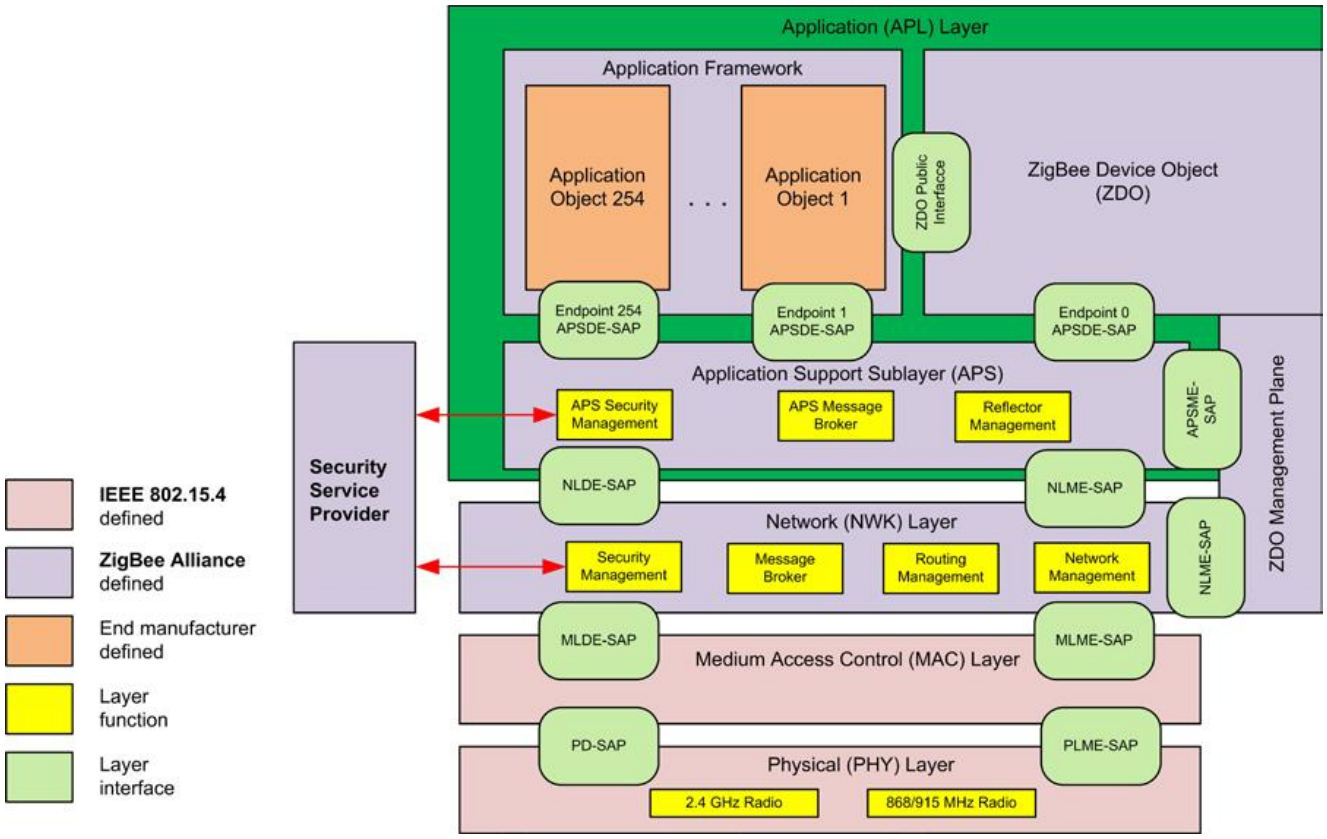


Рисунок 2.1 – Архітектура стеку ZigBee та IEEE 802.15.4 [4]

2.2 Фізичний рівень

IEEE 802.15.4 визначає функції рівня PHY та його взаємодію зі рівнем MAC. Рівень PHY є найближчим до апаратного забезпечення і безпосередньо керує та зв'язується з радіопередавачем.

Фізичний рівень відповідає за наступні функції:

– активація та деактивація радіопередавача;

– передача та отримання даних;

– вибір частоти каналу;

– виконання процедури виявлення енергії (англ. Energy Detection, ED);

– виконання процедури оцінки чистоти каналу (англ. Clear Channel Assessment, CCA).

– визначення показника якості зв'язку (англ. Link Quality Indicator, LQI).

Згідно останньої версії специфікації будь-який пристрій ZigBee повинен підтримувати, щонайменше один з наступних типів PHY:

– PHY у діапазоні 868/915 МГц з модуляцією BPSK;

– PHY у діапазоні 2,4 ГГц з модуляцією OQPSK.

2.2.1 Частотні діапазони

Згідно специфікації IEEE 802.15.4 канал описується сукупністю номеру каналу та сторінки каналу [6]. Сторінка каналу – це концепція, додана до стандарту, щоб розрізняти підтримувані типи PHY. Для PHY підтримуваних протоколом ZigBee використовуються сторінки 0–2. Сторінка під номером 0 об'єднує в собі 27 каналів, тоді як інші дві – лише 11 каналів.

На усіх трьох сторінках каналу з номером k = 0 відповідає діапазон 868 МГц з центральною частотою F_c = 868,3 МГц, а каналам з номерами k = 1, 2, …, 10 діапазон 915 МГц з центральною частотою F_c, (МГц) що обчислюється за формулою:

$$F_c = 906 + 2\,(k - 1); \tag{2.1}$$

Каналам з номерами k = 11, 12, …, 26 сторінки під номером 0 відповідає діапазон 2,4 ГГц. Центральні частоти F_c (МГц) цих каналів обчислюються за наступною формулою:

$$F_c = 2405 + 5\,(k - 11). \tag{2.2}$$

2.2.2 Формат пакету PHY

Формат пакету, який у специфікації називаються блоком даних протоколу PHY (англ. PHY Protocol Data Unit, PPDU) показаний на рисунку 2.2. PPDU складається з трьох компонентів: заголовка синхронізації (SHR), заголовка PHY (PHR) та корисного навантаження PHY [6].

		Octets		
		1		variable
Preamble	SFD	Frame length (7 bits)	Reserved (1 bit)	PSDU
SHR		PHR		PHY payload

Рисунок 2.2 – Структура PPDU

SHR дозволяє приймачу синхронізуватись у потоці бітів. PHR містить інформацію про довжину фрейму. Корисне навантаження PHY передається від рівня MAC і містить в собі дані, які потрібно необхідно іншому пристрою.

SHR складається з преамбули та роздільника початку фрейму (SFD). Поле преамбули використовується приймачем у процесі часової синхронізації. Біти в полі преамбули у всіх як і OQPSK PHY так і BPSK PHY є двійковими нулями. Довжина преамбули становить 32 біти (4 октети).

Поле SFD вказує кінець заголовка синхронізації і початок заголовка PHY. SFD має довжину 8 біт, сама послідовність наведена у таблиці 2.1.

Таблиця 2.1 – Структура поля заголовку синхронізації SFD

Bits: 0	1	2	3	4	5	6	7
1	1	1	0	0	1	0	1

Наступне поле в пакеті PHY – це довжина фрейму, яка визначає загальну кількість октетів у корисному навантаженні PHY (англ. PHY Service Data Unit, PSDU). Довжина PSDU може бути будь-яким значенням від 0 до 127 октетів.

Останнє поле – це блок даних служби PHY (PSDU). Вміст PSDU надається рівнем MAC. У IEEE 802.15.4 перший біт, який буде переданий – найменш значущий біт (LSB) SHR. Найбільш значущий біт (MSB) останнього октету корисного навантаження PHY передається останнім.

2.2.3 OQPSK модуляція

OQPSK PHY використовує наступну техніку модуляції сигналу (див. рис. 2.3). Кожні чотири інформаційні біти використовуються для вибору однієї з шістнадцяти майже ортогональних послідовностей псевдовипадкових шумів, які підлягають передачі. При чому вибір послідовностей є двоетапним процесом. Спочатку виконується перетворення чотирьох бітів у число, яке специфікація називає терміном "символ" (bit-to-symbol mapping) [6]. Потім кожному символу ставиться у відповідність послідовність так званих "чіпів" (symbol-to-chip mapping). Відповідність символів послідовностям представлена у таблиці 2.2.

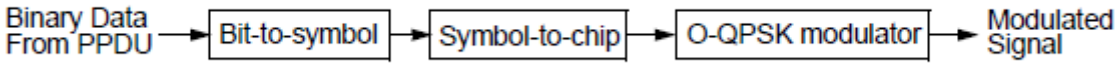


Рисунок 2.3 – Блок-схема кодування та модуляції OQPSK PHY [6]

Таблиця 2.2 – Відповідність бітів та послідовностей чіпів OQPSK PHY

Data symbol	Chip values (c_0 c_1 … c_{30} c_{31})
0	11011001110000110101001000101110
1	11101101100111000011010100100010
2	00101110110110011100001101010010
3	00100010111011011001110000110101
4	01010010000101110110110011100011
5	00110101001000101110110110011100
6	11000011010100100010111011011001
7	10011100001101010010001011101101
8	10001100100101100000011101111011
9	10111000110010010110000001110111
10	01111011100011001001011000000111
11	01110111101110001100100101100000
12	00000111011110111000110010010110
13	01100000011101111011100011001001
14	10010110000001110111101110001100
15	11001001011000000111011110111000

Утворена послідовність модулюється за допомогою чотирьохпозиційної фазової маніпуляції зі зсувом квадратур (OQPSK). У процесі модуляції парні чіпи утворюють сигнал I-складової, тоді як непарні Q-складової (детальніше про квадратурні складові можна дізнатися з розділу 3 даної роботи). Далі відбувається затримка чіпів Q-складової на половину періоду та кожному чіпу надається форма половинного синуса (англ. half-sine pulse), що описується наступною функцією:

$$p(t) = \begin{cases} \sin\left(\pi \frac{t}{2T_c}\right), 0 \le t \le 2T_c \\ 0\,, \text{ у іншому випадку} \end{cases}. \quad (2.3)$$

У результаті часова залежність квадратурних складових виглядає так, як представлено на рисунку 2.4.

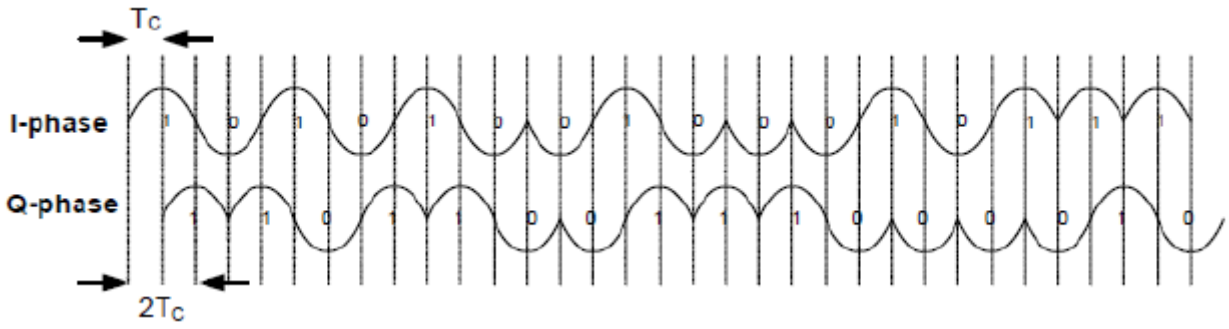


Рисунок 2.4 – Часова залежність квадратурних складових OQPSK сигналу

2.2.4 BPSK модуляція

BPSK PHY використовує розширення спектру методом прямої послідовності (англ. Direct Sequence Spread Spectrum, DSSS), двопозиційну фазову маніпуляцію BPSK, що використовується для модуляції чіпів і диференційне кодування для кодування символів даних [6].

Процес ілюструє блок-схема зображення на рисунку 2.5.

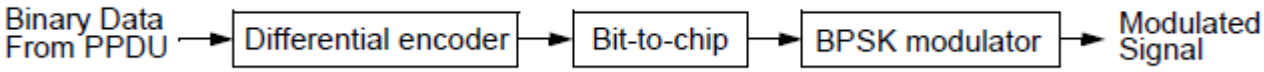


Рисунок 2.5 – Блок-схема кодування та модуляції BPSK PHY [6]

Диференціальне кодування являє собою операцію виключного АБО над поточним бітом даних R_n та попередньо закодованим бітом E_{n-1}. Закодований біт описується формулою $E_n = R_n \oplus E_{n-1}$.

У процесі розширення спектру кожному закодованому біту ставиться у відповідність послідовність псевдовипадкового шуму (див. табл. 2.3).

Таблиця 2.3 – Відповідність бітів та послідовностей чіпів BPSK PHY

Input bits	Chip values (c_0 c_1 … c_{14})
0	1 1 1 1 0 1 0 1 1 0 0 1 0 0 0
1	0 0 0 0 1 0 1 0 0 1 1 0 1 1 1

Отримана послідовність чіпів складає інформаційний сигнал, що модулюється на сигнал носій. Кожному чіпу надається форма, що описується наступною функцією (в англомовній літературі до неї застосовується термін "raised cosine pulse"):

$$p(t) = \begin{cases} \frac{\sin \pi t/T_c}{\pi t/T_c} \frac{\cos \pi t/T_c}{1-4t^2/{T_c}^2}, t \neq 0 \\ 1, t = 0 \end{cases}, \quad (2.4)$$

де t – час;

T_c – напівперіод чіпа.

2.2.5 Виявлення енергії

Коли пристрій планує передати повідомлення, він спочатку переходить у режим прийому для виявлення та оцінки рівня енергії сигналу в потрібному каналі. Ця завдання відоме як виявлення енергії (англ. Energy Detection, ED). Енергія сигналу у використовуваному частотному діапазоні усереднюється протягом часу, що дорівнює восьми символьним періодам. У процесі виявлення енергії приймач не намагається виявити тип сигналу чи декодувати його [6].

Запит на отримання рівня енергії (значення ED) виконує MAC у PHY рівня. Діапазон прийнятої потужності, що охоплюється значеннями ED, повинен бути не менше 40 дБ. The ED result shall be reported to the next higher layer using MLME-

SCAN.confirm. The minimum ED value (zero) shall indicate received power less than 10 dB above the lowest specified receiver sensitivity, in dBm, for the PHY. У цьому діапазоні відображення від прийнятої потужності в децибелах до значення ED повинно бути лінійним з точністю ± 6 дБ.

2.2.6 Показник якості зв'язку

Індикатор якості зв'язку LQI – це показник якості отриманих пакетів даних. Вимірювання LQI можуть виконуватися використовуючи як процедуру ED так і визначення відношення сигнал-шум (англ. Signal to Noise Ratio, SNR), або ж відразу обидва способи. Значення LQI лежить в межах від 0x00 до 0xff та обов'язково має використовувати щонайменше 8 унікальних рівнів [6].

Вимірювання LQI виконується для кожного прийнятого пакету. PHY повідомляє MAC-рівню значення LQI, з якого рівні NWK та APL можуть використовувати його для маршрутизації мережі. Наприклад, рівень NWK може використовувати повідомлені MAC-рівнем LQI пристроїв у мережі, щоб визначити, який шлях використовувати для передачі повідомлення. Найімовірніше, що саме через шлях, який має найвищий загальний рівень LQI, повідомлення до пункту призначення.

2.2.7 Оцінка чистоти каналу

PHY-рівень відповідальний за процедуру оцінки чистоти каналу. Вона необхідна для того, щоб переконатися, що канал не використовується жодним

іншим пристроєм. Процедура CCA повинна виконуватись згідно, щонайменше, одного з чотирьох існуючих режимів [6].

У першому режимі CCA враховується тільки результат виявлення енергії. Якщо рівень енергії перевищує поріг ED, тоді канал вважається зайнятим. Пороговий рівень ED не регламентується стандартом та може бути встановлений виробником.

Другий режим CCA використовує тільки прослуховування каналу. У випадку якщо виявлено сигнал, що сумісний зі стандартом IEEE 802.15.4, а також його тип модуляції та інші PHY-характеристики підтримуються пристроєм, канал вважається зайнятим.

Третій режим CCA є комбінацією перших двох. У ньому канал вважається зайнятим, коли значення енергії перевищує заданий поріг та (або) виявлений сигнал, що підтримується пристроєм.

У четвертому режимі CCA канал завжди вважається вільним. Цей режим зазвичай використовується застосувань "low duty cycle".

2.2.8 Служби PHY

Фізичний рівень надає два типи служб: служба даних PHY та служба управління PHY. Служба передачі даних PHY дозволяє передавати та приймати блоки даних протоколу PHY (англ. PHY Protocol Data Unit, PPDU) по радіоканалу. Рівень PHY включає об'єкт управління PLME. Цей об'єкт надає інтерфейси служби управління рівнем, через які можна викликати функції керування. PLME також відповідає за підтримку бази даних об'єктів, що відносяться до PHY. Ця база даних називається інформаційною базою PHY PAN (англ. PAN information base, PIB) [6].

2.2.9 Константи та атрибути

Константи визначають такі характеристики, які не можна змінювати у процесі роботи пристрою [6]. Усі константи PHY-рівня (так само як і MAC) мають загальний префікс "a".

Стандарт IEEE 802.15.4 визначає дві константи фізичного рівня, що мають зміст для типів PHY підтримуваних ZigBee:

– максимальний розмір фрейму, *aMaxPhyPacketSize*;

– час переключення від передачі на прийом, та навпаки, *aTurnaroundTime*.

Атрибути – це значення, які можуть змінюватися під час роботи. Атрибути PHY містяться в інформаційній базі PHY PAN (англ. PHY PAN informational base, PHY-PIB). Ці атрибути необхідні для управління службами PHY. Існують як атрибути, які дозволено змінювати вищим рівням у структурі стеку, так і ті які доступні їм лише для читання.

2.3 Базові поняття мережевого рівня

2.3.1 Вузли ZigBee

У мережі ZigBee можуть існувати три загальні типи вузлів, які відповідають ролям пристроїв [4]:

– координатор;

– маршрутизатор;

– кінцевий пристрій.

Усі мережі ZigBee повинні мати одного (і лише одного) Координатора.

На рівні мережі Координатор в основному потрібен при ініціалізації системи – це перший вузол, який запускається і виконує такі завдання ініціалізації:

– Вибирає частотний канал, який буде використовуватися мережею (як правило, той, з найменшою виявленою активністю).

– Запускає мережу.

– Дозволяє дочірнім вузлам приєднуватися до мережі через нього.

Координатор може додатково надавати інші послуги, такі як маршрутизація повідомлень та управління безпекою. Він також може надавати послуги на рівні програми. Якщо використовується будь-яка з цих додаткових послуг, координатор повинен мати можливість їх надавати у будь-який час. Однак якщо жодна з цих додаткових послуг не буде використана, мережа зможе нормально працювати, навіть якщо Координатор вийде з ладу або вимкнений.

Мережа ZigBee зазвичай має принаймні один маршрутизатор.

Основні завдання маршрутизатора:

– Відтворює повідомлення від одного вузла до іншого

– Дозволяє дочірнім вузлам приєднуватися до мережі через нього

– Зауважте, що маршрутизатор не може спати, оскільки він повинен бути завжди доступний для маршрутизації.

Основними завданнями Кінцевого пристрою на мережевому рівні є надсилання та отримання повідомлень. Кінцевий пристрій може спілкуватися лише безпосередньо зі своїм батьківським, тому всі повідомлення до та від кінцевого пристрою передаються через його батько.

Кінцевий пристрій може працювати від акумулятора і, не передаючи і не приймаючи, може спати, щоб заощадити енергію. Повідомлення, призначені для кінцевого пристрою, що підтримує сон, буферизується його батьківським пристроєм для збору Кінцевим пристроєм, коли він прокидається (див. Розділ 2.2.2 нижче).

Зауважте, що кінцеві пристрої не можуть передавати повідомлення та не можуть дозволяти іншим вузлам підключатися до мережі через них – тобто вони не можуть мати дочірніх вузлів.

2.3.2 Mesh-мережі

Стандарт ZigBee був розроблений для полегшення бездротових мереж з топологією Mesh [5].

Мережа Mesh складається з координатора, маршрутизаторів та кінцевих пристроїв. Координатор асоціюється з набором маршрутизаторів та кінцевих пристроїв – його дітей. Потім маршрутизатор може бути пов’язаний із більшою кількістю маршрутизаторів та кінцевих пристроїв – його дочірніх пристроїв. Це може продовжуватися до ряду рівнів. Зв'язки між вузлами повинні відповідати наступним правилам:

Координатор та маршрутизатори можуть мати дітей, і тому можуть бути батьками.

Маршрутизатор може бути як дочірнім вузлом, так і батьківським.

Кінцеві пристрої не можуть мати дочірні пристрої і тому не можуть бути батьківськими.

Правила зв'язування для mesh-мережі такі:

– Кінцевий пристрій може спілкуватися лише безпосередньо зі своїм батьківським (і без жодного іншого вузла).

– Маршрутизатор може безпосередньо спілкуватися зі своїми дітьми, з власним батьком та з будь-яким іншим маршрутизатором або координатором, що знаходиться в межах радіо.

– Координатор може безпосередньо спілкуватися зі своїми дітьми та з будь-яким маршрутизатором, що знаходиться в межах радіо.

У ZigBee максимальна глибина (кількість рівнів нижче координатора) мережі - 15. Максимальна кількість переходів, які може зробити повідомлення, пересуваючись між джерелами та пунктами призначення, становить 30 (вдвічі перевищує максимальну глибину).

Здатність вузла маршрутизації (маршрутизатора або координатора) безпосередньо спілкуватися з іншими вузлами маршрутизації (в межах

радіодіапазону) є специфічною властивістю, яка відрізняє мережу Mesh від мережі дерева. Ця властивість призводить до дуже ефективного та гнучкого поширення повідомлень, і означає, що альтернативні маршрути можуть бути знайдені, якщо посилання не працює або є перевантаженість.

Зауважте, що кінцевий пристрій, який може спати , не може безпосередньо приймати повідомлення. Повідомлення, призначене для Кінцевого пристрою, який підтримує сон, завжди буферизується у своєму батьківському вузлі, якщо Кінцевий пристрій спить, коли повідомлення надходить. Як тільки Кінцевий пристрій прокидається, він повинен запитувати або "опитувати" батьків із повідомленнями.

У топології Mesh передбачена функція "відкриття маршруту" , яка дозволяє мережі знайти найкращий доступний маршрут для повідомлення. Розкриття маршруту описано далі в Розділі 2.5.2.

Зауважте, що розповсюдження повідомлень обробляється програмним забезпеченням мережевого рівня та є прозорим для прикладних програм, що працюють на вузлах.

2.3.3 Таблиці сусідніх вузлів

Вузол маршрутизації (маршрутизатор або координатор) містить інформацію про сусідні вузли. У таблиці сусідів містяться записи про безпосередніх дочірніх вузлів, про батьківський вузол та в mesh-мережі для всіх маршрутизаторів, з якими вузол має прямий радіозв'язок [5].

Можна визначити максимальну кількість записів у таблиці сусідів. Якщо для цього параметра встановлено низьке значення, це призведе до "довгої тонкої мережі".

2.3.4 Мережева адресація

У мережі ZigBee кожен вузол повинен мати унікальну ідентифікацію. Це досягається за допомогою двох адрес: MAC-адреси та мережевої адреси [7].

Адреса IEEE (MAC) – це 64-бітна адреса, виділена IEEE, яка однозначно ідентифікує пристрій - жоден два пристрої у світі не можуть мати однакову IEEE-адресу. Його часто називають MAC-адресою, а в мережі ZigBee іноді називають "розширеною" адресою.

Мережева адреса ідентифікує вузол у мережі та є локальною для цієї мережі (таким чином, два вузли в окремих мережах можуть мати однакову мережеву адресу). Мережева адреса має займає 16 біт тому, її іноді називають "короткою" адресою.

У ZigBee мережева адреса вузла динамічно призначається батьківським вузлом випадковим 16-бітовим значенням, коли вузол вперше приєднується до мережі. Незважаючи на випадковість, батьківський вузол гарантує, що обрана адреса вже не була призначена одному з її сусідів. У випадку, коли трапиться, що адреса вже існує в мережі та присвоєна небезпосередньому сусіду, існує механізм автоматичного виявлення та вирішення конфлікту. Виділена коротка адреса залишається закрпленою за пристроєм навіть, якщо він покине мережу по потім приєднається знову, на цей раз – до іншого вузла.

Координатор завжди має мережеву адресу 0x0000.

Хоча програма на вузлі може використовувати як MAC-адреси так і мережеві адреси для ідентифікації віддалених вузлів, стек ZigBee завжди використовує для цього мережеві адреси. Для полегшення переходу між MAC-адресами та мережевими адресами на вузлі може підтримуватися таблиця адрес, де кожен запис таблиці містить пару адрес для віддаленого вузла.

2.3.5 Ідентичність мережі

Мережа ZigBee повинна бути однозначно ідентифікованою. Це дозволяє працювати кільком мережам ZigBee в безпосередній близькості одна від одної – вузли, що працюють в тому ж просторі, повинні бути в змозі визначити, до якої мережі вони належать. Для цього ZigBee використовує два ідентифікатори: PAN ID та розширений PAN ID [7].

16-бітовий індетифікатор під назвою PAN ID (Ідентифікатор персональної локальної мережі) використовується в комунікації між вузлами (реалізовані на рівнях підтримуваних IEEE 802.15.4) для ідентифікації відповідної мережі. Значення для ідентифікатора PAN вибирається випадковим чином координатором при запуску мережі. Коли інші вузли приєднуються до мережі, вони дізнаються ідентифікатор PAN мережі та використовують її у всіх наступних комунікаціях з мережею.

Можливо, що ідентифікатор PAN, сформований для знову встановленої мережі, зіткнеться з ідентифікатором PAN іншої мережі, яка вже працює у тому ж частотному діапазоні (у тому ж каналі). У цьому випадку ZigBee автоматично вирішує такий конфлікт, генеруючи інший випадковий ідентифікатор PAN для нової мережі, поки не буде отримано значення , яке не зіткнеться з ідентифікатором PAN будь-якої іншої мережі.

Розширений ідентифікатор PAN (Extended PAN ID, EPID) має розмір 64 біт Він використовується для формування мережі та подальшої модифікації мережі, якщо це необхідно. Цей ідентифікатор можна заздалегідь встановити на випадкове значення в користувацькій програмі, яка працює на координаторі. У іншому випадку ідентифікатор встановлюється у значення нуль, і в цьому випадку координатор при запуску мережі присвоїть розширеному ідентифікатору PAN значення 64-бітної MAC-адреси і у такий спосіб буде досягнена унікальність значення індетифікатора.

Коли маршрутизатор або кінцевий пристрій вперше намагається знайти мережу для приєднання, він використовуватиме розширений ідентифікатор PAN будь-яким із нижче описаних способів.

Якщо в користувацькій програмі для маршрутизатора або кінцевого пристрою було попередньо встановлено розширений ідентифікатор PAN, вузол приєднається до мережі, яка має цей розширений ідентифікатор PAN (за умови виявлення цієї мережі).

Якщо за маршрутизатором або кінцевим пристроєм немає попередньо встановленого розширеного ідентифікатора PAN, вузол приєднається до першої виявленої мережі, незалежно від розширеного ідентифікатора PAN. Потім вузол, що приєднується, дізнається розширений ідентифікатор PAN своєї мережі та пізніше використовувати цей ідентифікатор для повторного приєднання до мережі, якщо з якихось причин він втратить контакт з мережею.

2.4 Створення мережі

2.4.1 Запуск мережі

Координатор відповідальний за запуск мережі. Після запуску координатора, він проходить наступні етапи ініціалізації мережі [4]:

1. Встановлення EPID (англ. extended PAN ID) та адреси координатора
2. Вибір радіоканала
3. Встановлення ідентифікатора PAN мережі
4. Очікування запитів на приєднання від інших пристроїв

Розглянемо процес детальніше.

Координатор спочатку встановлює значення розширений ідентифікатор PAN (EPID) у 64-бітне значення, вказане в програмі координатора (якщо це значення дорівнює нулю, EPID буде встановлено на 64-бітну MAC-адресу пристрою

координатора). Також координатор присвоює 16-бітній мережевій адресі координатора значення 0x0000.

Потім координатор вибирає радіоканал, по якому буде функціонувати мережа, у межах вибраного частотного діапазону. Координатор виконує Energy Detection Scan, в якому він сканує радіочастотний діапазон, щоб знайти вільний канал (сканування можна запрограмувати на «прослуховування» конкретних каналів). Обирається канал з найменшою виявленою активністю.

Після вибору радіоканалу координатор вибирає 16-бітний ідентифікатор PAN для мережі. Для цього він прослуховує в каналі трафік з інших мереж і ідентифікує ідентифікатори PAN цих мереж (якщо такі є). Щоб уникнути конфліктів, координатор присвоює своїй власній мережі випадковий ідентифікатор PAN, який не використовується іншою мережею.

Тепер координатор готовий приймати запити від інших пристроїв (маршрутизаторів та кінцевих пристроїв) для бездротового підключення до мережі через нього.

2.4.2 Приєднання до мережі (маршрутизатори та кінцеві пристрої)

Маршрутизатори та кінцеві пристрої можуть приєднатися до існуючої мережі, вже створеної координатором. Координатор та маршрутизатори мають можливість дозволяти іншим вузлам приєднуватися до мережі через них. Процес приєднання описаний нижче.

Новий вузол спочатку сканує канали відповідної смуги радіочастот, щоб знайти мережу. Навіть на одному каналі можуть працювати кілька мереж, і відповідальність за вибір мережі бере на себе програма пристрою (наприклад, це рішення може базуватися на заздалегідь визначеному розширеному ідентифікаторі PAN) [5].

Тепер даний вузол вибирає батьківський вузол у вибраній мережі, прослуховуючи мережеву активність. Вузол може «чути» декілька маршрутизаторів та координатора з мережі. Вузол вибирає батьківський з найменшою глибиною в мережі – тобто такий, що найближчий до координатора (для координатора глибина має нульове значення).

Новий вузол надсилає повідомлення бажаному вузлу, просячи приєднатися до мережі.

Тепер вузол чекає відповіді від потенційного батьківського вузла, який визначає, чи вузол є дозволеним пристроєм і чи дозволяє батьківський вузол в даний час приєднуватися іншим пристроям. Щоб визначити, чи приєднаний вузол є дозволеним пристроєм, батьківський вузол звертається до довірчого центру (якщо він сам ним не являється). Якщо ці критерії будуть задоволені, батьківський вузол дозволить вузлу приєднатися до мережі як його дочірньому. Далі батьківський вузол виділяє 16-бітну мережеву адресу для дочірнього вузла та надсилає його у відповіді про прийняття.

Потенційний батьківський вузол може відхилити запит на приєднання, та надішле відповідь про це цьому вузлу, після чого останній спробує підключитись до іншого вузла (чи іншої мережі).

Новий вузол взнає ідентифікатор PAN та розширений ідентифікатор PAN мережі, а також мережеву адресу, якій він був призначений. Ідентифікатор PAN необхідний для комунікації з мережею, а розширений ідентифікатор PAN потрібен, якщо в майбутньому йому доведеться знову приєднатися до мережі (після повторного приєднання він також може повторно використовути свою мережеву адресу).

Маршрутизатор або координатор можуть бути налаштовані таким чином, щоб мати час, протягом якого дозволяється приєднання (контролюється статусом "permit joining"). Час приєднання може бути заданий користувачем, наприклад натисканням кнопки. Також можна встановити нескінченний період з'єднання, щоб дочірні вузли могли приєднатися до батьківського вузла в будь-який час.

2.5 Базові поняття прикладного рівня

2.5.1 Програми та кінцеві точки

Вузол може мати кілька програм, що працюють на ньому – наприклад, вузол у розумній домашній мережі може виконувати функції датчика присутності та перемикач світла, кожна з яких є програмою. Насправді кожна програма реалізує тип пристрою ZigBee. Доступ до програм відбувається через кінцеві точки (англ. endpoint), які виконують роль портів зв'язку для програм [9].

Для того, щоб надіслати повідомлення на відповідний програму (application instance) у вузлі, необхідно вказати відповідну кінцеву точку. Кінцеві точки нумеруються від 1 до 240. Отже, для зв'язку з віддаленою програмою в мережі ZigBee, вам потрібно надати адресу віддаленого вузла разом із необхідним номером кінцевої точки на вузлі.

Кінцева точка 255 – це номер кінцевої точки широкомовної передачі. Вказавши адресатом даний номер кінцевої точки, дані будуть надіслані всім програмам на вузлі.

2.5.2 Дескриптори

Програмі може знадобитися отримати інформацію про вузли мережі, в якій вона працює. Для цього використовується інформація, що зберігається в дескрипторах у вузлах.

Є три обов'язкові дескриптори та два необов'язкові дескриптори. Обов’язковими дескрипторами є дескриптор вузла, дескриптор живлення вузла та простий дескриптор, тоді до необов'язкових дескрипторів входять комплексний

(англ. Complex descriptor) дескриптор та дескриптор користувача (англ. User descriptor) [9].

Дескриптори вузла та живлення вузла створюються по одному для кожного вузла, тоді як прості дескриптори – для кожної кінцевої точки.

Розглянемо детальніше дескриптори вузла, живлення вузла та простий дескриптор.

Простий дескриптор програми включає наступні дані:

– кінцева точка , на якій працює програма;

– тип пристрою ZigBee, який реалізує програма;

– кластери ZigBee, які реалізує тип пристрою;

– чи є відповідні комплексні дескриптори та дескриптори користувачів;

– списки вхідних та вихідних кластерів, які програма використовує та надає відповідно.

Дескриптор вузла містить наступну інформацію про можливості вузла:

– тип (кінцевий пристрій, маршрутизатор або координатор);

– діапазон частот, що використовується (868 МГц, 902 МГц або 2400 МГц);

– Можливості IEEE 802.15.4 MAC:

- Пристрій може бути координатором PAN
- вузол реалізує повнофункціональний або пристрій з зменшеною функціональністю IEEE 802.15.4
- чи пристрій живиться від мережі
- чи ристрій здатний використовувати захист MAC

– чи приймач залишається увімкненим протягом періоду очікування

– код виробника

– максимальний розмір буфера (найбільший пакет даних, який може бути надісланий програмою за одну операцію)

Дескриптор живлення вузла містить наступну інформацію:

– режим живлення (визначає чи приймач пристрою увімкнено, або прокидається з періодом визначеним мережею, або лише тоді, коли вимагає програма)

– доступні джерела живлення (вказує, якими джерелами можна живити пристрій: акумуляторами чи одноразовими батареями, наприклад);

– поточні джерела живлення, які вказують, яке джерело живлення (електромережі, акумуляторні або одноразові батареї) використовується зараз для живлення пристрою;

– поточний рівень джерела живлення, вказує рівень заряду поточного джерела живлення.

2.5.3 Профілі програм

Однією з цілей ZigBee 3.0 є об'єднання специфічних для ринку профілів додатків ZigBee, які збирають разом пов'язані між собою типи пристроїв. Ідентифікатори профілю програм все ще потрібні в ZigBee 3.0 (це забезпечує зворотну сумісність з більш ранніми версіями ZigBee), але відбулося деяке об'єднання (consolidation) ідентифікаторів – наприклад, ZigBee Light Link та Home Automation були обєднані ідентифікатором профілю програми 0x0104, який зараз відповідає пристроям ZigBee Lighting and Occupancy (ZLO) [9].

2.5.4 Типи пристроїв

Щоб забезпечити сумісність вузлів ZigBee від різних виробників, ZigBee Alliance визначив набір стандартних типів пристроїв. Тип пристрою – це програмне забезпечення, яке визначає функціональність пристрою. Ця функціональність сама визначається кластерами, що входять до типу пристрою, де кожен кластер відповідає певному функціональному аспекту (наприклад, контроль рівня заряду) пристрою [4].

Пристрій – це екземпляр типу пристрою і реалізується програмою, яка працює на кінцевій точці. Тип пристрою зазвичай підтримує як обов'язкові, так і необов'язкові кластери, тому пристрій можна налаштувати за допомогою останніх. Тип пристрою, реалізований програмою, визначений у простому дескрипторі програми. Вузол може реалізувати навіть більше чим один тип пристроїв, кожен з яких відповідає програмі пристрою, яка працює на власній кінцевій точці.

Кожен вузол ZigBee повинен використовувати базовий пристрій ZigBee, який забезпечує основу для використання типів пристроїв ZigBee і обробляє основні операції, наприклад, такі як введення в експлуатацію (базовому пристрою не потрібна кінцева точка).

2.5.5 Кластери та атрибути

Смисл даних (наприклад, вимірювання температури), що обробляється кінцевою точкою ZigBee, називається атрибутом. Програма може спілкуватися через набір атрибутів – наприклад, програма термостата може мати атрибути температури, мінімальної температури, максимальної температури та допуску [5].

Програми ZigBee використовують поняття "кластер" для передачі значень атрибутів. Кластер містить набір пов'язаних атрибутів разом з набором команд для взаємодії з атрибутами – наприклад, команди для читання значень атрибутів.

Кластер відповідає за частину функцій, які виконує пристрій. Загальна функціональність програми визначається типом пристрою ZigBee, який він реалізує, та кластерами, які використовує тип пристрою. Таким чином, кластери – це функціональні будівельні блоки пристроїв [9].

Кластер має два аспекти, які відповідно стосуються прийому та відправки команд (один або обидва аспекти може використовуватися програмою ZigBee):

Вхідний кластер або сервер: Ця сторона кластера використовується для зберігання атрибутів та отримання команд для маніпулювання збереженими

атрибутами – наприклад, вхідний кластер зберігає виміри температури та пов'язані з ними атрибути, і відповідає на команди, які вимагають читання цих атрибутів.

Вихідний кластер або клієнт: Ця сторона кластера використовується для маніпулювання атрибутами у відповідному вхідному кластері, надсилаючи до нього команди (та отримуючи відповіді). Зазвичай це команди запису для встановлення значень атрибутів та команди зчитування для отримання значень атрибутів.

Кластери введення та кластери виходу, передані через кінцеву точку, перераховані (окремо) у простому дескрипторі кінцевої точки (див. Розділ 2.4.2.1).

Для узгодженості та сумісності Альянс ZigBee визначив ряд стандартних кластерів для різних функціональних областей. Вони збираються разом у бібліотеці кластерів ZigBee (ZCL). Таким чином, розробники можуть використовувати стандартні кластери з ZCL у своїх додатках для пристроїв.

Також доступний кластер за замовчуванням (з ідентифікатором 0xFFFF). Якщо кластер за замовчуванням присутній у кінцевій точці та отримано повідомлення, призначене кластеру, який не знаходиться у списку підтримуваних вхідних кластерів кінцевої точки, це повідомлення все одно буде передано додатку (за умови, що воно надходить із визначеного профілю програми). Кластер за замовчуванням повинен бути явно доданий до кінцевої точки.

2.5.6 Виявлення

Специфікація ZigBee забезпечує можливість пристроїв дізнаватися про можливості інших вузлів у мережі, наприклад, їх адреси, які типи програм працюють на них, їх джерело живлення та режим сну. Ця інформація зберігається в дескрипторах на кожному вузлі і використовується запитуючим вузлом для адаптації його поведінки до вимог мережі. Виявлення зазвичай використовується, коли вузол вводиться в налаштовану користувачем мережу, наприклад, внутрішню

систему безпеки або управління освітленням. Для запуску інтеграції пристрою в мережу може знадобитися натиснути кнопку або щось подібне. Перше завдання – з’ясувати, чи є відповідні пристрої, з якими новий вузол може спілкуватися [5].

Виявлення пристрою повертає інформацію про адреси мережевого вузла. Отримана інформація може бути MAC-адресою вузла з заданою мережевою адресою або мережевою адресою вузла з заданою MAC-адресою. Якщо вузол, який допитують, є маршрутизатором або координатором, він може додатково надавати адреси всіх пристроїв, які з ним пов'язані, а також свою власну адресу. Таким чином можна виявити всі пристрої в мережі, запитавши цю інформацію у Координатора (мережева адреса 0x0000), а потім скориставшись списком адрес, що відповідають дітям Координатора, для запуску інших запитів про їх дочірні вузли.

Виявлення сервсів дозволяє вузлу запитувати інформацію від віддаленого вузла про можливості віддаленого вузла. Ця інформація зберігається в ряді дескрипторів на віддаленому вузлі і включає:

– Тип пристрою та можливості вузла

– Силові характеристики вузла

– Інформація про кожну програму, що працює на вузлі

– Додаткова інформація, така як серійні номери

– Інша інформація, визначена користувачем – наприклад, легко зрозумілі імена, такі як "MtgRoomLight"

Запити щодо цих дескрипторів здійснюються пристроєм під час процесу виявлення, який, як правило, є частиною конфігурації пристрою та інтеграції в мережу ZigBee.

2.5.7 Об'єкти пристрою ZigBee (ZDO)

Спеціальна програма, спільна для всіх пристроїв ZigBee, надається для управління різними описаними процесами. Ця програма – це об'єкти пристрою

ZigBee або ZDO [9]. Він знаходиться в прикладному рівні вузла і може спілкуватися з віддаленими вузлами через кінцеву точку 0, використовуючи профіль пристрою ZigBee (ZDP) та пов'язані з ним кластери. Він виконує наступні ролі:

– Визначає тип мережевого пристрою: координатор, маршрутизатор або кінцевий пристрій.

– Ініціалізує вузол, щоб дозволити запуск програм.

– Виконує процеси виявлення пристрою та виявлення служби.

– Реалізує процеси, необхідні для дозволу координатору створити мережу, а маршрутизатори та кінцеві пристрої приєднатися до мережі та покинути її.

– Ініціює і відповідає за зв'язування (binding) запитів.

– Надає служби безпеки, що дозволяють встановлювати захищені з'єднання між програмами.

– Дозволяє віддаленим вузлам отримувати інформацію з вузла, наприклад таблиці маршрутизації та прив'язки, та виконувати віддалене управління вузлом, наприклад, доручаючи йому залишати мережу.

ZDO використовує сервіси в стеку для реалізації цих ролей і надає засоби, що дозволяють користувачам програмам отримувати доступ до послуг стеку.

2.6 Мережева маршрутизація

Основна операція в мережі – це передача даних з одного вузла в інший. Дані отримуються з вхідного пристрою (можливо, комутатора або датчика) на вихідний вузол і передаються іншому вузлу, який може інтерпретувати та використовувати дані.

При найпростішому виді передачі дані передаються безпосередньо від вузла джерела до вузла призначення. Однак якщо два вузли знаходяться далеко один від одного або перебувають у складній обстановці, з'єднання напряму може бути неможливим. У цьому випадку необхідно передати дані на інший вузол в

радіодіапазоні, який потім передає його на інший вузол, і так далі, поки не буде досягнуто вузла призначення – тобто використовувати один або кілька проміжних вузлів. Процес прийому даних, призначених для іншого вузла, і передачі їх далі відомий як маршрутизація.

Маршрутизація дозволяє розширити дальність мережі на відстані, що не підтримувані прямим радіозв'язком. Віддалені пристрої можуть приєднатися до мережі, підключившись до маршрутизатора.

2.6.1 Адресація та поширення повідомлень

Якщо повідомленню, яке надіслане з одного вузла в інший, потрібно пройти через один або кілька проміжних вузлів, щоб досягти свого кінцевого пункту (дозволено до 30 таких переходів), повідомлення містить дві адреси призначення:

– Адреса кінцевого пункту призначення

– Адреса вузла, який є наступним переходом (hop)

ZigBee PRO призначений для mesh-мереж, в яких шлях поширення повідомлення (маршрут) залежить від того, чи знаходиться цільовий вузол у радіодіапазоні [9]:

– Якщо цільовий вузол знаходиться в діапазоні, використовується тільки адреса "остаточного призначення".

– Якщо цільовий вузол не знаходиться в діапазоні, адреса "наступного стрибка" – адреса першого вузла в маршруті до кінцевого пункту призначення.

Адреса "наступного переходу" визначається за допомогою даних, що зберігаються в таблиці маршрутизації у вузлі маршрутизації (маршрутизаторі або координаторі). Запис цієї таблиці містить інформацію про віддалений вузол, включаючи мережеві адреси віддаленого вузла та наступного вузла маршрутизації в маршруті до віддаленого вузла. Таким чином, коли повідомлення отримується вузлом маршрутизації, він шукає адресу призначення у своїй таблиці

маршрутизації та витягує з наступної таблиці адресу «наступного переходу» для вставки в повідомлення. Потім повідомлення передається і поширення продовжується таким чином, поки не буде досягнуто цільового вузла.

Зауважте, що якщо джерелом повімлення є кінцевий пристрій, повідомлення завжди буде спочатку передане батьківському вузлу джерела, перш ніж передаватися.

2.6.2 Виявлення маршруту

Мережевий рівень стека ZigBee підтримує функцію "виявлення маршруту", яка знаходить найкращий доступний маршрут до пункту призначення при відправці повідомлення. Повідомлення зазвичай направляється по вже виявленому mesh-маршруту, якщо такий існує, інакше вузол маршрутизації (маршрутизатор або координатор), що бере участь у надсиланні повідомлення, ініціює пошук маршруту. Після завершення повідомлення буде надіслано по розрахованому маршруту [9].

Механізм пошуку маршруту між двома кінцевими пристроями має наступні кроки:

1. Трансляція (broadcast) виявлення маршруту надсилається батьківським вузлом вихідного кінцевого пристрою і містить мережеву адресу кінцевого пристрою.

2. Усі вузли маршрутизації з часом отримають трансляцію, у тому числі і батьківський вузол кінцевого пристрою, який є вузлом призначення.

3. Цей батьківський вузол надсилає назад відповідь, адресовану батьківському вузлу джерела.

4. Коли відповідь проходить назад по мережі, записується кількість стрибків і міра якості сигналу для кожного переходу. Кожен вузол маршрутизації може створити запис таблиці маршрутизації, що містить найкращий шлях до кінцевого

пристрою призначення. Вибір найкращого шляху, як правило, відповідає маршруту з найменшою кількістю стрибків. але якщо якість сигналу на такому маршруті низька (і, отже, більший шанс, що потрібні повторні спроби), може бути вибраний маршрут з більшою кількістю переходів.

5. Кожен вузол маршрутизації буде мати запис таблиці маршрутизації та встановлюється маршрут від джерела до кінцевого пристрою.

Якщо джерелом виявляється маршрутизатор або координатор, то воно реалізує пошук маршруту аналогічно описаному вище, за винятком того, що маршрутизатор транслює власне повідомлення про виявлення маршруту (не вимагаючи від цього свого батьківського вузла).

2.6.3 Маршрутизація "від багатьох до одного"

Поширеним для бездротової мережі є сценарій, коли є необхідність багатьох вузлів мережі спілкуватися з одним вузлом, який виконує певну централізовану функцію, наприклад шлюз. Такий вузол часто називають концентратором.

Щоб встановити зв'язок з концентратором, кожен віддалений вузол може ініціювати «пошук маршруту», що відобразиться у таблиці маршрутизації кожного вузла маршрутизації. Якщо більшість вузлів мережі хочуть встановити з'єднання з концентратором, може бути розпочато виявлення таких маршрутів. Якщо дані маршрути мають спільну гілку, відповідні записи таблиці маршрутизації не будуть дублюватися, а будуть спільними. Однак велика кількість одночасних маршрутів може зажадати значного простору пам’яті у вузлах біля концентратора для тимчасового зберігання інформації про виявлення маршруту, і, можливо, це призведе до переповнення пам’яті та перевантаженості трафіку [7].

Більш ефективний метод встановлення маршрутів до концентратора полягає в тому, щоб концентратор ініціював відкриття маршруту "від багатьох до одиного" для маршрутів з усіх інших вузлів мережі до себе. Для цього концентратор

транслює запит на пошук маршруту, а таблиці маршрутизації оновлюються в міру поширення трансляції по мережі. Оскільки відповіді не створюються, тимчасове зберігання інформації про виявлення маршруту не потрібно, а перевантаженість мережевого трафіку мінімальна.

Виявлення маршруту від багатьох до одного проілюстровано на рисунку нижче.

Щоб уникнути зберігання зворотних маршрутів (від концентратора) у таблицях маршрутизації проміжних вузлів, використовується техніка маршрутизації джерела – концентратор запам'ятовує вихідний маршрут, який він отримав з прийнятого повідомлення, і вбудовує його у відповідь. У цьому випадку відповідь може містити не більше 30 адрес вузлів уздовж зворотного маршруту (так як максимальна кількість дозволених переходів – 30).

2.7 Мережеві комунікації

У цьому розділі розглядаються процеси, необхідні для того, щоб дозволити мережі обмінюватися інформацією та виконувати корисні функції. Для спілкування один з одним два вузли повинні бути сумісними. Інакше кажучи, один вузол створює дані, які інший вузол сприймає та інтерпретує відповідно до їх змісту. Наприклад, вузол датчика температури виконує вимірювання температури, яке вузол контролера опалення може використовувати для управління центральною системою опалення.

Коли новий вузол приєднується до мережі, він повинен знаходити сумісні вузли, з якими він може спілкуватися – цьому процесу сприяє механізм виявлення сервісу. Потім він повинен вибрати, з яким із сумісних вузлів він буде спілкуватися. Спосіб сполучення вузлів для полегшеної комункації забезпечується механізмом зв'язування (англ. binding).

Виявлення сервісу та зв'язування описані в підрозділах нижче.

2.7.1 Виявлення сервісу

Пристрій, який приєднується до мережі, повинен вміти знаходити інші пристрої в мережі, які можуть використовувати інформацію, яку вона надає, або які можуть генерувати інформацію, необхідну пристрою для виконання власної функції. Вузол може використовувати виявлення сервісу для пошуку вузлів, з якими він може спілкуватися [4].

Вузол запитує необхідні послуги від інших вузлів за допомогою широкомовного повідомлення, яке розповсюджується по всій мережі. Будь-який вузол, у якого є запитувані послуги, потім в односторонньому порядку надсилає відповідь запитуючому вузлу. Це означає, що запитуючий вузол може отримати більше однієї відповіді.

Відповідь включає мережеву адресу віддаленого вузла, що містить запитувані послуги. Вузол зберігає цю адресу локально, і програма може потім використовувати адресу для всіх майбутніх комунікацій з віддаленим вузлом. Це називається прямою адресацією.

Крім того, замість використання прямої адреси у своїх комунікаціях, два вузли можуть спілкуватися через механізм зв'язування, описаний у розділі 2.6.2 нижче.

2.7.2 Зв'язування

Після того, як два вузли виявились сумісними за допомогою виявлення сервісу, вони можуть бути спарені задля подальшої комункації. Наприклад, перемикач може бути спарений з певним освітлювальним пристроєм, і ми повинні переконатися, що цей перемикач вмикає та вимикає лише те джерело світла, яким

йому призначено керувати. Простий спосіб сполучення вузлів для зв'язку забезпечується механізмом зв’язування [4].

Зв'язування дозволяє спаровувати вузли таким чином, що певний тип вихідних даних з одного вузла автоматично переводиться на парний вузол, без необхідності кожного разу вказувати адресу призначення та кінцеву точку. Два вузли спочатку повинні бути зв'язані разом, використовуючи адресу та відповідний номер кінцевої точки для кожного вузла – їх можна отримати за допомогою виявлення служб (сервісів). Зв'язка з'єднує вузол джерела та вузол призначення, відповідно до напрямку, в якому дані будуть надсилатися між вузлами (від джерела до місця призначення). Деталі зв'язки зберігаються як запис у таблиці зв’язування, як правило, утримується у вихідному вузлі прив'язки (але іноді і в іншому номінованому вузлі).

Запит на встановлення зв'язки може бути здійснено мож будь-яким із наступних способів:

– Запит на зв'язку подається до вузла-джерела для зв'язування або самим вузлом-джерелом, або віддаленим вузлом (не одним із вузлів, що зв'язані).

– Запити на прив'язку подаються координатору джерелами та кінцевими вузлами для прив'язки (наприклад, натисканням кнопки на кожному вузлі для створення запиту прив'язки). Два запити на зв'язування повинні бути отримані протягом певного періоду часу.

Під час процесу зв’язування таблиця зв'язок для вихідного вузла оновлюється (або створюється, якщо необхідно).

Зв'язування відбувається на рівні програми за допомогою кластерів. Для того, щоб два додатки були зв'язані, вони повинні підтримувати один і той же кластер.

Прив'язка між двома програмами визначається:

– адреса вузла та номер кінцевої точки джерела прив'язки (наприклад, перемикач світла)

– Адреса вузла та номер кінцевої точки призначення прив'язки (наприклад, контролер навантаження для світла)

– Ідентифікатор кластера для зв'язкування

– Можуть бути досягнуті наступні типи зв'язування:

– Один з одним: Це проста прив'язка, в якій кінцева точка пов'язана з однією (і лише однією) іншою кінцевою точкою, що вимагає єдиного запису таблиці прив'язки.

– Один з багатьма: Це прив'язка, у якій кінцева точка джерела пов'язана з більш ніж однією кінцевою точкою призначення. Зв'язування досягається наявністю декількох записів таблиці прив'язки для однієї кінцевої точки джерела.

– Багато з одним: це прив'язка, в якій більше однієї кінцевої кінцевої точки пов'язано з однією кінцевою точкою призначення. Зв'язування досягається безліччю вузлів, що мають прив'язку один до одного для однієї кінцевої точки призначення.

Як приклад цих прив’язок розглянемо перемикач і контролер навантаження для освітлення:

– У випадку "один з одним" один перемикач керує одним світлом.

– У випадку "один з багатьма" один перемикач керує декількома джерелами світла.

– У випадку "багато з одним" один вимикач керує одним світлом, наприклад, світлом біля сходів, де є вимикачі вгорі та внизу, кожен з яких можна використовувати для включення світла.

Можна також передбачити прив'язки типу "багато з багатьма", коли в останньому сценарії на сходах є декілька вогнів, які контролюються будь-яким вимикачем.

Спосіб налаштування прив’язок залежить від типу мережі наступним чином:

– Попередньо налаштована система: Прив’язки налаштовані фабрично та зберігаються у зображенні програми.

– Система самоконфігурування: прив’язки створюються автоматично під час встановлення мережі за допомогою програм виявлення, які знаходять сумісні вузли чи кластери.

– Спеціальна система: Прив'язки створюються вручну системним інтегратором або інженером із встановлення, який може використовувати

графічний програмний інструмент для малювання ліній зв'язку між кластерами на вузлах.

2.8 Висновки до розділу 2

У даному розділі роботи надано короткий огляд архітектури протоколу ZigBee.

Досліджено ролі основних рівнів стеку протоколу (фізичного рівня PHY, рівня доступу до середовища MAC, мережевого рівня NWK та прикладного рівня APL).

Показано як узгоджена взаємодія цих рівнів (модуляція та демодуляція радіосигналів, взаємодія двох окремо взятих приймача та передавача, маршрутизація повідомлень) забезпечує повноцінне функціонування мережі ZigBee.

3 НЕСИГНАЛЬНЕ ТЕСТУВАННЯ ZIGBEE ПРИСТРОЇВ

3.1 Сигнальний режим тестування

Загальний підхід до тестування безпровідних пристроїв, полягає у так званому сигнальному тестуванні. Під час тестування, зазвичай, відбувається повна симуляція роботи пристрою у реальному часі [10].

При використання цього методу тестування проходить у два етапи.

Перший етап полягає у вимірюванні радіочастотних параметрів пристрою, що тестується (англ. Device under test, DUT) та порівнянні їх з еталонними значеннями. Потім відбувається коригування параметрів та збереження їх у пристрої, що зазвичай потребує його калібрування. Зазвичай такими параметрами є потужність передачі та показник рівня прийнятого сигналу (англ. Received Signal Strength Indication, RSSI) для різних діапазонів частот і технологій.

На другому етапі відбувається перевірка (верифікація) того, що відкалібрований пристрій працює коректно. Виробник вимірює такі параметри передачі, як якість модуляції, спектр та потужність, і порівнює їх з параметрами технології, для якої призначений пристрій (GSM, Bluetooth, WLAN тощо). У більшості випадків абсолютна чутливість приймача визначається за допомогою тесту на коефіцієнт бітових помилок (англ. Bit Error Rate, BER).

У сигнальному методі тестування приймають участь система контролю, тестувальне обладнання та DUT (див. рис 3.1). У сигнальному режимі відбувається повноцінне протокольне з’єднання між тестером та DUT.

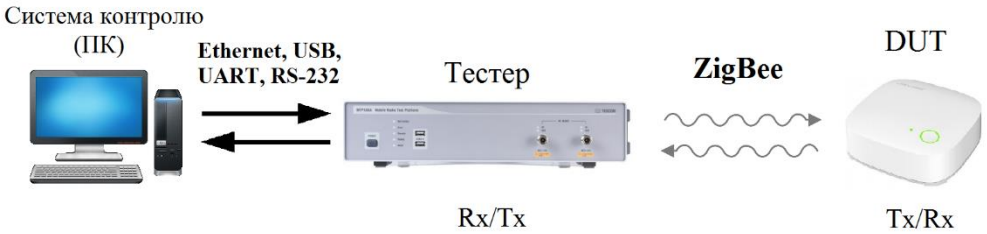


Рисунок 3.1 – Схема тестування у сигнальному режимі

Основним недоліком сигнального методу є необхідність підтримки тестувальним обладнанням повного стеку протоколу. Підключення до пристрою, що тестується, при підтримці повноцінного протокольного з'єднання займає тривалий час, що збільшує загальний час тестування та, як наслідок, призводить до збільшення вартості кінцевого продукту.

У сучасних реаліях швидкого розвитку сфери IoT, пристрої часто повинні підтримувати не один протокол безпровідного зв'язку. Через те, що у різних країнах існують різні вимоги до частотних діапазонів, які дозволено займати, пристрої вимушені підтримувати багаточастоні та багатомодові режими. Це все також створює додаткові складнощі для інженерів-тестувальників.

Спростити задачу покликаний несигнальний метод тестування безпровідних протоколів зв'язку.

3.2 Несигнальний режим тестування

У несигнальному підході до тестування пристрій використовується у спеціальному тестовому режимі, який підтримує калібрування та верифікацію оптимізовану в часі. Під час тестування пристрій дистанційно керується за допомогою провідного інтерфейсу зв'язку. Вимірювальне обладнання для несигнального тестування включає в собі функції аналізатора та генератора, але не емулює повністю функції мережі [11].

При тестуванні передавача безпровідним шляхом DUT лише передає дані, а його робота керується напряму системою контролю через провідний інтерфейс. У випадку тестування приймача, схема аналогічна: тільки у цьому випадку DUT безпровідним способом приймає дані, та надсилає їх системі контролю - провідним. Схема тестування представлена на рис. 3.2.

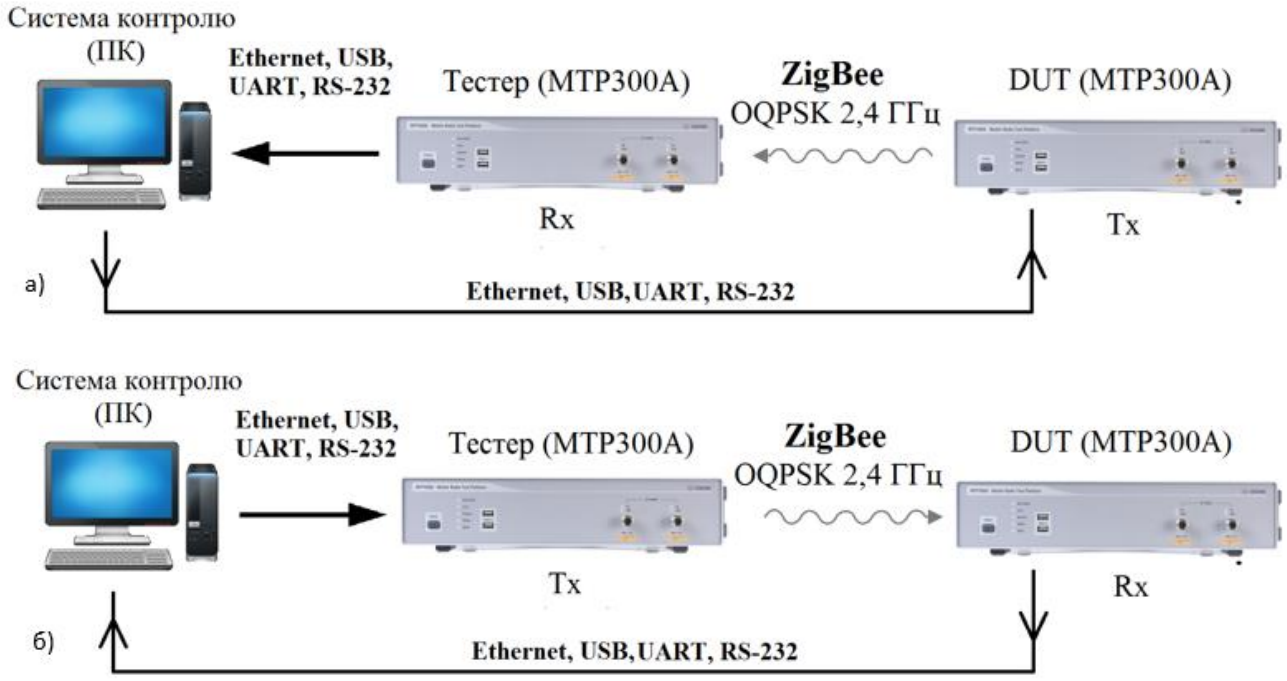


Рисунок 3.2 – Схема тестування передавача (а) та приймача (б) у несигнальному режимі

Несигнальний метод усуває основний недолік сигнального тестування, оскільки за його використання не потрібно повністю підтримувати протокол. А це дає виграш у тривалості тестування, головним чином, за рахунок скорочення часу на з'єднання тестувальної платформи з пристроєм DUT [10].

Зрозуміло, що для використання цього методу необхідні програма для віддаленого керування, а також пристрій повинен підтримувати провідний інтерфейс дистанційного керування.

Такий підхід дозволяє значно пришвидшити процес тестування, але він не може гарантувати, що пристрій у спеціальному режимі буде поводити себе так само, як і в реальних умовах.

Для підвищення надійності несигнальний та сигнальний методи можна комбінувати: калібрування виконувати у несигнальному режимі, а перевірку відкаліброваного пристрою за допомогою сигнального тестування [11].

3.3 Дискретизація сигналу на основі виділення квадратурних складових

Розклад сигналу s(t) з центральною частотою $\omega_ц$ та шириною смуги спектру $\Delta\omega$ на квадратурні складові має наступний вигляд [12]:

$$s(t) = I(t)\cos(\omega_0 t) - Q(t)\sin(\omega_0 t), \quad (3.1)$$

де I(t) – синфазна складова,

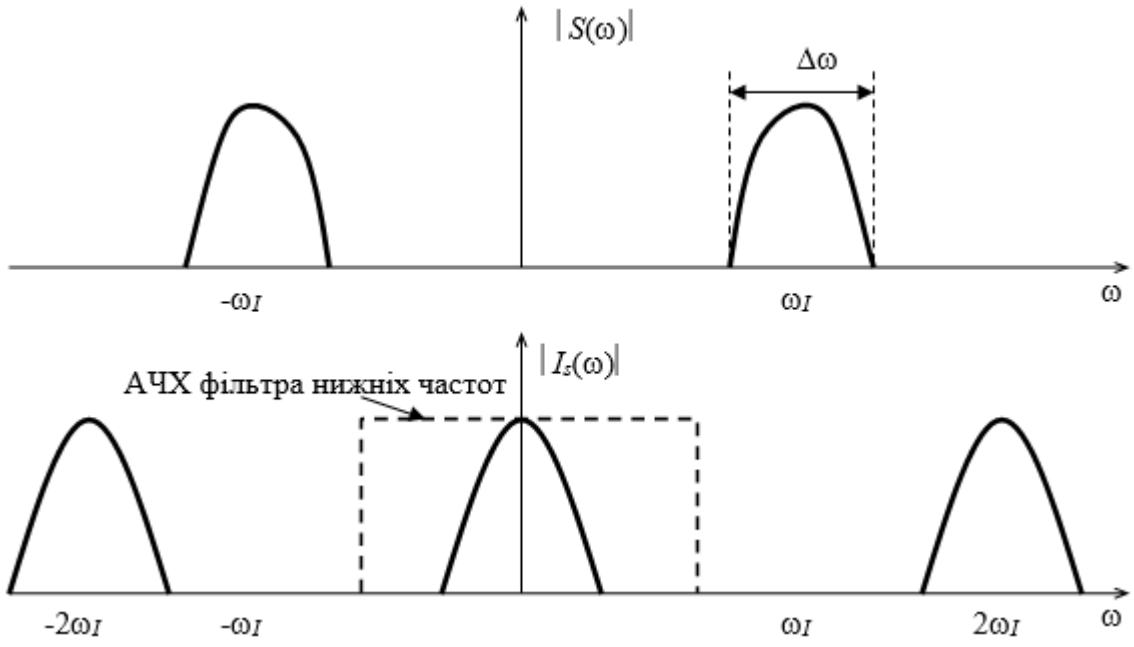
Q(t) – квадратурна складова,

ω_0 – опорна частота квадратурного представлення, що зазвичай приймається такою ж як і центральна частота $\omega_ц$.

Зручність використання такого представлення полягає у тому, що при зсуві спектру сигналу s(t) вліво на значення опорної частоти (при виконанні умови $\omega_0 = \omega_ц$) центр його спектру опиниться на нульовій частоті та смуга співпадатиме зі смугою його квадратурних складових. Таким чином сигнали I(t), Q(t) є звичайними низькочастотними сигналами зі скінченним спектром з верхньою частотою $\Delta\omega/2$. Їх можна дискретизувати і відновлювати як звичайний сигнал з фінітним спектром, а вже маючи відновлені копії квадратурних складових I(t), Q(t), можна, підставивши їх у формулу (3.1), відновити високочастотне заповнення або, іншими словами, зробити зворотний зсув по частоті в область вихідних (високих) частот і отримати відновлену копію вихідного сигналу s(t).

Дискретизація вузькосмугового сигналу на основі виділення його квадратурних складових ілюструється діаграмами в частотній області на рис. 3.3 і схемою на рис. 3.4. Помноживши вихідний сигнал s(t) на гармонічний сигнал опорної частоти ω_0 формуються сигнали $I_s(t)$, $Q_s(t)$, в спектрі яких містяться складові суми ($\omega_I+\omega_0$) і різниці ($\omega_I-\omega_0$) частот. За допомогою фільтрів ФНЧ-1 (їх частота зрізу однакова і не повинна перевищувати частоту $2\omega_0-\Delta\omega/2$) складові з

сумарними частотами придушуються, а сигнали $I_s(t)$, $Q_s(t)$, що містять тільки різницеві частоти, є квадратурними складовими і повинні пропускатися фільтрами без спотворень. Після їх дискретизації з частотою $f_д \geq \Delta f$ отримуємо дві послідовності відліків. Кожна з них може бути відновлена, наприклад, за допомогою ідеальної низькочастотної фільтрації за допомогою ФНЧ-2 (його частота зрізу повинна бути узгоджена з частотою 0,5 $f_д$). Отримані таким чином відновлені копії квадратурних складових $I_f(t)$, $Q_f(t)$ підставляються в формулу (3.1), обчислення за якою дозволяє для кожного моменту часу t обчислити значення відновленої копії вхідного сигналу $s'(t)$. У разі точного виконання всіх зазначених операцій виконується точно рівність $s'(t) = s(t)$. В іншому випадку виникають похибки.



|S(ω)| – модуль спектра вихідного смугового сигналу; |Is(ω)| – модуль спектра синфазної складової після помноження на опорне коливання

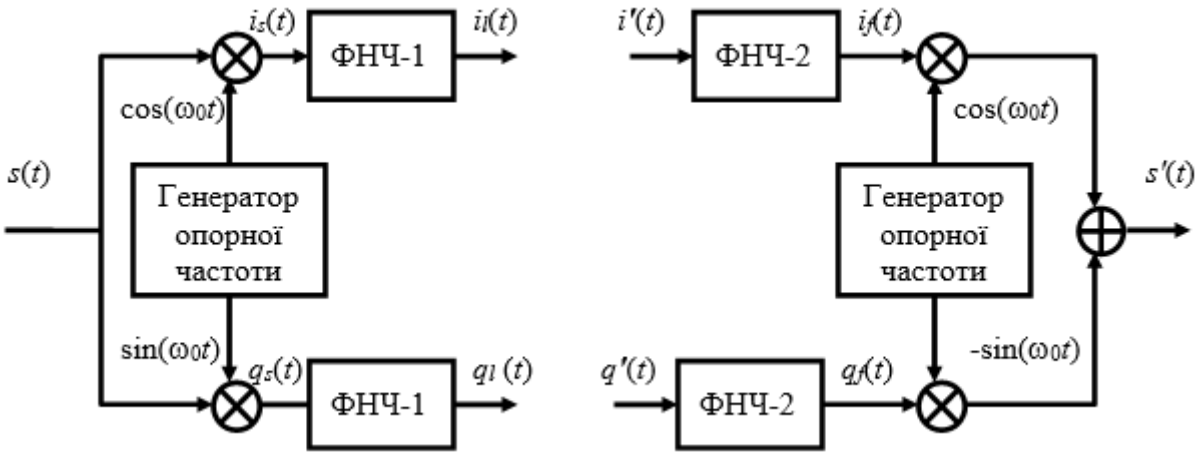
Рисунок 3.3 – Ілюстрація процесу виділення синфазної складової в частотній області

Процес виділення квадратурних складових із сигналу $s(t)$ можна інтерпретувати на основі тригонометричних перетворень. Розглянемо сигнали $I_s(t)$ та $Q_s(t)$:

$$I_s(t) = s(t)\cos(\omega_0 t) = I(t)\cos^2(\omega_0 t) + Q(t)\sin(\omega_0 t)\cos(\omega_0 t) =$$
$$= \frac{I(t)}{2} + \frac{1}{2}I(t)\cos(2\omega_0 t) + \frac{1}{2}Q(t)\sin(2\omega_0 t); \quad (3.2)$$
$$Q_s(t) = s(t)\sin(\omega_0 t) = I(t)\sin(\omega_0 t)\cos(\omega_0 t) - Q(t)\sin^2(\omega_0 t) =$$
$$= -\frac{Q(t)}{2} + \frac{1}{2}I(t)\sin(2\omega_0 t) + \frac{1}{2}Q(t)\sin(2\omega_0 t). \quad (3.3)$$

Якщо гранична частота ФНЧ-1 менша за $2\omega_0 - \frac{\Delta\omega}{2}$, то останні дві складові у формулах (3.2), (3.3) будуть відфільтровані і сигнал на виході фільтра будуть дорівнювати:

$$I_s(t) = \frac{I(t)}{2}; \quad (3.4)$$
$$Q_s(t) = -\frac{Q(t)}{2}. \quad (3.5)$$



s(t) – вихідний вузькосмуговий сигнал; $I_s(t)$, $Q_s(t)$ – сигнали, отримані шляхом помноження на гармонічний сигнал опорної частоти; $I_l(t)$, $Q_l(t)$ – відновлені копії квадратурних складових; I'(t), Q'(t) –квадратурні складові, що визначають модулюючий сигнал; s'(t) – копія вихідного вузькосмугового сигналу; ФНЧ-1 – розділюючий фільтр нижніх частот; ФНЧ-2 – відновлюючий фільтр нижніх частот

Рисунок 3.4 – Дискретизація на основі виділення квадратурних складових.

3.4 Використовуване обладнання та програмне забезпечення

Для проведення вимірювань було використано платформу MTP300A компанії TESCOM Co., LTD. яка використовується як і за основним призначенням (у ролі тестера), так і для моделювання пристрою, що тестується (див. рис. 3.2). У платформі MTP300A присутні генератор та аналізатор безпровідних сигналів. Керування тестером відбувалось за допомогою інтерфейсу VISA (англ. Virtual Instrument Software Architecture), що є широко використовуваним інтерфейсом зв'язку для контролю програмованого тестувального обладнання. Команди передавались MTP300A з персонального комп'ютера за допомогою провідного інтерфейсу Ethernet.

Типова VISA-команда для MTP300A має наступний вигляд (приклад):

CONFigure:GPRF:MEAS:SPEC:SWEep:TIME [час у мкс]

Набір таких команд вручну на клавіатурі та виклик з командного рядка програми керування платформою MTP300A є дуже незручним та довготривалим способом тестування протоколу. Тому для спрощення управління процесом було створено додаток з графічним інтерфейсом користувача засобами пакету прикладних програм MATLAB. Зображення інтерфейсу додатка зображено на рисунку 3.5. Лістинг програмного коду додатка наведено у додатку А та додатку Б.

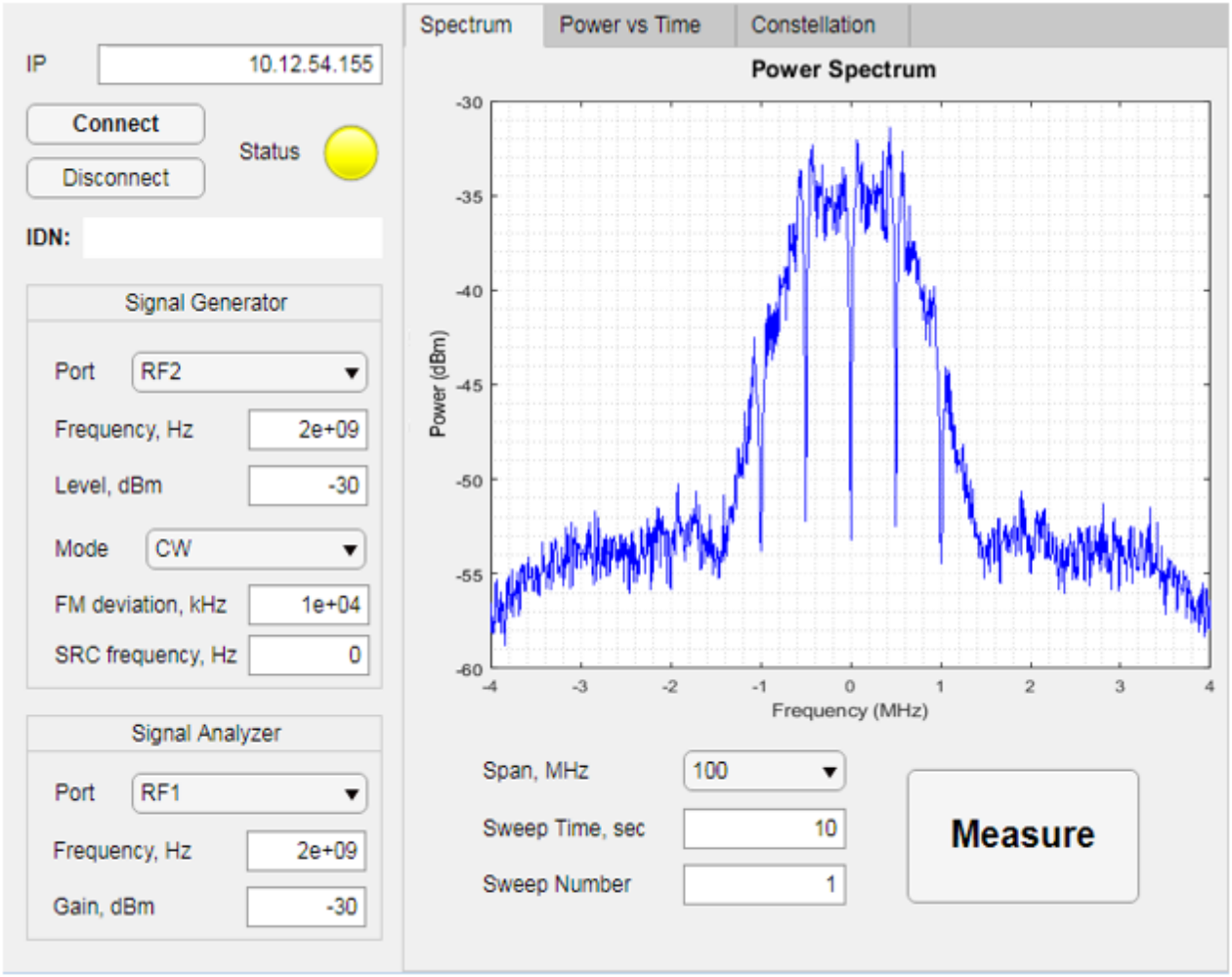


Рисунок 3.5 – Інтерфейс додатка

До всіх елементів інтерфейсу прив'язані callback-функції, що реалізують виклик все тих же VISA-команд, але додаток приховує їх виклик, а виводить лише результат виконання (у вигляді тексту, індикації, графіків залежностей).

Створений додаток виконує наступні функції:

– підключення до вимірювального пристрою (MTP300A);

– виведення статусу підключення (за допомогою світлового індикатора);

– задання параметрів генератора та аналізатора сигналів;

– завантаження відліків IQ з текстового файлу на вимірювальний пристрій;

– запуск вимірювання спектра потужності, залежності потужності від часу та відліків IQ сигналу.

– завантаження результатів вимірювань та виведення їх на графіках.

– обчислення значення EVM.

Кінцевим результатом вимірювання є графіки спектра потужності сигналу, залежності потужності від часу та діаграма сигнального сузір'я (англ. Constellation diagram) (див. п.п. 3.6.1-3.6.3).

3.5 Тестування ZigBee пристроїв

Під час несигнального тестування передавача важливими є три типи вимірювань: вимірювання спектральних характеристик сигналу отриманого від DUT, вимірювання потужності та вимірювання якості модуляції.

Інформація про спектр сигналу говорить про те, в якій частотній смузі працює передавач, і чи може він бути сумісний з іншими пристроями, що працюють в ISM-діапазоні [13].

Часова залежність потужності сигналу показує рівень сигналу, а також з цієї залежності можна винести інформацію про тривалість пакету даних.

Тим часом, якість модуляції обмежує відстань, на якій передавач буде надавати надійний радіозв'язок. Таким показником зокрема є величина вектора похибки EVM (англ. Error Vector Magnitude). У специфікації IEEE 802.15.4 висувається вимога, щоб значення цього параметра було меншим ніж 35% [6].

Обраховується EVM наступним чином.

Спочатку вимірюється N комплексних значень чіпів сигналу. Далі для кожного отриманого комплексного чіпа приймається рішення про те, яке значення було передано насправді. Ідеальне положення обраної комплексного чіпа (центр вікна прийняття рішення) представлено вектором (I_j, Q_j). Вектор похибки (δI_j, δQ_j) визначається як відстань від цього ідеального положення до фактичного положення виміряної точки, як показано на рис. 3.6.

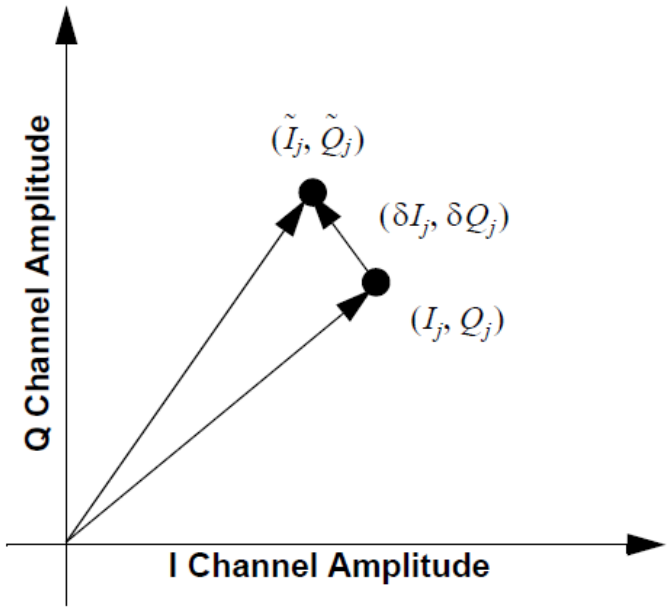


Рисунок 3.6 – До обчислення величини вектора похибок

З вище сказаного випливає, що вектор отриманого комплексного чіпа є сумою вектора ідеального положення та вектора похибки:

$$(\tilde{I}_j, \tilde{Q}_j) = (I_j, Q_j) + (\delta I_j, \delta Q_j).$$

Саме значення EVM обчислюється за формулою [6]:

$$\text{EVM} = \sqrt{\frac{\sum_{j=1}^{N}\left(\delta {I_j}^2 + \delta {Q_j}^2\right)}{N \cdot S^2}} \cdot 100\%, \tag{3.6}$$

де S – величина вектора ідеального положення;

N – кількість чіпів;

$(\delta I_j, \delta Q_j)$ – вектор похибки.

Для більш повної оцінки якості модуляції сигналу окрім середньоквадратичного значення, що обчислюється за формулою (3.6) часто використовуються мінімальне та максимальне відхилення від ідеального положення.

Для тестування приймача використовується значення коефіцієнту помилок пакетів PER (англ. Packet Error Rate). Цей показник являє собою відношення

кількості некоректно отриманих приймачем пакетів даних до кількості усіх надісланих пакетів.

Для обчислення PER виконується тестування приймача за схемою, що була наведена у пункті 3.2 цього розділу (див. рис. 3.2б). У процесі проводиться вимірювання I/Q відліків протягом заданого проміжку часу та їх відновлення за схемою описаною у 3.6.3 даного розділу.

Останнім етапом є спроба декодування отриманих пакетів даних за допомогою засобів MATLAB та підрахунок декодованих пакетів (тобто таких пакетів, преамбулу яких не вдалось виявити, або інформація, подана у їх заголовках, не відповідає отриманим даним).

3.6 Вимірювання радіочастотних характеристик передавача

3.6.1 Вимірювання амплітудного спектра сигналу

Спектр сигналу засобами тестера MTP300A обчислюється виходячи з виміряних значень I/Q за допомогою швидкого перетворення Фур'є. Алгоритм обчислення полягає у наступному.

Спектр обчислюється у діапазоні частот, що задається параметром *Span*. Тому на першому етапі, відбувається проріджения I/Q відліків сигналу (що дискретизований з частотою F_s) так, щоб фактична частота дискретизації дорівнювала параметру *Span* (проріджения з коефіцієнтом $r = F_s/Span$).

На наступному етапі відбувається обчислення швидкого перетворення Фур'є для відліків з номерами від 1 до N_p ($N_p = Span/RBW$, параметр *RBW* – роздільна ширина смуги), потім для відліків з номерами від $N_p/4$ до $5N_p/4$ і т.д. щоразу зміщуючись на $N_p/4$, поки не не буде отримано комплексний спектр для усіх відліків сигналу.

Відліки результуючого спектра залежно від параметра *Detector*, знаходяться по-різному. У випадку якщо *Detector* встановлено у значення *AVERAGE_MODE*, то

відліки кінцевого спектру знаходяться як середнє арифметичне відповідних відліків знайдених на попередньому етапі. У випадку якщо *Detector* встановлено у значення *MAXHOLD_MODE*, то у ролі відліків кінцевого спектру вибирається максимальні значення з відповідних.

Для знаходження амплітудного спектру виконується знаходження абсолютного значення кожного комплексного відліку результуючого спектра.

На рисунку 3.7 зображено спектр потужності пакету даних.

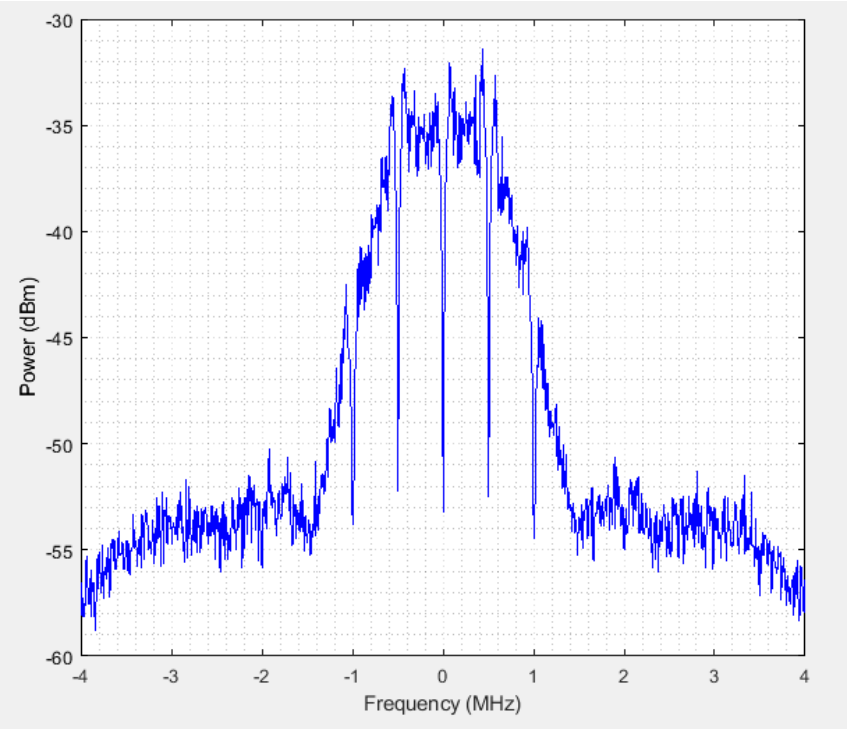


Рисунок 3.7 – Спектр потужності вимірюваного сигналу

3.6.2 Вимірювання залежності потужності від частоти

У платформі MTP300A обчислення потужності кожний часовий відлік t_j здійснюється з відомих значень I та Q складових:

$$P_j = 10\lg\left(I_j^2 + Q_j^2\right) - P_{0\,\text{дБм}} + G, \tag{3.5}$$

де P_j – потужність у момент часу t_j, дБм;

I_j – значення I складової у момент часу t_j;

Q_j – значення Q складової у момент часу t_j;

$P_{0\ дБм}$ – константа, що відповідає рівню 0 дБм, дБм;

G – коефіцієнт підсилення передавача, дБ.

Отримана при виміюванні залежність потужності від часу предачвлена на рис. 3.8.

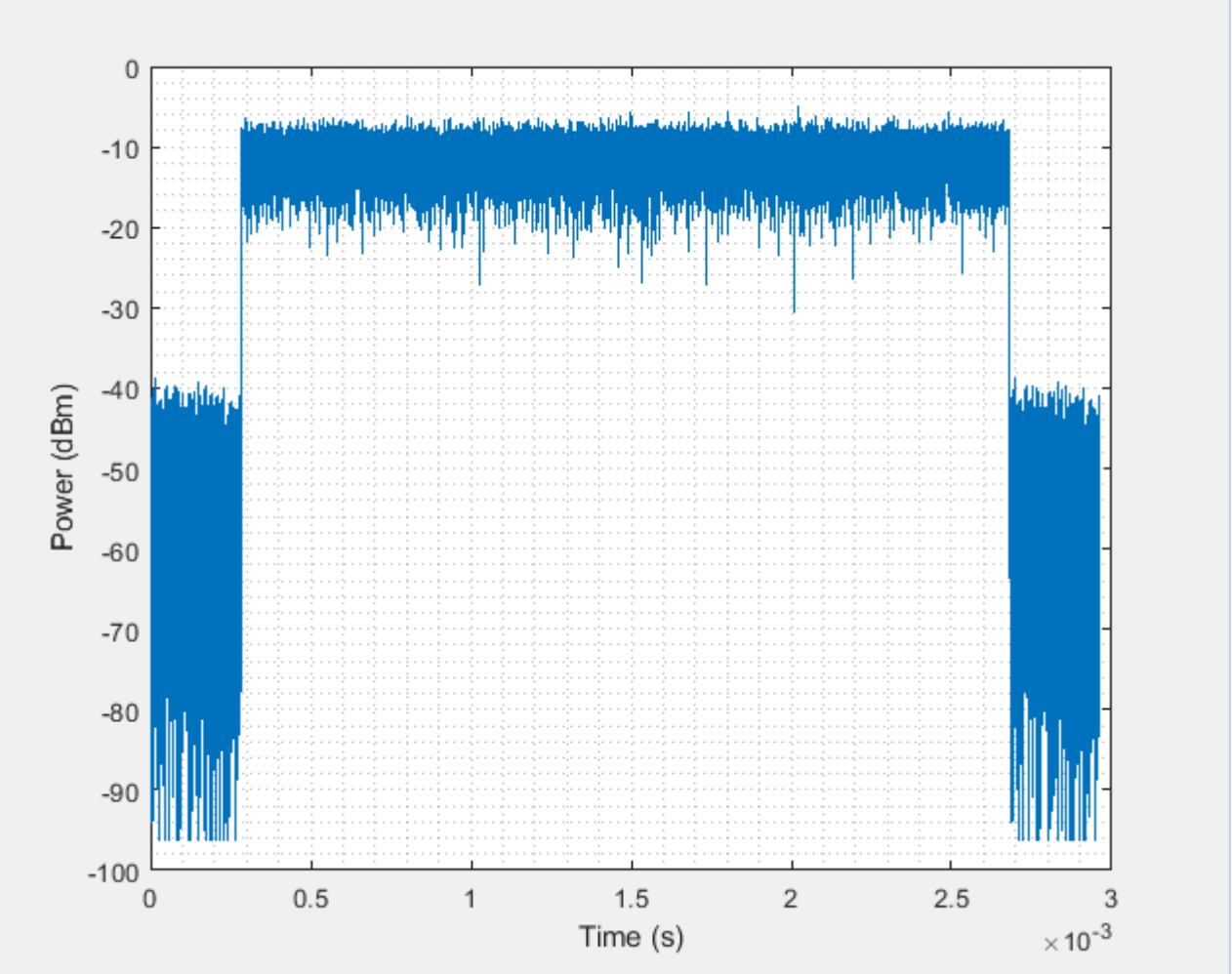


Рисунок 3.8 – Залежність потужності вимірюваного сигналу від часу

3.6.3 Відновлення сигналу та побудова сигального сузір'я

Так як сигнал при безпровідній передачі отримуються не напряму, а поширюється через неідеальний канал зв'язку, то отримані відліки I/Q не точно відповідають тим, що були відправлені. Сигнал потребує відновлення. У ході роботи для цього процесу використовувались засоби пакету MATLAB.

У процесі відновлення I/Q відліків відбуваються наступні етапи [14]:

– фільтрація;

– компенсація зсуву частоти носія;

– часова синхронізація;

– фазова синхронізація.

Фільтрація OQPSK-модульованого сигналу базується на використанні узгодженого фільтра, який максимізує відношення сигнал-шум отриманого сигналу. Передавальною характеристикою використаного фільтра є напівперіод функції синус. Саме такий вигляд має період квадратурних складових OQPSK-сигналу згідно специфікації IEEE 802.15.4 [6].

Компенсація частоти відбувається за алгоритмом, що використовує спектральні характреистики квадрату сигналу та швидке перетворення Фур'є. Спектр квадрата ідеального OQPSK-сигналу буде містити дві спектральні лінії віддалені від центральної частоти на величину символьної швидкості. Компенсація частоти неідеального сигналу відбувається шляхом усереднення та ділення на два частот спектральних піків [15].

Часова синхронізація символів – це процес наближення тактового сигналу приймача, який вирівняний як по фазі, так і по частоті з тактовим сигналом передавача та використовується для генерації даних. Оскільки виділяти частину каналу для передачі окремого тактового сигналу від передавача до приймача для цілей часової синхронізації неефективно, тактовий сигнал повинний бути знайдені з зашумлених отриманих вейвформ, які несуть дані. Для приймача, блок-схема

якого була описана у пункті 3.3 (також див. рис. 3.4) синхросигнал використовується для відбору необхідних відліків квадратурних складових [16].

Символьна синхронізація здійснюється за алгоритмом, що базується на основі фазового автопідлаштування частоти (англ. Phase locked loop, PLL), блок-схема (рис. 3.9) якого складається з таких компонентів:

– детектор похибки частової синхронізації (англ. timing error detector, TED);

– інтерполятор;

– контролер інтерполяції;

– циклічний фільтр.

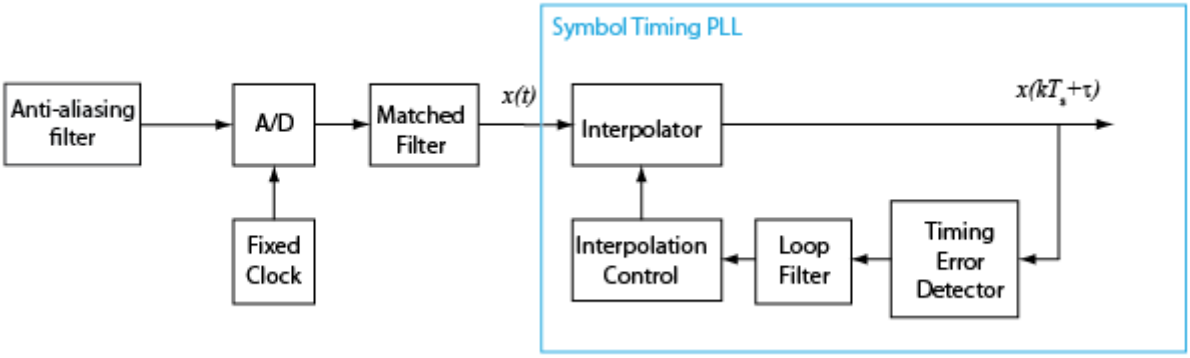


Рисунок 3.9 – Блок-схема алгоритму символьної синхронізації [16]

Після проведення усіх описаних вище етапів залишається невизначеність фази сигналу: сигнальне сузір'я сигналу може бути повернуте на кут кратний π/2. Фазова синхронізація відбувається шляхом пошуку у отриманій послідовності відліків I/Q преамбули OQPSK. Якщо ж преамбула не знайдена відбувається зміщення фази сигналу на величину π/2, та пошук повторюється заново. Поворот фази реалізується за допомогою зміни знаку всіх відліків синфазної і (або) квадратурної складової.

Сигнальне сузір'я виміряних відліків I/Q після процесу відновлення сигналу зображено на рис. 3.10.

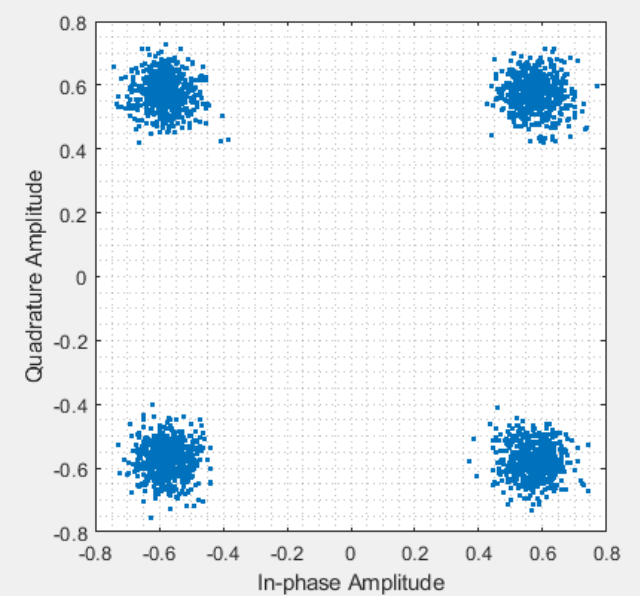


Рисунок 3.10 – Діаграма сигнального сузір'я

Також для отриманих відліків I/Q було обраховано значення EVM за допомого методу comm.EVM пакету MATLAB, що реалізує формулу (3.1). Обчислене середньоквадратичне значення становить 20,2%, що відповідає нормам описаним у специфікації IEEE 802.15.4. Максимальне значення EVM становить 42,5%, мінімальне – 2,8%.

3.7 Висновки до розділу 3

У даному розділі було проведено порівняння сигнального та несигнального методів тестування протоколів безпровідних мереж. Показано, що використання несигнального методу тестування може прискорити час тестування за рахунок зменшення часу з'єднання тестувального обладнання з пристроєм, який тестується, унаслідок підтримки пристроєм спеціального тестувального режиму.

Розроблено додаток з графічним інтерфейсом користувача для вимірювання радіочастотних характеристик пристроїв ZigBee за допомогою тестера MTP300A, що полегшив процес тестування.

Розроблено несигнальний метод тестування передавача ZigBee, який полягає у вимірі спектру сигналу від передавача, часової залежності потужності відліків I/Q та обчисленні основного параметру якості модуляції – EVM. Проведено моделювання передавач ZigBee за допомогою тестера MTP300A та виміряно його характеристики у процесі розробленого несигнального методу тестування. Характеристики дослідженого передавача відповідають вимогам описаним у стандарті IEEE 802.15.4. Середньоквадратичне значення величини вектора похибок не перевищує 35%. Зроблено висновок про відповідність фізичного рівня PHY дослідженого передавача протоколу IEEE 802.15.4/ZigBee.

ВИСНОВКИ

У даній роботі було досліджено основні можливості та характеристики стандарту ZigBee, такі як робочі частотні діапазони, швидкість передачі даних, типи пристроїв та види підтримуваних топологій. Особлива увага приділена mesh-топології. Показано, що характеристики ZigBee визначають сферу застосування протоколу – LR-WPAN мережі, такі безпровідні сенсорні мережі.

У результаті вивчення архітектуру стеку ZigBee зроблено висновок про те, що саме функції основних рівнів стеку (фізичного рівня, рівня керування доступом до середовища, мережевого рівня та прикладного рівня) та їх взаємодія забезпечує ефективну комунікацію пристроїв у мережі ZigBee і визначають її надійність.

Проведено порівняльний аналіз сигнального та несигнального методів тестування безпровідних протоколів. Використання несигнального методу тестування може прискорити час тестування за рахунок зменшення часу з'єднання тестувального обладнання з пристроєм, який тестується, унаслідок підтримки пристроєм спеціального тестувального режиму.

Розроблено додаток з графічним інтерфейсом користувача для вимірювання радіочастотних характеристик пристроїв ZigBee за допомогою тестера MTP300A, що полегшив процес тестування.

Розроблено несигнальний метод тестування передавача ZigBee, який полягає у вимірі спектру сигналу від передавача, часової залежності потужності відліків I/Q та обчисленні основного параметру якості модуляції – EVM. Проведено моделювання передавач ZigBee за допомогою тестера MTP300A та виміряно його характеристики у процесі розробленого несигнального методу тестування. Зроблено висновок про відповідність фізичного рівня PHY дослідженого передавача протоколу IEEE 802.15.4/ZigBee.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. 5G. Пятое поколение мобильной связи. – Режим доступу до ресурсу : https://www.tadviser.ru/index.php/%D0%A1%D1%82%D0%B0%D1%82%D1%8C%D1%8F:5G_(%D0%BF%D1%8F%D1%82%D0%BE%D0%B5_%D0%BF%D0%BE%D0%BA%D0%BE%D0%BB%D0%B5%D0%BD%D0%B8%D0%B5_%D0%BC%D0%BE%D0%B1%D0%B8%D0%BB%D1%8C%D0%BD%D0%BE%D0%B9_%D1%81%D0%B2%D1%8F%D0%B7%D0%B8)#.D0.A6.D0.B5.D0.BB.D1.8C_.D1.81.D0.BE.D0.B7.D0.B4.D0.B0.D0.BD.D0.B8.D1.8F_.D0.B8_.D0.BD.D0.B0.D0.B7.D0.BD.D0.B0.D1.87.D0.B5.D0.BD.D0.B8.D0.B5_.D1.81.D0.B5.D1.82.D0.B5.D0.B9_5G
2. Mi Smart Sensor Set. – Режим доступу до ресурсу : https://www.mi.com/ru/mi-smart-sensor-set/
3. ZigBee Specification – Режим доступу до ресурсу : https://zigbeealliance.org/wp-content/uploads/2019/11/docs-05-3474-21-0csg-zigbee-specification.pdf
4. Офіційний сайт Zigbee Alliance. – Режим доступу до ресурсу : https://zigbeealliance.org/
5. Farahani, S., 2011. Zigbee Wireless Networks And Transceivers. Elsevier Science.
6. IEEE Standard for Low-Rate Wireless Networks," in IEEE Std 802.15.4-2015 (Revision of IEEE Std 802.15.4-2011) , vol., no., pp.1-709, 22 April 2016, doi: 10.1109/IEEESTD.2016.7460875.
7. M. Hillman, "An Overview of ZigBee Networks," MWR InfoSecurity.
8. What is ZIGBEE Technology in IoT - Architecture, Network Topologies, Applications. – Режим доступу до ресурсу : https://electricalfundablog.com/zigbee-technology-architecture/
9. "Getting Started with ZigBee and IEEE 802.15.4," Daintree Networks.
10. Nonsignaling technique improves RF test. – Режим доступу до ресурсу : https://www.edn.com/nonsignaling-technique-improves-rf-test/

11. Cellular device manufacturing. – Режим доступу до ресурсу : https://www.rohde-schwarz.com/ua/solutions/test-and-measurement/wireless-communication/mobile-device-testing/manufacturing-and-service/cellular-device-manufacturing-/cellular-device-manufacturing_231302.html
12. К. Феер. Беспрводная цифровая связь: методы модуляции. Пер. с англ.//Под. Ред. Журавлёва. — М.: Радио и связь, 2000. — 520 с.
13. The Basics of ZigBee Transmitter Testing. – Режим доступу до ресурсу : ftp://ftp.ni.com/evaluation/rf/The_Basics_of_ZigBee_Transmitter_Testing.pdf
14. Recovery of IEEE 802.15.4 OQPSK Signals. – Режим доступу до ресурсу : https://www.mathworks.com/help/comm/examples/recovery-of-ieee-802-15-4-oqpsk-signals.html
15. Compensate for frequency offset for PAM, PSK, or QAM – Режим доступу до ресурсу: https://www.mathworks.com/help/comm/ref/comm.coarsefrequencycompensator-system-object.html
16. Correct symbol timing clock skew. – Режим доступу до ресурсу : https://www.mathworks.com/help/comm/ref/comm.symbolsynchronizer-system-object.html

Додаток А

Лістинг коду головного файлу додатка керування платформою MTP300A

```
classdef visa_gui < matlab.apps.AppBase

    properties (Access = private)
        Instrument
        ConnectionStatus = 'No connection';
        IP
        Address
        visaObj
        gen_port
        gen_freq
        gen_level
        gen_mode
        FM_dev
        SRC_freq
        meas_port
        meas_freq
        gain
        triger = 0;
        span_kHz
        sweep_time
        sweep_num
        SW_time
        S_time
    end

    methods (Access = private)

        function configureGen(app)
            wqc(app.visaObj, 'ROUTe:GEN:PORT', app.gen_port);
            wqc(app.visaObj, 'SOURce:GEN:RFSetting:LEVel', app.gen_level);
            wqc(app.visaObj, 'SOURce:GEN:RFSetting:FREQuency', app.gen_freq);
            wqc(app.visaObj, 'SOURce:GPRF:GEN:MODE', app.gen_mode);
            wqc(app.visaObj, 'SOURce:GPRF:GEN:FM:DEViation', app.FM_dev);
            wqc(app.visaObj, 'SOURce:GPRF:GEN:FM:SRCFrequency', app.SRC_freq);
        end

        function configureMeas(app)
            wqc(app.visaObj, 'ROUTe:MEAS:PORT', app.meas_port);
            wqc(app.visaObj, 'CONFigure:MEAS:RFSetting:FREQuency', app.meas_freq);
            wqc(app.visaObj, 'CONFigure:MEAS:RFSetting:REFLevel', app.gain);
        end
```

```
function configureSpec(app)
        wqc(app.visaObj, 'CONFigure:GPRF:MEAS:TRIGger', app.triger);
        wqc(app.visaObj, 'CONFigure:GPRF:MEAS:SPEC:SPAN', app.span_kHz);
        wqc(app.visaObj, 'CONFigure:GPRF:MEAS:SPEC:SWEep:TIME', app.sweep_time);
        wqc(app.visaObj, 'CONFigure:GPRF:MEAS:SPEC:SWEep:NUM', app.sweep_num);
    end

    function configurePVT(app)
        wqc(app.visaObj, 'CONFigure:GPRF:MEAS:PVTime:SWTime', app.SW_time);
        wqc(app.visaObj, 'CONFigure:GPRF:MEAS:PVTime:STime', app.S_time);
%           fprintf(app.visaObj, 'CONFigure:GPRF:MEAS:PVTime:SWTime %.1f', app.SW_time);
%           fprintf(app.visaObj, 'CONFigure:GPRF:MEAS:PVTime:STime %.1f', app.S_time);
    end

    function initializeParameters(app)
        app.IP = app.IPEditField.Value;
        app.Address = ['TCPIP0::' app.IP '::inst0::INSTR'];
        app.gen_port = app.GenPortDropDown.Value;
        app.gen_freq = app.GenFrequencyHzEditField.Value;
        app.gen_level = app.LeveldBmEditField.Value;
        app.gen_mode = app.ModeDropDown.Value;
        app.FM_dev = app.FMdeviationkHzEditField.Value;
        app.SRC_freq = app.SRCfrequencyHzEditField.Value;
        app.meas_port = app.MeasPortDropDown.Value;
        app.meas_freq = app.MeasFrequencyHzEditField.Value;
        app.gain = app.GaindBmEditField.Value;
        app.triger = 0;
        app.span_kHz = str2double(app.SpanMHz.Value)*1000;
        app.sweep_time = app.SweepTimesecEditField.Value;
        app.sweep_num = app.SweepNumberEditField.Value;
        app.SW_time = app.SWTimeusEditField.Value;
        app.S_time = app.STimeusEditField.Value;
    end

    function initializeUIAxes(app)
        app.UIAxes.XGrid = 'on';
        app.UIAxes.YGrid = 'on';
    end
end

methods (Access = private)

    % Code that executes after component creation
    function startupFcn(app)
        initializeParameters(app);
    end
```

```
% Button pushed function: ConnectButton
    function ConnectButtonPushed(app, event)
        app.IP = app.IPEditField.Value;
        app.Address = ['TCPIP0::' app.IP '::inst0::INSTR'];

        app.Instrument = instrfind('Type', 'visa-tcpip', 'RsrcName', app.Address, 'Tag', '');
        if isempty(app.Instrument)
            app.visaObj = visa('AGILENT', app.Address);
        else
            fclose(app.Instrument);
            app.visaObj = app.Instrument(1);
        end
        set(app.visaObj,'InputBufferSize', 100000);
        % To connect the visa object to the instrument:
        fopen(app.visaObj);
        clrdevice(app.visaObj);
        app.visaObj.EOSMode = 'write';

        % IDN
        app.IDNRespLabel.Text = query(app.visaObj, '*IDN?');

        % Change status
        app.ConnectionStatus = 'OK';
        app.StatusLamp.Color = [0.00,1.00,0.00]; % Green

        configureGen(app);
        configureMeas(app);
        configureSpec(app);
        configurePVT(app);
    end

    % Button pushed function: DisconnectButton
    function DisconnectButtonPushed(app, event)
        fclose(app.visaObj);
        app.IDNRespLabel.Text = '';
        app.ConnectionStatus = 'No connection';
        app.StatusLamp.Color = [1.00,1.00,0.00]; % Yellow
    end

    % Value changed function: GenPortDropDown
    function GenPortDropDownValueChanged(app, event)
        app.gen_port = app.GenPortDropDown.Value;
        wqc(app.visaObj, 'ROUTe:GEN:PORT', app.gen_port);
    end
```

```
% Button pushed function: SpecMeasButton
    function SpecMeasButtonPushed(app, event)
        wqc(app.visaObj, 'SOURce:GEN:STATe', 'ON');
        spec_resp = query(app.visaObj, 'READ:GPRF:MEAS:SPEC?');
        wqc(app.visaObj, 'SOURce:GEN:STATe', 'OFF');
        spec = str2double(split(spec_resp,','));

        fstart_MHz = app.meas_freq*1e-6 - app.span_kHz*5e-4;
        fstop_MHz = app.meas_freq*1e-6 + app.span_kHz*5e-4;
        f_MHz = linspace(fstart_MHz, fstop_MHz, length(spec));

        plot(app.UIAxes, f_MHz, spec);
    end

    % Close request function: UIFigure
    function UIFigureCloseRequest(app, event)
        fclose(app.visaObj);
        delete(app)
    end

    % Value changed function: GenFrequencyHzEditField
    function GenFrequencyHzEditFieldValueChanged(app, event)
        app.gen_freq = app.GenFrequencyHzEditField.Value;
        wqc(app.visaObj, 'SOURce:GEN:RFSetting:FREQuency', app.gen_freq);
    end

    % Value changed function: LeveldBmEditField
    function LeveldBmEditFieldValueChanged(app, event)
        app.gen_level = app.LeveldBmEditField.Value;
        wqc(app.visaObj, 'SOURce:GEN:RFSetting:LEVel', app.gen_level);
    end

    % Value changed function: ModeDropDown
    function ModeDropDownValueChanged(app, event)
        app.gen_mode = app.ModeDropDown.Value;
        wqc(app.visaObj, 'SOURce:GPRF:GEN:MODE', app.gen_mode);
    end

    % Value changed function: SRCfrequencyHzEditField
    function SRCfrequencyHzEditFieldValueChanged(app, event)
        app.SRC_freq = app.SRCfrequencyHzEditField.Value;
        wqc(app.visaObj, 'SOURce:GPRF:GEN:FM:DEViation', app.FM_dev);
    end

    % Value changed function: FMdeviationkHzEditField
    function FMdeviationkHzEditFieldValueChanged(app, event)
        app.FM_dev = app.FMdeviationkHzEditField.Value;
        wqc(app.visaObj, 'SOURce:GPRF:GEN:FM:SRCFrequency', app.SRC_freq);
    end
```

```
% Value changed function: MeasPortDropDown
    function MeasPortDropDownValueChanged(app, event)
        app.meas_port = app.MeasPortDropDown.Value;
        wqc(app.visaObj, 'ROUTe:MEAS:PORT', app.meas_port);
    end

    % Value changed function: MeasFrequencyHzEditField
    function MeasFrequencyHzEditFieldValueChanged(app, event)
        app.meas_freq = app.MeasFrequencyHzEditField.Value;
        wqc(app.visaObj, 'CONFigure:MEAS:RFSetting:FREQuency', app.meas_freq);
    end

    % Value changed function: GaindBmEditField
    function GaindBmEditFieldValueChanged(app, event)
        app.gain = app.GaindBmEditField.Value;
        wqc(app.visaObj, 'CONFigure:MEAS:RFSetting:REFLevel', app.gain);
    end

    % Value changed function: SweepTimesecEditField
    function SweepTimesecEditFieldValueChanged(app, event)
        app.sweep_time = app.SweepTimesecEditField.Value;
        wqc(app.visaObj, 'CONFigure:GPRF:MEAS:SPEC:SWEep:TIME', app.sweep_time);
    end

    % Value changed function: SweepNumberEditField
    function SweepNumberEditFieldValueChanged(app, event)
        app.sweep_num = app.SweepNumberEditField.Value;
        wqc(app.visaObj, 'CONFigure:GPRF:MEAS:SPEC:SWEep:NUM', app.sweep_num);
    end

    % Value changed function: SpanMHz
    function SpanMHzValueChanged(app, event)
        value = app.SpanMHz.Value;
        app.span_kHz = str2double(value)*1000;
        wqc(app.visaObj, 'CONFigure:GPRF:MEAS:SPEC:SPAN', app.span_kHz);
    end

    % Button pushed function: PVTMeasButton
    function PVTMeasButtonPushed(app, event)
        wqc(app.visaObj, 'SOURce:GEN:STATe', 'ON');
        pvt_resp = query(app.visaObj, 'READ:GPRF:MEAS:PVTime?');
        wqc(app.visaObj, 'SOURce:GEN:STATe', 'OFF');
        pvt = str2double(split(pvt_resp,','));
        plot(app.PVTUIAxes, pvt);
    end

    % Value changed function: SWTimeusEditField
    function SWTimeusEditFieldValueChanged(app, event)
        app.SW_time = app.SWTimeusEditField.Value;
        wqc(app.visaObj, 'CONFigure:GPRF:MEAS:PVTime:SWTime', app.SW_time);
    end
```

```
        % Value changed function: STimeusEditField
        function STimeusEditFieldValueChanged(app, event)
            app.S_time = app.STimeusEditField.Value;
            wqc(app.visaObj, 'CONFigure:GPRF:MEAS:PVTime:STime', app.S_time);
        end
    end
end
```

Додаток Б

Лістинг коду залежних функцій головного файлу додатка керування платформою MTP300A

```
function resp = wqc(instr, command, val, verbose)
%
% WQC: Write. Query. Check. (compares written VAL to received RESP)
%
   verbose_default = 0;
   if nargin < 4
      verbose = verbose_default;
   end
   if isa(val, 'numeric')
      valStr = num2str(val);
   elseif isa(val, 'char')
      valStr = val;
   else
      error('INVALID DATATYPE!');
   end
   valStr = strip(valStr);
   fprintf(instr, [command ' ' valStr]);

   resp = query(instr, [command '?']);
   if isempty(resp)
      warning('There is no responce for %s?', command);
   else
      resp = strip(resp);
      if ~strcmp(resp, valStr) && all(str2double(resp)~=val)
         fprintf('\nVISA I/O\n');
         allerr(instr);
         warning("%s\n  Can't set %s.\n  %s is used instead.\n", ...
            command, valStr, resp);
      elseif verbose
         fprintf("OK. %s %s\n", command, valStr);
      end
   end
end

function idn(instr)
   global v
   if nargin == 0
      instr = v;
   end
   resp = query(instr, '*IDN?');
   disp(resp)
end
```