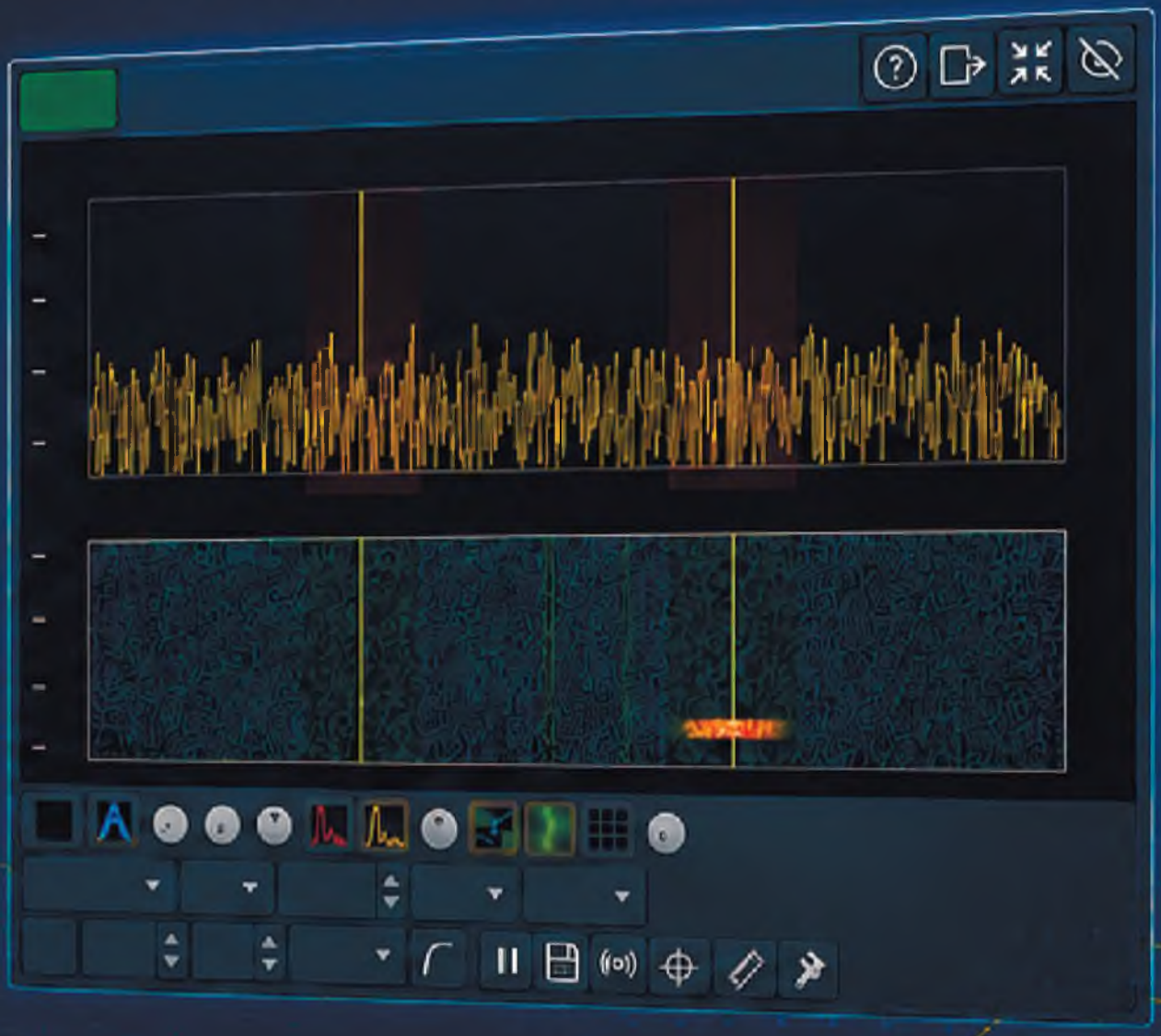
І. Ш. НЕВЛЮДОВ, О. І. ФИЛИПЕНКО, С. П. НОВОСЕЛОВ, О. В. СИЧОВА

КІБЕРБЕЗПЕКА

АВТОМАТИЗОВАНИХ СИСТЕМ УПРАВЛІННЯ ТЕХНОЛОГІЧНИМИ ПРОЦЕСАМИ



Навчальний посібник

ЧАСТИНА 1

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ХАРКІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ РАДІОЕЛЕКТРОНІКИ

І. Ш. Невлюдов, О. І. Филипенко, С. П. Новоселов, О. В. Сичова

КІБЕРБЕЗПЕКА АВТОМАТИЗОВАНИХ СИСТЕМ УПРАВЛІННЯ ТЕХНОЛОГІЧНИМИ ПРОЦЕСАМИ

Частина І

Навчальний посібник

Харків, Вид-во Іванченка І. С., 2026

УДК 004.056:681.518
Н40

Рекомендовано Вченою радою
Харківського національного університету радіоелектроніки
(протокол № 8/5 від 29.05.2026 р.)

Н40 Кібербезпека автоматизованих систем управління технологічними процесами. Частина І : Навчальний посібник / І. Ш. Невлюдов, О. І. Филипенко, С. П. Новоселов, О. В. Сичова. – Харків: Видавництво Іванченка І. С., 2026. – 240 с.

Cybersecurity of automated control systems of technological processes. Part I: Textbook / I. Nevliudov, O. Filipenko, S. Novoselov, O. Sychova. – Kharkiv: Ivanchenko I. Publishing House, 2026. – 247 p.

ISBN 978-617-8742-04-1
DOI: 10.30837/978-617-8742-04-1

Навчальний посібник присвячено питанням кібербезпеки автоматизованих систем управління технологічними процесами та промислового Інтернету речей. У виданні розглянуто архітектуру сучасних промислових систем, принципи їх побудови та функціонування, а також ключові підходи до забезпечення їх захищеності від кіберзагроз.

Навчальний посібник призначено для підготовки здобувачів першого (бакалаврського), другого (магістерського) рівнів вищої освіти спеціальності G7 Автоматизація, комп'ютерно-інтегровані технології та робототехніка. Може бути корисним здобувачам вищої освіти технічних спеціальностей, які вивчають дисципліни, пов’язані з АСУТП, IIoT та захистом інформації, а також для всіх, хто цікавиться питаннями кібербезпеки промислових систем і сучасними методами виявлення кіберзагроз.

УДК 004.056:681.518

Рецензенти:
– Снігуров А. В., канд техн. наук, доцент, декан факультету кібербезпеки ХНУРЕ;
– Цимбал О. М., д-р техн. наук, професор, професор кафедри комп’ютерно-інтегрованих технологій, автоматизації, робототехніки та безпекової інженерії, ХНУРЕ;
– Карпов Г. В., канд. техн. наук, доцент, начальник технічного відділу ТОВ «Харківський завод спеціальних машин».

ISBN 978-617-8742-04-1
DOI: 10.30837/978-617-8742-04-1

© І. Ш. Невлюдов, О. І. Филипенко, С. П. Новоселов, О. В. Сичова, 2026.

ЗМІСТ

Скорочення та умовні познаки 7
Вступ 10
1 Автоматизована система управління технологічними процесами з точки зору кібербезпеки 13
1.1 Використання ПЛК для створення автоматизованої системи управління технологічними процесами 13
1.2 Архітектура АСУТП 18
1.2.1 Функція спостереження 19
1.2.2 Функція моніторингу 20
1.2.3 Контрольна функція 22
1.2.4 Програмовані логічні контролери 23
1.2.5 Панелі оператора 25
1.2.6 Система диспетчерського контролю та збору даних 25
1.2.7 Розподілена система управління 26
1.2.8 Система безпеки 27
1.3 Модель Пердью для промислових систем управління 28
1.3.1 Корпоративна зона 29
1.3.2 Рівень 5 – Корпоративна мережа 31
1.3.3 Рівень 4 – Бізнес-планування та логістика виробництва 31
1.3.4 Промислова демілітаризована зона 32
1.3.5 Виробнича зона 33
1.3.6 Рівень 3 – управління виробництвом 33
1.3.7 Рівень 2 – зона наглядового контролю 34
1.3.8 Рівень 1 – базовий контроль 35
1.4 Контрольні запитання та завдання 35
2 Огляд можливих сценаріїв втручання у дії АСУТП 36
2.1 Спрощена структура мережі Пердью 36
2.2 Атака на технологічний сегмент через суміжні системи 37
2.2.1 EDR система 38
2.2.2 Мережева пісочниця 42
2.2.3 Система класу SIEM 43
2.3 Контрольні запитання та завдання 43
3 Протоколи промислового інтернету речей 44
3.1 Рівні архітектури IIOT 44
3.2 Основні протоколи організації зв’язку в IIOT 49
3.3 Особливості та принцип використання протоколу MQTT 58

3.3.1 Основні характеристики протоколу MQTT 58
3.3.2 Формат повідомлень за протоколом MQTT 61
3.4 Розгортання брокера MQTT Eclipse Mosquitto 71
3.4.1 Основні відомості про Eclipse Mosquitto 71
3.4.2 Порядок встановлення брокера Eclipse Mosquitto на ОС Linux 72
3.5 Використання графічної платформи Cedalo для управління брокером Eclipse Mosquitto 78
3.6 Безпека Інтернету речей 90
3.7 Класифікація загроз промислового Інтернету речей 93
3.7.1 Відмови або вихід з ладу елементів системи 96
3.7.2 Умисні дії 96
3.7.3 Правове порушення 98
3.7.4 Ненавмисне пошкодження елементів системи 99
3.7.5 Фізична атака 99
3.7.6 Вимкнення пристроїв 100
3.7.7 Підслуховування, перехоплення, крадіжка інформації 100
3.7.8 Катастрофа 102
3.8 Контрольні запитання та завдання 102
4 Методика аналізу трафіку промислових мереж з метою виявлення непередбачуваного втручання 103
4.1 Протокол Modbus. Базові поняття 103
4.2 Організація обміну даними за протоколом Modbus 105
4.3 Різновиди протоколу Modbus 107
4.3.1 Modbus RTU (Remote Terminal Unit) 107
4.3.2 Modbus ASCII 109
4.3.3 Modbus TCP 109
4.4 Регістри та функції протоколу Modbus 110
4.4.1 Стандартна адресація регістрів Modbus 110
4.4.2 Опис функції читання вихідних контактів (дискретних виходів) (01) 112
4.4.3 Функція читання дискретних входів (02) 115
4.4.4 Функція читання регістрів зберігання (03) 117
4.4.5 Запис одного регістра зберігання (06) 119
4.4.6 Запис декількох регістрів зберігання (0x10) 120
4.4.7 Зміна стану одного вихідного контакту (05) 123
4.4.8 Зміна стану декількох вихідних контактів (0F) 123
4.4.9 Читання стану вхідних регістрів (04) 125

4.5 Огляд інструментів для аналізу мережевого трафіку на прикладі програмного засобу Wireshark 127
4.6.1 Підготовка до проведення експерименту 131
4.6.2 Програма ModRSsim2 132
4.6.3 Програмний симулятор «Майстер мережі Modbus TCP/IP» 137
4.6.4 Проведення експерименту з аналізу пакетів протоколу Modbus 138
4.7 Контрольні запитання та завдання 147
5. Моніторинг радіочастотного середовища для виявлення кіберзагроз в АСУТП 148
5.1 Особливість бездротових сегментів АСУТП з точки зору кібербезпеки 148
5.2 Методи та технології моніторингу радіочастотного середовища для виявлення кіберзагроз в АСУТП 153
5.2.1 Класифікація радіочастотних атак 153
5.2.2 Спеціалізовані системи виявлення вторгнень (WIDS) 154
5.2.3 Промислові системи радіомоніторингу 155
5.2.4 Програмно-визначена радіосистема (SDR) 156
5.3 Спеціалізовані пристрої для аудиту безпеки бездротових мереж АСУТП та методи протидії 158
5.3.1 Платформа для атак на бездротові мережі WiFi Pineapple 158
5.3.2 Пристрій для атак відмови в обслуговуванні WiFi Deauther 161
5.3.3 Методи виявлення атак із застосуванням WiFi Pineapple 163
5.3.4 Методи виявлення WiFi Deauther 164
5.3.5 Методи протидії та захисту 165
5.4 Застосування технології SDR для моніторингу радіочастотного середовища з метою виявлення кібератак 167
5.4.1 Реалізація програмно-апаратної системи радіомоніторингу на базі SDR 167
5.4.2 Апаратне забезпечення для проведення моніторингу радіочастото середовища 168
5.4.3 Методи виявлення кіберзагроз за допомогою програмно-апаратної системи SDR 173
5.5 Практична реалізація методу виявлення незареєстрованого передавача LoRa WAN 174
5.5.1 Опис лабораторного макету, що використовується для проведення експериментального дослідження 174
5.5.2 Принцип організації обміну повідомленнями в мережі IoT за допомогою модулів LoRa 184

5.5.3 Комп'ютерне моделювання та вибір оптимальних параметрів шлюзу 191
5.5.4 Опис методу виявлення і візуальної ідентифікації LoRa-сигналу 200
5.5.5 Популярні інструменти для декодування прийнятого LoRa-сигналу 212
5.5.6 Декодування прийнятого LoRa-сигналу за допомогою SDRangel 213
5.5.7 Декодування прийнятого LoRa-сигналу за допомогою GNU Radio 222
5.6 Контрольні запитання та завдання 233
Перелік джерел посилань 234

СКОРОЧЕННЯ ТА УМОВНІ ПОЗНАКИ

АРМ – автоматизоване робоче місце;
АСУТП – автоматизована система управління технологічними процесами;
АСУ – автоматизована система управління;
АЦП – аналого-цифровий перетворювач;
ДМЗ – демілітаризована зона;
КМПД – корпоративна мережа передачі даних;
МНК – моноблочний контролер;
ОС – операційна система;
ПЗ – програмне забезпечення;
ПК – персональний комп’ютер;
ПЛК – програмований логічний контролер;
ТП – технологічний процес;
ЦАП – цифро-аналоговий перетворювач;
ЧПУ – числове програмне управління;
ADU – Application Data Unit;
AI – Artificial Intelligence;
AMQP – Advanced Message Queuing Protocol;
ARP – Address Resolution Protocol;
BLE – Bluetooth Low Energy;
BW – Bandwidth;
CLI – Command Line Interface;
CoAP – Constrained Application Protocol;
CR – Coding Rate;
CSS – Chirp Spread Spectrum;
DCS – Distributed Control System;
DDS – Data Distribution Service;
DMZ – Demilitarized Zone;
EDR – Endpoint Detection and Response;
EPP – Endpoint Protection Platform;
ERP – Enterprise Resource Planning;
ETL – Extract, Transform, Load;
GUI – Graphical User Interface;
HMI – Human Machine Interface;

HTTP – HyperText Transfer Protocol;
ICS – Industrial Control Systems;
IDMZ – Industrial Demilitarized Zone;
IDS – Intrusion Detection System;
IIoT – Industrial Internet of Things;
IoT – Internet of Things;
IT – Information Technology;
JMS – Java Message Service;
MES – Manufacturing Execution System;
MQ – Message Queue;
MQTT – Message Queuing Telemetry Transport;
NDR – Network Detection and Response;
OFDM – Orthogonal Frequency-Division Multiplexing;
OLED – Organic Light-Emitting Diode;
OMG – Object Management Group;
OT – Operational Technology;
PDU – Protocol Data Unit;
PID – Proportional-Integral-Derivative;
PLC – Programmable Logic Controller;
PSK – Pre-Shared Key;
RPK – Raw Public Key;
RTU – Remote Terminal Unit;
SAE – Simultaneous Authentication of Equals;
SAP – Systems, Applications and Products;
SAP MM – SAP Materials Management;
SAP PM – SAP Plant Maintenance;
SAP PP – SAP Production Planning;
SASL – Simple Authentication and Security Layer;
SCADA – Supervisory Control and Data Acquisition;
SDR – Software-Defined Radio;
SF – Spreading Factor;
SFD – Start Frame Delimiter;
SIEM – Security Information and Event Management;
SIS – Safety Instrumented System;
SPI – Serial Peripheral Interface;

SSID – Service Set Identifier;
SSL – Secure Sockets Layer;
TCP – Transmission Control Protocol;
TLS – Transport Layer Security;
URH – Universal Radio Hacker;
VFD – Variable Frequency Drive;
WNOC – Wireless network-on-chip;
XML – eXtensible Markup Language;
XMPP – Extensible Messaging and Presence Protocol.

ВСТУП

Промислові системи керування (англ. Industrial Control Systems, ICS) є основою функціонування критичної інфраструктури в таких галузях, як енергетика, водопостачання, транспорт, виробництво та інші. В сучасних умовах кібератаки дедалі частіше спрямовуються саме на ці системи через їх ключову роль у забезпеченні життєдіяльності суспільства. Порушення функціонування ICS може призвести до значних економічних втрат, зниження рівня національної безпеки та дестабілізації суспільних процесів.

Типова архітектура ICS є багаторівневою та включає такі компоненти, як системи диспетчерського керування і збору даних (SCADA) та розподілені системи керування (DCS). Порівняно з традиційними інформаційними системами, промислові системи керування безпосередньо взаємодіють із фізичними процесами, що зумовлює їх особливу чутливість до кіберінцидентів і потенційно серйозні наслідки їх реалізації.

Однією з найбільш поширених моделей опису архітектури ICS є модель Пердью, яка визначає ієрархічну структуру промислових мереж та дозволяє розмежувати два ключові домени: інформаційні технології (IT) та операційні технології (OT). Операційні технології забезпечують керування фізичними процесами, обробку подій, моніторинг і контроль обладнання, тоді як інформаційні технології орієнтовані на підтримку бізнес-процесів, обробку даних і корпоративну інфраструктуру.

Важливою особливістю є відмінність у пріоритетах безпеки для цих доменів. Для IT-систем ключовими є конфіденційність, цілісність і доступність інформації, тоді як для OT-систем першочергового значення набувають безпека функціонування, надійність, безперервність роботи та відмовостійкість.

За останні роки у світі зафіксовано значне зростання кількості кібератак на промислові системи керування. Серед найбільш відомих прикладів можна виділити атаки із застосуванням програм-вимагачів на трубопровідну інфраструктуру, шкідливе програмне забезпечення, спрямоване на системи безпеки технологічних процесів, а також кібератаки на енергетичні системи, що призводили до масштабних відключень електроенергії. Окреме місце займають складні цільові атаки на кшталт Stuxnet, які продемонстрували можливість впливу на фізичні процеси через кіберпростір.

Історично системи ICS функціонували як ізольовані середовища, що використовували спеціалізоване апаратне та програмне забезпечення. Однак сучасні тенденції цифровізації, зокрема впровадження концепцій Індустрії 4.0, віддаленого доступу, хмарних технологій і аналітики даних, призвели до інтеграції ICS з корпоративними IT-системами. Така конвергенція забезпечує нові можливості для підвищення ефективності виробництва, але одночасно розширює поверхню атаки та створює додаткові кіберризики.

Забезпечення кібербезпеки в середовищі ICS ускладняється низкою факторів, серед яких наявність застарілого обладнання, обмеження на оновлення програмного забезпечення, використання пропрієтарних протоколів та високі вимоги до безперервності технологічних процесів. У багатьох випадках традиційні методи захисту, характерні для IT-систем, не можуть бути безпосередньо застосовані в OT-середовищі.

У зв’язку з цим особливої актуальності набувають методи виявлення кіберзагроз, зокрема системи виявлення вторгнень та підходи до аналізу мережевого трафіку. Незважаючи на значний розвиток таких технологій у сфері IT, їх застосування в промислових системах лише набуває поширення та потребує подальших досліджень.

Одним із перспективних напрямів підвищення ефективності виявлення атак є інтеграція даних різної природи, зокрема мережевого трафіку та даних фізичних датчиків. Використання лише одного джерела інформації може бути недостатнім для точної ідентифікації інцидентів, оскільки відхилення у фізичних параметрах не завжди свідчать про кібератаку і можуть бути зумовлені технічними несправностями або впливом зовнішніх факторів. Поєднання багатодоменних даних дозволяє підвищити точність виявлення, зменшити кількість хибних спрацьовувань та забезпечити більш глибоке розуміння механізмів атак.

Таким чином, існує об’єктивна потреба у розробленні нових підходів до забезпечення кібербезпеки промислових систем керування, що враховують специфіку їх функціонування, архітектурні особливості та сучасні кіберзагрози. Навчальний посібник спрямований на формування системного розуміння цих питань, а також на ознайомлення з сучасними методами аналізу та виявлення кіберінцидентів у середовищі АСУТП та промислового Інтернету речей.

Структура навчального посібника складається з п’яти розділів.

У першому розділі розглянуто основи побудови автоматизованих систем управління технологічними процесами з позиції кібербезпеки. Проаналізовано роль програмованих логічних контролерів, основні функції АСУТП, а також складові системи, зокрема HMI, SCADA та DCS. Окрему увагу приділено моделі Пердью, яка визначає ієрархічну структуру промислових мереж і є базою для реалізації принципів сегментації та захисту.

У другому розділі наведено огляд можливих сценаріїв втручання у функціонування АСУТП. Розглянуто приклади атак на технологічний сегмент через суміжні інформаційні системи, а також описано сучасні засоби виявлення кіберзагроз, зокрема системи класу EDR, мережеві пісочниці та SIEM-рішення.

Третій розділ присвячено протоколам промислового Інтернету речей. У ньому розглянуто рівні архітектури IIoT, основні протоколи передачі даних, а також детально проаналізовано протокол MQTT, його характеристики та формат повідомлень. Окремо описано процес розгортання брокера MQTT Eclipse Mosquitto та використання платформи Cedalo для його адміністрування. Також розглянуто питання безпеки IoT та класифікацію загроз у промисловому середовищі.

У четвертому розділі представлено методику аналізу мережевого трафіку промислових систем з метою виявлення несанкціонованого втручання. Основну увагу приділено протоколу Modbus, його різновидам, регістрам та функціям. Розглянуто інструменти аналізу мережевого трафіку, зокрема Wireshark, а також наведено практичні приклади дослідження мережевих пакетів.

П’ятий розділ присвячено моніторингу радіочастотного середовища в АСУТП. Розглянуто особливості бездротових мереж з точки зору кібербезпеки, класифікацію радіочастотних атак, методи їх виявлення та протидії. Описано сучасні технології радіомоніторингу, включаючи програмно-визначені радіосистеми (SDR), а також подано практичні приклади виявлення несанкціонованих передавачів у мережах LoRaWAN.

Таким чином, навчальний посібник охоплює як теоретичні основи, так і практичні аспекти забезпечення кібербезпеки промислових систем, що дозволяє сформувати комплексні компетентності у сфері захисту автоматизованих систем управління технологічними процесами та промислового Інтернету речей.

1 АВТОМАТИЗОВАНА СИСТЕМА УПРАВЛІННЯ ТЕХНОЛОГІЧНИМИ ПРОЦЕСАМИ З ТОЧКИ ЗОРУ КІБЕРБЕЗПЕКИ

1.1 Використання ПЛК для створення автоматизованої системи управління технологічними процесами

Автоматизовану систему управління технологічними процесами (АСУТП) можна подати у вигляді окремих її частин, які називаються підсистемами. Підсистема – це частина основної системи, виділена за будь-якою ознакою. Сукупність підсистем автоматизованої системи управління технологічними процесами, незалежно від сфери застосування, як правило, одна й та сама.

Структура будь-якої АСУТП може бути представлена сукупністю підсистем, що її забезпечують, серед яких зазвичай виділяють програмно-технічне забезпечення, інформаційне, математичне (іноді алгоритмічне), організаційно-методичне та правове. Основною змінною величиною, що характеризує АСУТП, є критерій управління.

Критерієм управління називається співвідношення, яке залежить від якості функціонування системи в цілому, і приймає конкретні числові значення в залежності від керуючих впливів, що використовуються.

Як було сказано вище, автоматизована система управління технологічними процесами, складається з підсистем. Розглянемо докладніше значення кожної з них:

– технічне забезпечення, яке включає обчислювальні і керуючі пристрої, засоби отримання інформації (датчики), засоби перетворення, зберігання, відображення та реєстрації інформації, передачі і перетворення сигналів, і виконавчі пристрої. Прикладами таких пристроїв можуть бути: комп’ютери, програмовані логічні контролери, маршрутизатори, електронні реєстратори, промислові панелі оператора;

– програмне забезпечення, що складається із сукупності програм, необхідних для реалізації функцій АСУТП і забезпечення керуючого алгоритму, без якого неможливе функціонування комплексу технічних засобів;

– інформаційне забезпечення містить відомості, що характеризують стан системи управління, системи класифікації та кодування технологічної та техніко-економічної інформації, масиви даних і документів, необхідних для

виконання функцій АСУТП, в тому числі нормативно-довідкову і правову інформацію;

– організаційне забезпечення – це сукупність описів функціональних, технічних і організаційних структур, а також інструкцій для оперативного персоналу; дана сукупність повинна забезпечити належне функціонування перерахованих структур;

– оперативний персонал – це технологи-оператори, які здійснюють контроль за управлінням системи;

– експлуатаційний персонал – це персонал, який забезпечує експлуатацію, технічне обслуговування та ремонт системи.

Типова структура автоматизованої системи управління промисловим підприємством подана на рис. 1.1.

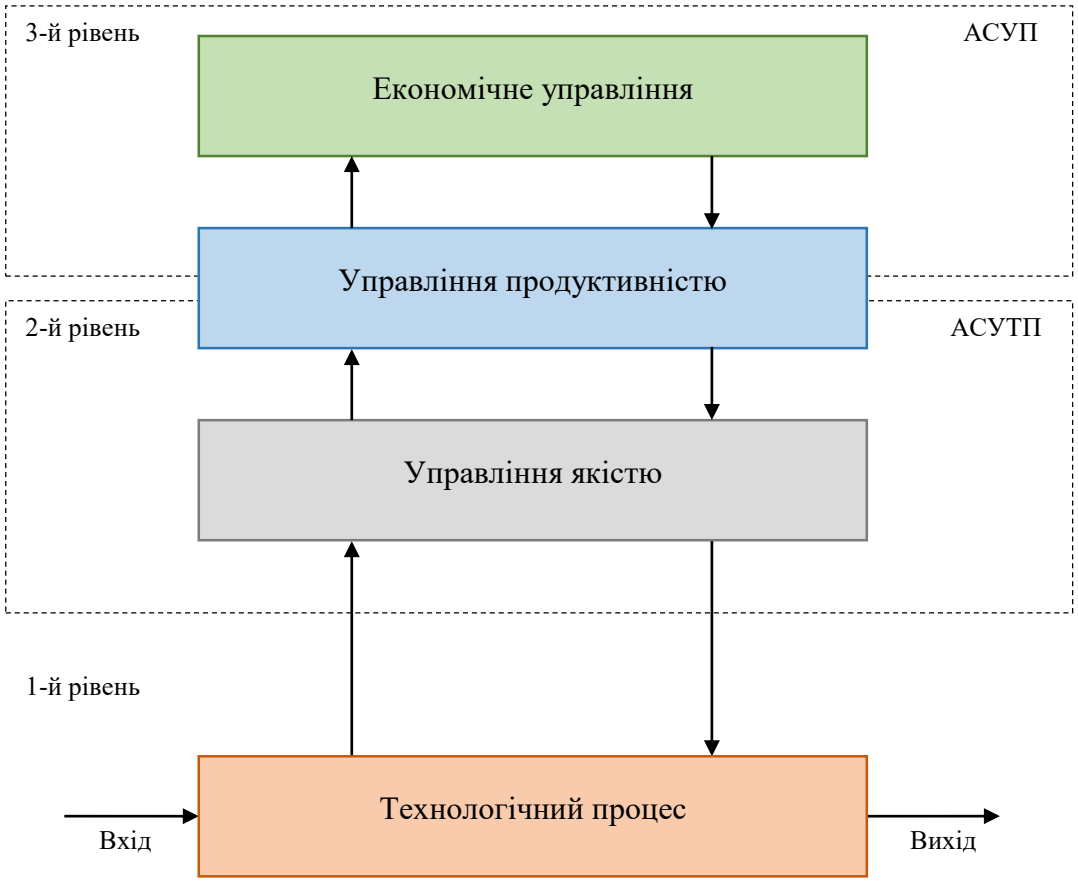


Рисунок 1.1 – Структура АСУ промислового підприємства

На даному рисунку показано місце підсистем і компонентів в складі системи управління підприємством. Виділено рівні технологічного процесу,

АСУТП і АСУП. Виділено компоненти економічного управління, управління якістю і продуктивністю.

В процесі побудови засобів сучасної промислової автоматики у вигляді автоматизованої системи управління технологічними процесами, використовується ієрархічна інформаційна структура із застосуванням на різних рівнях обчислювальних засобів різної потужності.

На рис. 1.2 подана загальна структура автоматики АСУТП.

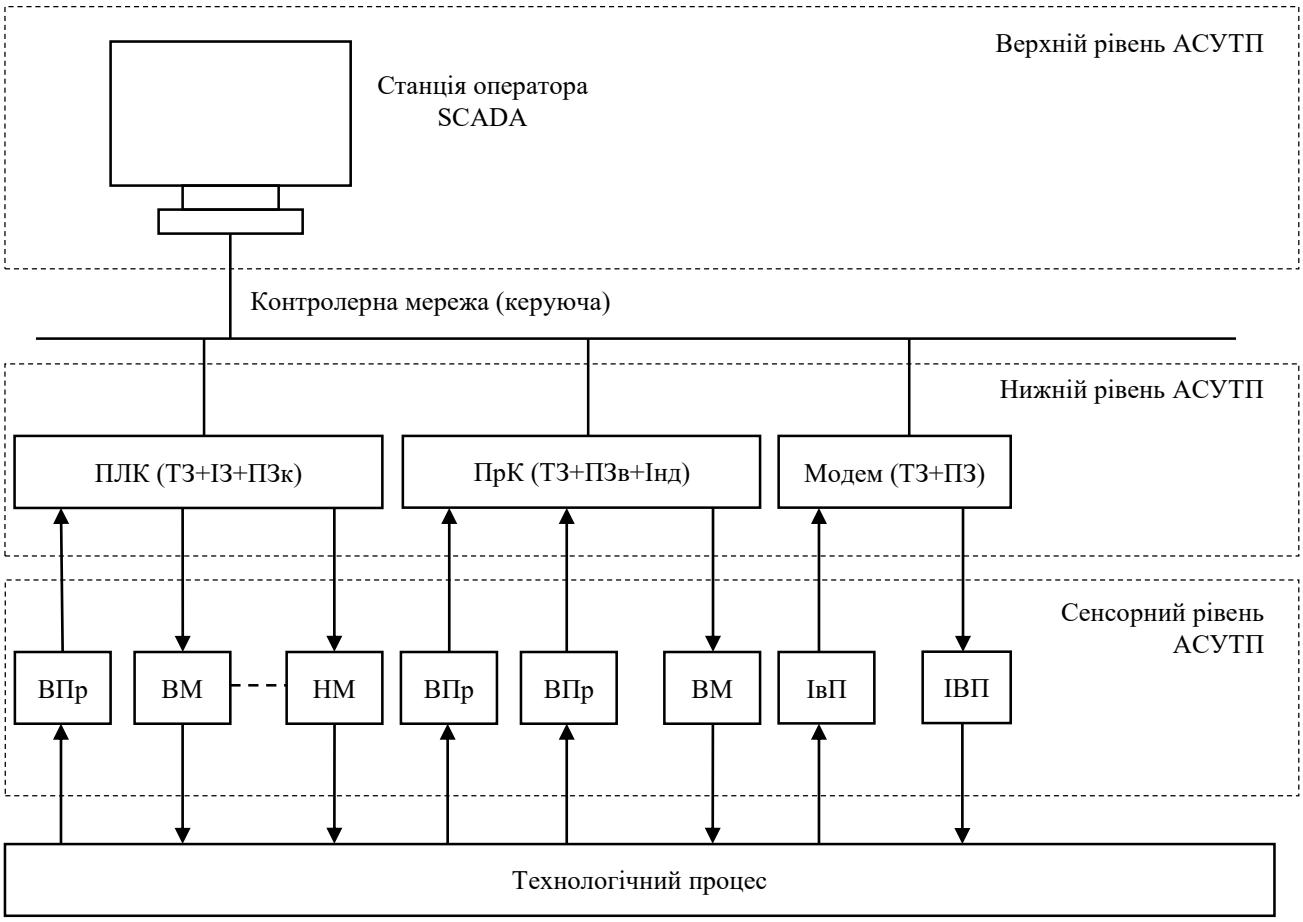


Рисунок 1.2 – Загальна схема автоматики АСУТП

На рис. 1.2 використані такі позначення:

– ВПр – вимірювальні перетворювачі (датчики);

– ВМ – виконуючи механізми;

– ПЛК – програмований логічний контролер;

– ПрК – програмований контролер;

– ІвП – інтелектуальний вимірювальний перетворювач;

– ІВП – інтелектуальні виконуючи пристрої;

– Модем – модулятор/демодулятор сигналів;

– ТЗ – технічне забезпечення (апаратна частина);

– ІЗ – інформаційне забезпечення (бази даних);

– ПЗ – програмне забезпечення;

– КЗ – комунікаційне забезпечення (послідовний порт и ПЗ);

– ПЗк – програмне забезпечення користувача;

– ПЗв – програмне забезпечення виробника;

– Інд – індикатор.

Частіше автоматизовані системи управління технологічними процесами створюють за такими типовими схемами:

– однорівнева схема (децентралізована, або іншими словами, локальна система), що містить програмований логічний контролер (ПЛК), або моноблочний контролер, що налаштовується (МНК), або програмований логічний контролер і панель оператора. Крім безпосередньо самого завдання управління, всі ці технічні засоби забезпечують індикацію і сигналізацію стану контрольованого або регульованого технологічного параметра за допомогою систем людино-машинного інтерфейсу. Це можуть бути як прості текстові панелі, так і сенсорні панелі оператора з великою функціональністю;

– дворівнева схема (централізована система), що включає в себе: на нижньому рівні – декілька ПЛК з підключеними до них датчиками і виконавчими пристроями, на верхньому рівні – одна, можливо декілька, операторських (робочих) станцій. Робоча станція представляє собою комп’ютер для створення автоматизованого робочого місця (АРМ) оператора. Зазвичай робоча станція або АРМ укомплектована системою збору та візуалізації даних (SCADA-системою) того чи іншого рівня складності.

Дворівневі системи промислової автоматики дають величезні переваги в глибині зберігання інформації, створення архівів, доступу до інформації з будь-якої точки світу через Web-інтерфейс. Також перевагою дворівневої системи є розподіл завдань між вузлами системи, що дозволяє підвищити надійність її функціонування. Типова функціональна схема дворівневої АСУТП подана на рис. 1.3.

Усі ПЛК та автоматизовані робочі станції об’єднуються промисловими інформаційними мережами, що забезпечують постійний обмін даними.

Основні функції нижчого рівня:

– збір, електрична фільтрація і перетворення сигналів з первинних перетворювачів (датчиків) з аналогової в цифрову форму;

– реалізація локальних автоматизованих систем управління конкретного технологічного процесу в об’ємі функцій ПЛК однорівневої системи;

– реалізація аварійної і попереджувальної сигналізації;

– організація системи захисту і блокувань;

– обмін поточними даними з ПК верхнього рівня через промислову мережу по запитах ПК.

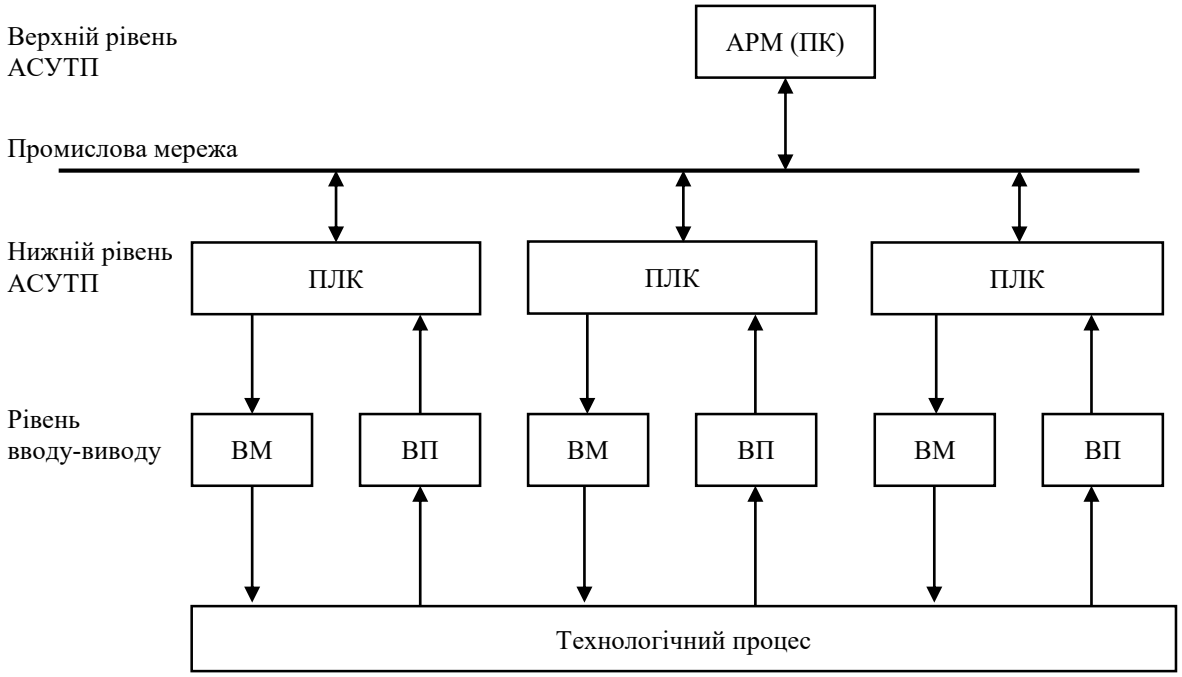


Рисунок 1.3 – Типова функціональна схема дворівневої АСУТП

Основні функції верхнього рівня:

– візуалізація стану технологічного процесу;

– поточна реєстрація характеристик технологічного процесу;

– оперативний аналіз стану устаткування і технологічного процесу;

– реєстрація дій оператора, у тому числі під час аварійних повідомлень;

– архівація і тривале зберігання значень протоколів технологічного процесу;

– реалізація алгоритмів «системи порадника»;

– супервізорне управління;

– реалізація безпосереднього цифрового управління;

– управління в режимі збору даних – зберігання і ведення баз цих параметрів техпроцесів, критичних параметрів устаткування, ознак аварійних станів технологічного процесу;

– реалізації системи контролю доступу операторів і їх паролів.

Таким чином, нижній рівень реалізує алгоритми управління устаткуванням, верхній – вирішення стратегічних питань функціонування АСУТП.

Наприклад, рішення увімкнути, вимкнути насос, змінити його частоту обертання починається на верхньому рівні, а подання усіх необхідних сигналів керування, перевірка теплового перевантаження насосу, реалізація механізму блокувань і захисту по «сухому» ходу виконується на нижньому рівні.

Ієрархічна структура автоматизованої системи управління технологічними процесами передбачає:

– потік команд спрямований від верхнього рівня до нижнього;

– нижній рівень надає відповіді верхньому за його запитами.

Ця структура підвищує надійність усієї системи управління в цілому, оскільки при виході з ладу комплексу технічних засобів верхнього рівня працездатність нижнього рівня АСУТП зберігається. Такі несправності сприймаються нижнім рівнем лише як відсутність нових команд і запитів, але безпосередньо на безпечний хід технологічного процесу не впливають.

В процесі конфігурації ПЛК встановлюється: до якого часу після отримання останнього запиту ПЛК продовжує функціонувати, підтримуючи останній заданий режим, після чого переходить в потрібний в цій аварійній ситуації режим роботи.

1.2 Архітектура АСУТП

На рис. 1.4 подано архітектуру сучасної автоматизованої системи управління технологічними процесами.

Автоматизована система управління технологічними процесами розташована в області, яка позначена промисловою зоною (Industrial Zone). Однак, оскільки більшість АСУТП взаємодіють із корпоративною зоною (Enterprise Zone), щоб ефективно захистити систему в цілому, слід також звернути увагу на системи і в цій, корпоративній, зоні.

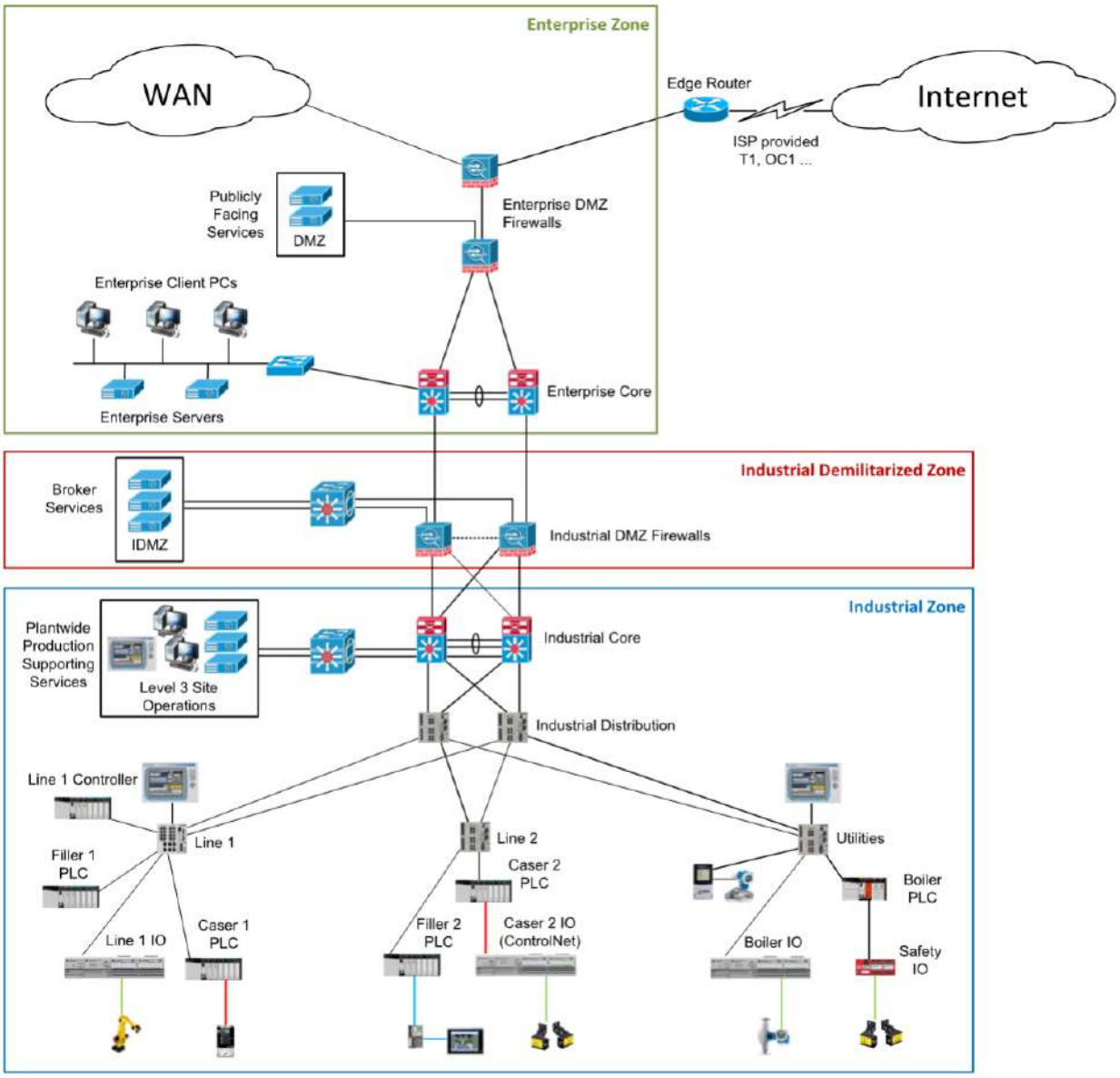


Рисунок 1.4 – Архітектура сучасної автоматизованої системи управління технологічними процесами

АСУТП – це різноманітні системи керування та пов’язані з ними інструменти, що використовуються в технології промислового виробництва для досягнення спільної мети, наприклад створення продукту чи надання послуги. З точки зору високого рівня, АСУТП можна класифікувати за їх функцією. Вони можуть мати одну або кілька функцій. Розглянемо ці функції докладно.

1.2.1 Функція спостереження

Функція спостереження включає в себе можливість спостерігати за поточним станом системи автоматизації в реальному часі (рис. 1.5). Ці дані можуть використовуватися операторами, керівниками, інженерами з технічного

обслуговування та іншим персоналом для прийняття бізнес-рішень або виконання коригувальних дій. Наприклад, коли оператор бачить, що температура печі падає, він може вирішити збільшити подачу пари в піч, щоб компенсувати це. Процес спостереження є пасивним за своєю природою, він просто надає інформацію, щоб людина реагувала на визначені зміни в ході технологічного процесу.

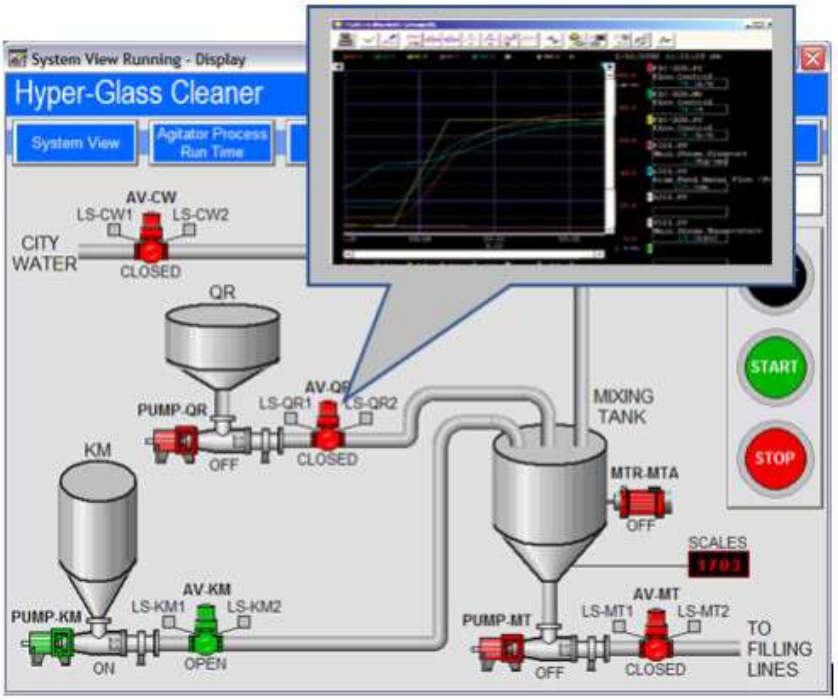


Рисунок 1.5 – Функція спостереження за станом ТП

З точки зору безпеки, якщо зловмисник може маніпулювати представленням оператора про стан системи управління або, іншими словами, може змінити значення, на яких оператор базує свої рішення, зловмисник ефективно контролює реакцію, а значить і увесь процес. Наприклад, маніпулюючи показаним значенням температури печі, зловмисник може змусити оператора подумати, що температура надто низька або зависока, і змусити його діяти на основі маніпуляційних даних.

1.2.2 Функція моніторингу

Функція моніторингу часто є частиною циклу контролю, наприклад, автоматизованої системи, яка підтримує стабільний рівень у баку з рідиною. Функція моніторингу стежитиме за критичними значеннями, такими як тиск, температура, рівень тощо, і порівнюватиме поточне значення з попередньо

визначеними пороговими значеннями, а також подаватиме тривогу або взаємодіятиме залежно від налаштування функції моніторингу (рис. 1.6).

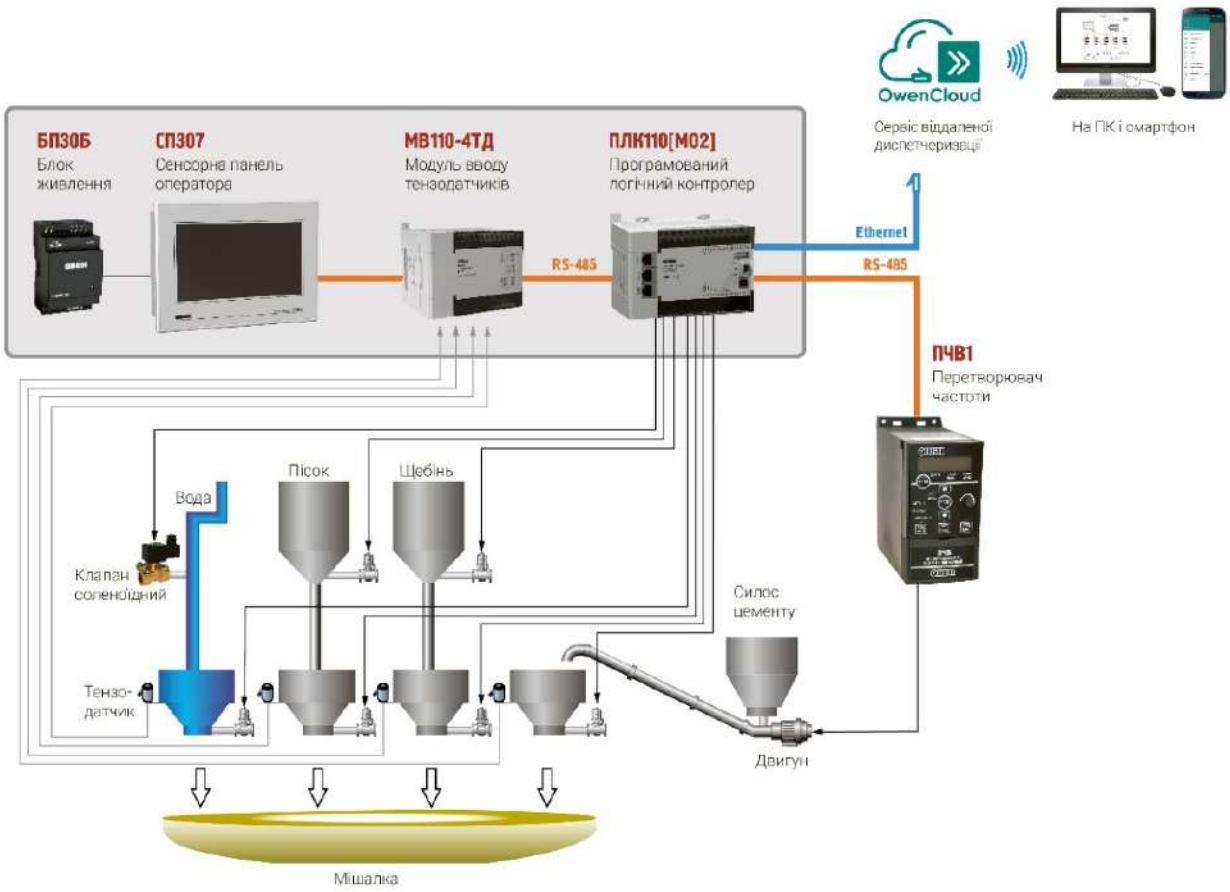


Рисунок 1.6 – Функція моніторингу стану ТП

Ключова відмінність між функціями спостереження і моніторингу полягає у визначенні відхилення. За допомогою функцій моніторингу визначення є автоматизованим процесом, тоді як за допомогою функції спостереження це визначення виконується людиною, яка переглядає значення. Реакція функції моніторингу може варіюватися від спливаючого екрана тривоги до автоматизованого вимкнення системи.

З точки зору безпеки, якщо зловмисник може контролювати значення, яке контролює функція моніторингу, реакцію функції можна запустити або, навпаки, запобігти її спрацюванню. Наприклад, система моніторингу перевіряє температуру печі, запобігаючи перевищенню температури 300 °C. Якщо зловмисник введе в систему значення менше 300 °C, цю систему обманом змусять повірити, що все гаразд, тоді система може зазнати збою.

1.2.3 Контрольна функція

Контрольна функція в автоматизованій системі управління технологічним процесом (АСУТП) є однією з основних функцій, яка забезпечує безперервний моніторинг та керування виробничими процесами з метою підвищення їх ефективності, надійності та безпеки (рис. 1.7).

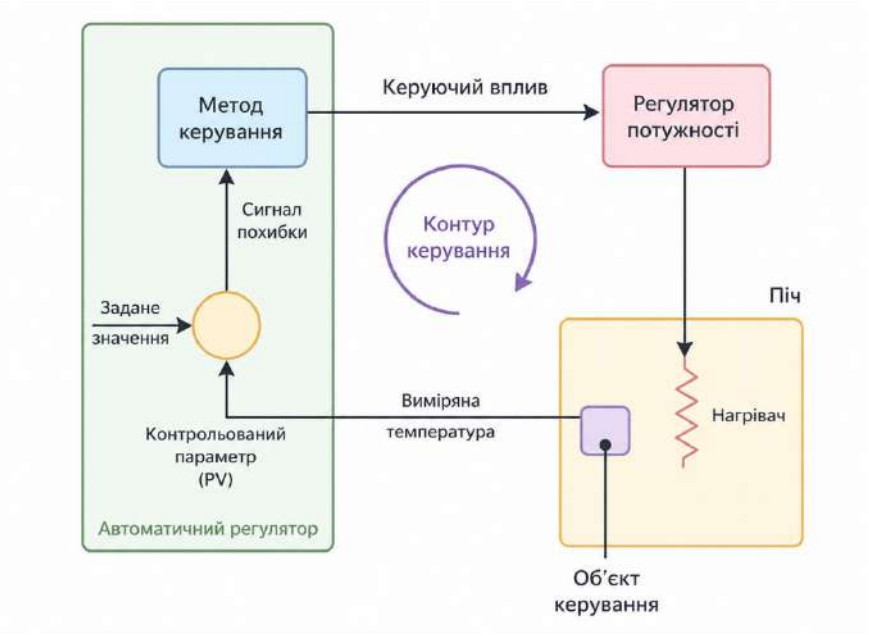


Рисунок 1.7 – Контрольна функція АСУТП

Система контролю збирає дані з різних датчиків, лічильників та інших технічних засобів, які вимірюють параметри процесу (температуру, тиск, витрати, рівень, швидкість тощо). У разі виявлення відхилень або аномалій система контролю може автоматично змінювати параметри роботи обладнання, щоб привести процес у відповідність до заданих норм.

Система керування – це те, що змушує приводи рухатись, відкривати клапани та працювати двигуни. Дії керування ТП можуть бути ініційовані оператором, натисканням кнопки або зміною певного параметру на екрані панелі оператора. Також це може бути автоматизована відповідь ПЛК, як частина керування технологічним процесом.

З точки зору безпеки, якщо зловмисник може маніпулювати значеннями (вхідними даними), на які реагує система керування, або якщо зловмисник може змінити чи маніпулювати самою контрольною функцією (програмою керування), систему можна обманом змусити робити те, що вона не мала робити, або не була призначена для виконання цих дій.

Промислова система управління ТП (рис. 1.8) – це термін, який використовується для різних систем автоматизації та їх пристроїв, таких як програмовані логічні контролери, людино-машинний інтерфейс (HMI), системи диспетчерського керування та збору даних (SCADA), розподілені системи керування (DCS), системи безпеки (SIS) та багато інших.

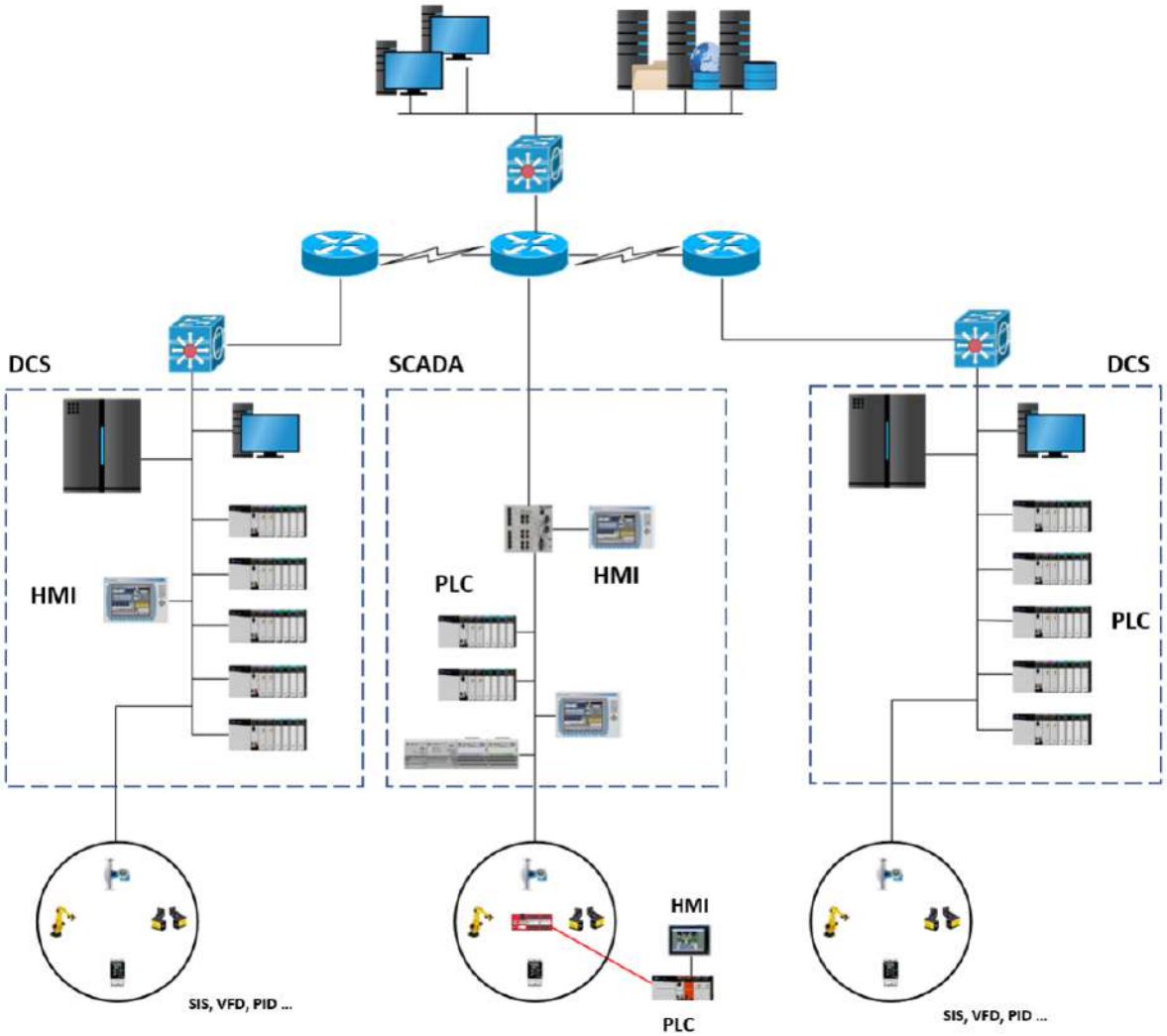


Рисунок 1.8 – Промислова система управління ТП

1.2.4 Програмовані логічні контролери

Програмовані логічні контролери, або ПЛК, є основою майже кожної промислової системи керування (рис. 1.9). Це пристрої, які приймають дані від датчиків через вхідні канали та керують виконавчими механізмами через вихідні канали. Типовий ПЛК складається з мікроконтролера і масиву вхідних і вихідних каналів. Вхідні та вихідні канали можуть бути аналоговими, цифровими або

доступними для мережі. Ці канали вводу/виводу часто постачаються як додаткові карти, які приєднуються до задньої панелі ПЛК. Таким чином, ПЛК можна налаштувати відповідно до багатьох різних функцій і реалізацій.

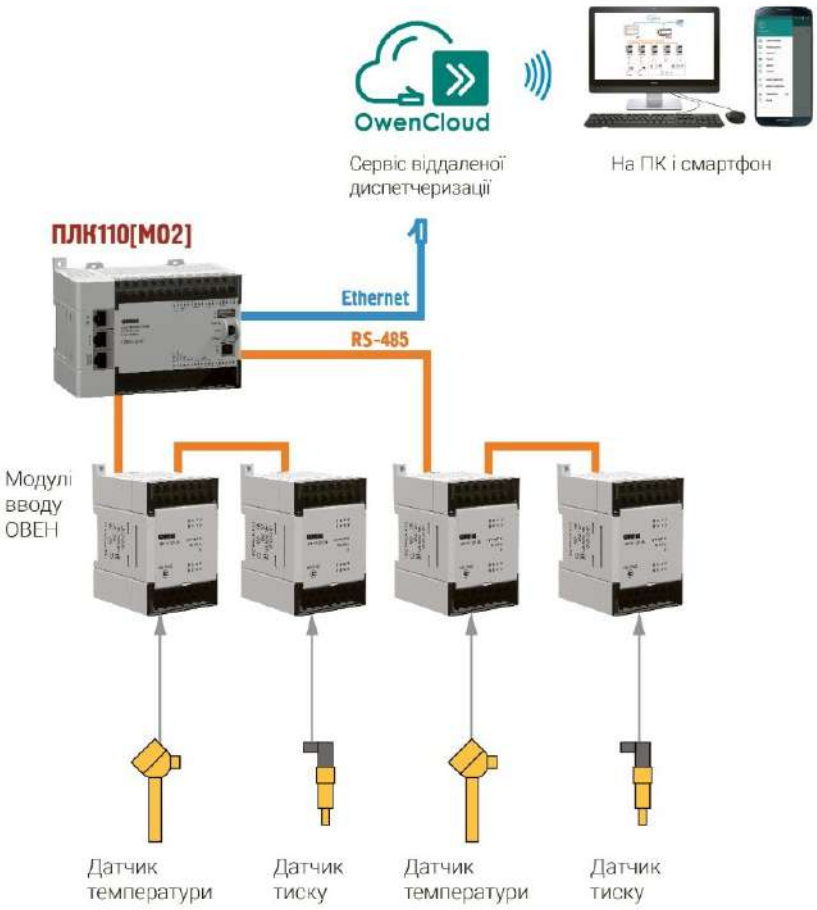


Рисунок 1.9 – Використання ПЛК в системі автоматизації ТП

Програмування шини ПЛК можна виконати через виділений USB або послідовний інтерфейс на пристрої або через мережеву комунікаційну шину, яка вбудована в пристрій або поставляється як додаткова карта. Поширеними типами мереж є Modbus, Ethernet, ControlNet, PROFINET тощо.

ПЛК можуть бути розгорнуті як автономні пристрої, що контролюють певну частину виробничого процесу, наприклад окрему машину, або вони можуть бути розгорнуті як розподілені системи, що охоплюють кілька заводів у розрізнених місцях з тисячами точок вводу/виводу та численними взаємопов’язаними частинами.

1.2.5 Панелі оператора

Людино-машинний інтерфейс (HMI) – це вікно в систему керування, що візуалізує запущений процес, дозволяючи перевіряти та маніпулювати значеннями процесу, відображати сигнали тривоги та тренди контрольних значень. У своїй найпростішій формі HMI – це автономний сенсорний пристрій, який спілкується через послідовний або інкапсульований протокол Ethernet. Більш просунуті системи HMI можуть використовувати розподілені сервери, щоб запропонувати резервну поставку екранів і даних HMI. На рис. 1.10 подано приклад застосування панелі оператора.

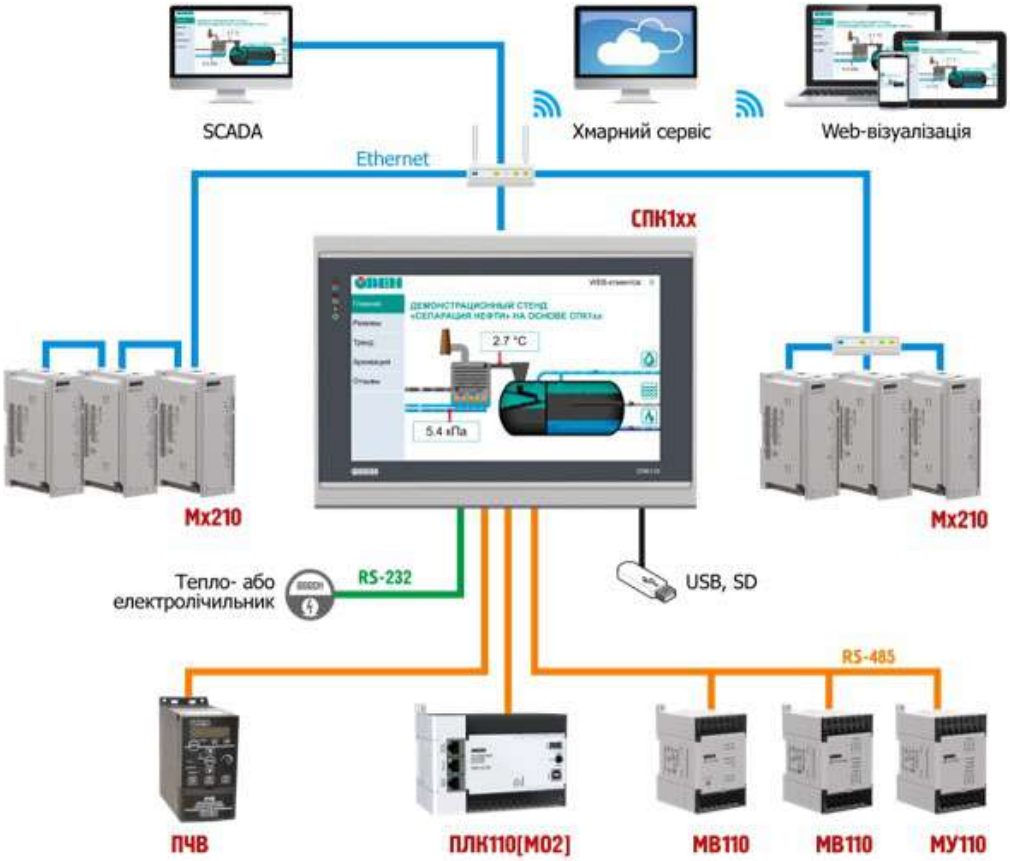


Рисунок 1.10 – Приклад застосування панелі оператора

1.2.6 Система диспетчерського контролю та збору даних

Система диспетчерського контролю та збору даних – це термін, який використовується для опису спільного використання технічних засобів АСУТП, які працюють разом над спільним завданням. На рис. 1.11 подано приклад мережі SCADA.

Тут мережа SCADA складається з обладнання та компонентів автоматизованої системи, які разом утворюють загальну систему. Системи SCADA часто розповсюджуються на широку географічну територію в результаті застосування до автоматизованого управління електромережами, водопостачанням, трубопроводами та іншими системами керування, які використовують віддалені робочі станції.

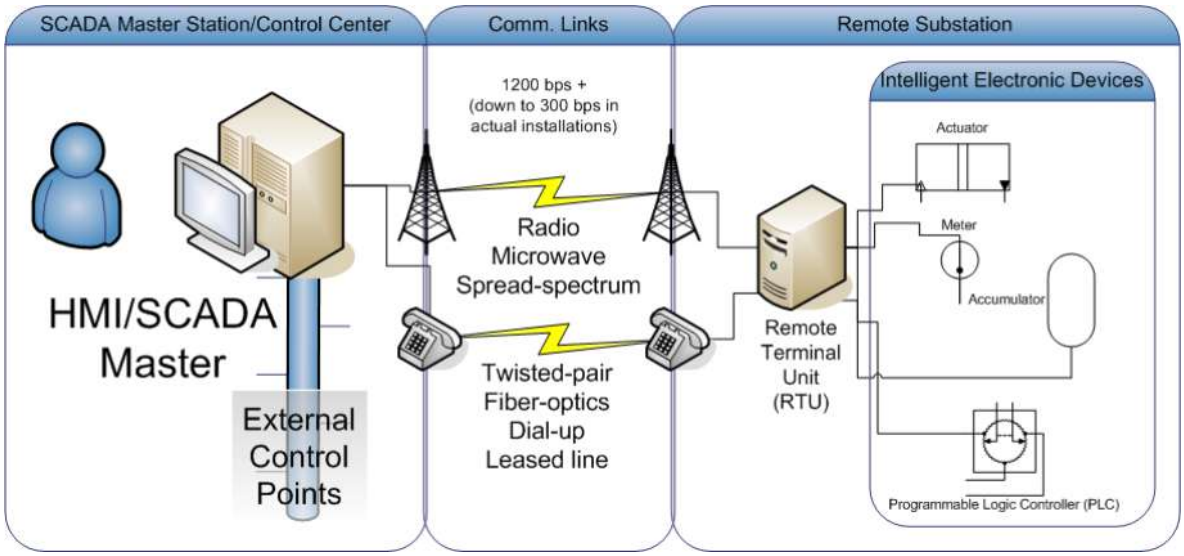


Рисунок 1.11 – Приклад мережі SCADA

1.2.7 Розподілена система управління

З системою SCADA тісно пов’язана розподілена система управління (DCS). Відмінності між системою SCADA та DCS дуже незначні, і з часом вони майже не відрізняються. Традиційно системи SCADA використовувалися для завдань автоматизації, які охоплювали більшу географічну територію, тобто частини системи SCADA розташовувалися в окремих будівлях або об’єктах, оскільки DCS частіше обмежувалася одним об’єктом. DCS часто є великомасштабною високотехнічною системою з дуже конкретним завданням.

DCS використовує централізований блок нагляду, який може контролювати тисячі точок вводу/виводу. Промислова мережа створена для довговічності з резервуванням, застосованим до всіх рівнів інсталяції, від резервних мереж і мережевого інтерфейсу, підключених до резервних наборів серверів, до резервних контролерів і датчиків, і все це з урахуванням створення жорсткої та надійної платформи автоматизації. На рис. 1.12 подана типова структура розподіленої системи керування ТП.

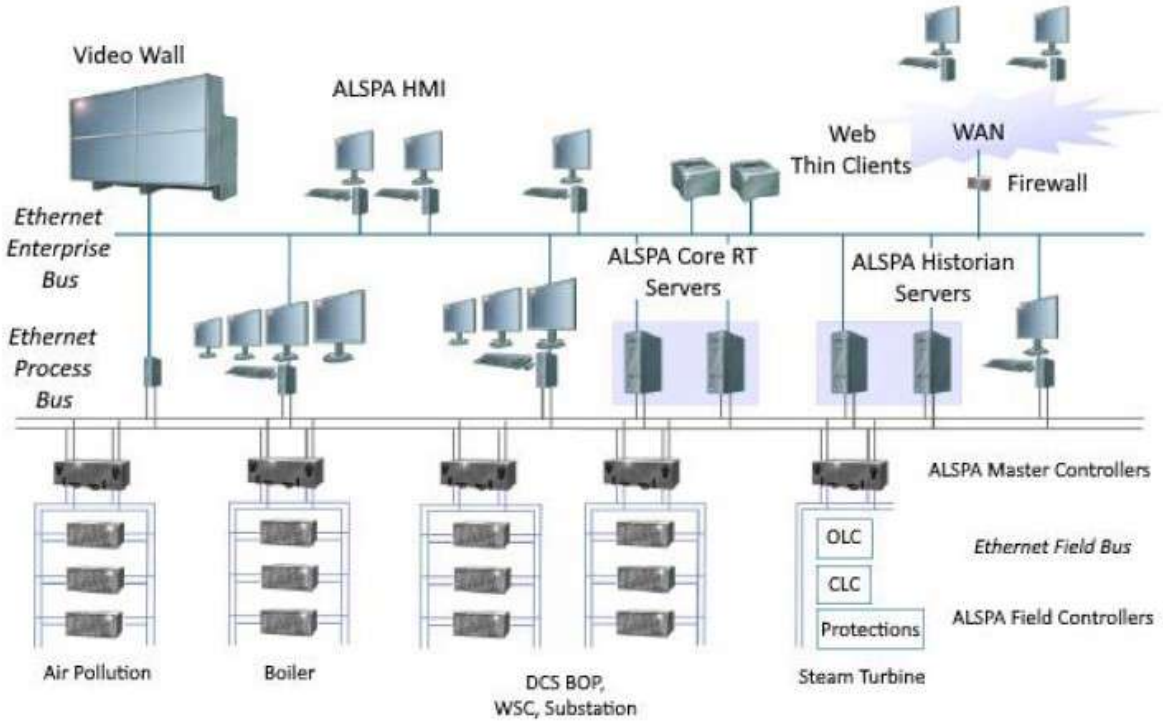


Рисунок 1.12 – Типова структура розподіленої системи керування ТП

1.2.8 Система безпеки

Системи безпеки, або SIS, є спеціальними системами моніторингу безпеки. Вони призначені для безпечного й обережного вимкнення системи, що контролюється, або переведення цієї системи в попередньо визначений безпечний стан у разі несправності апаратного забезпечення. SIS використовує набір систем голосування, щоб визначити, чи працює система правильно (рис. 1.13).

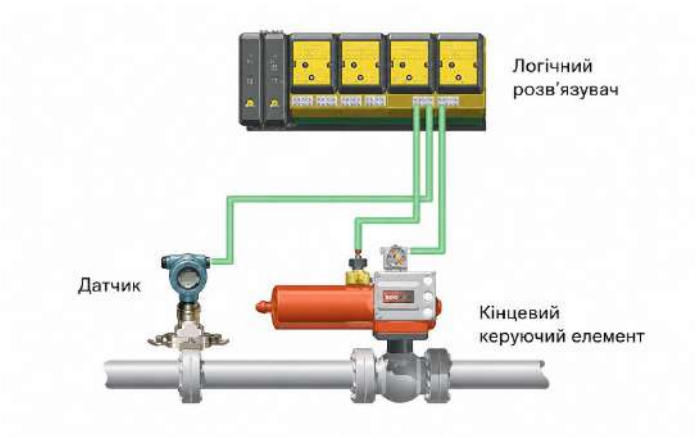


Рисунок 1.13 – Система безпеки в АСУТП

1.3 Модель Пердью для промислових систем управління

Як показано на рис. 1.14, модель Пердью була взята з моделі Purdue Enterprise Reference Architecture (PERA) ISA-99 і використана як концептуальна модель для сегментації мережі АСУТП.

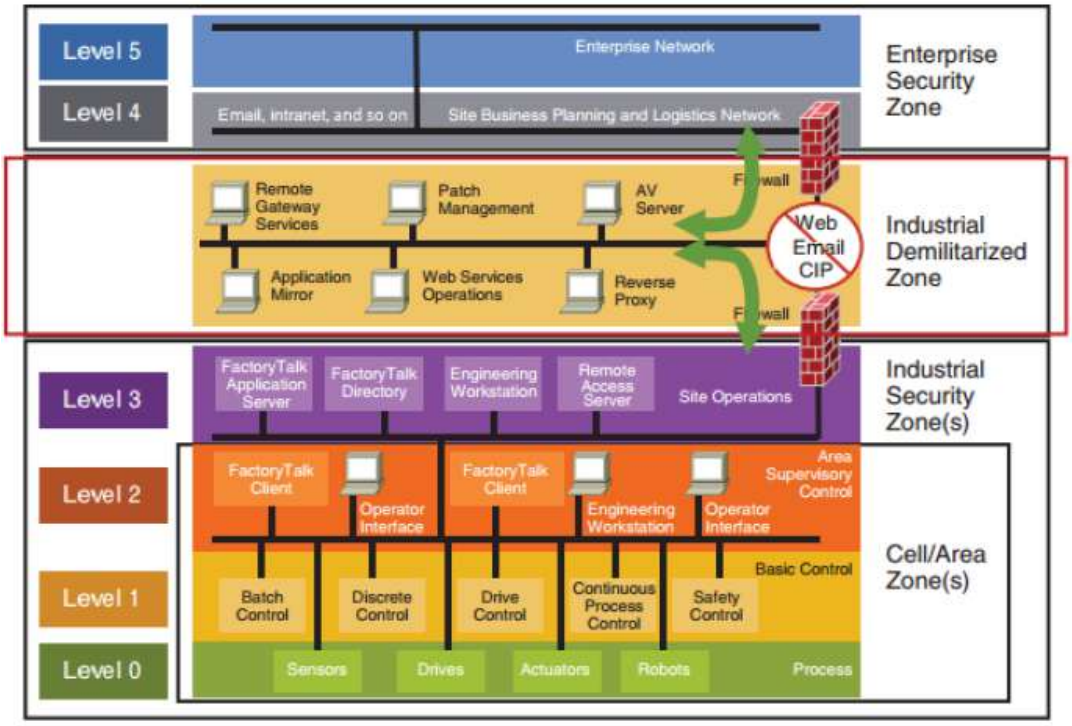


Рисунок 1.14 – Модель Пердью для промислових систем управління

Це прийнята галуззю автоматизації еталонна модель, яка показує взаємозв'язки та взаємозалежності всіх основних компонентів типової АСУТП.

Модель Пердью розділяє архітектуру АСУТП на три зони та шість рівнів. Починаючи згори, це:

а) підприємство:

– рівень 5: корпоративна мережа;

– рівень 4: бізнес-планування та логістика;

б) промислова демілітаризована зона;

в) виробнича зона (також звана промисловою зоною):

– рівень 3: управління виробництвом;

– рівень 2: зона наглядового контролю;

– рівень 1: базовий контроль;

– рівень 0: виробничі процеси.

1.3.1 Корпоративна зона

На діаграмі Пердью ця зона розповсюджується на два рівні (рис. 1.15):

– рівень 5: корпоративна мережа;

– рівень 4: бізнес-планування та логістика.

На цих рівнях виконуються такі завдання, як планування та управління ланцюгом поставок.

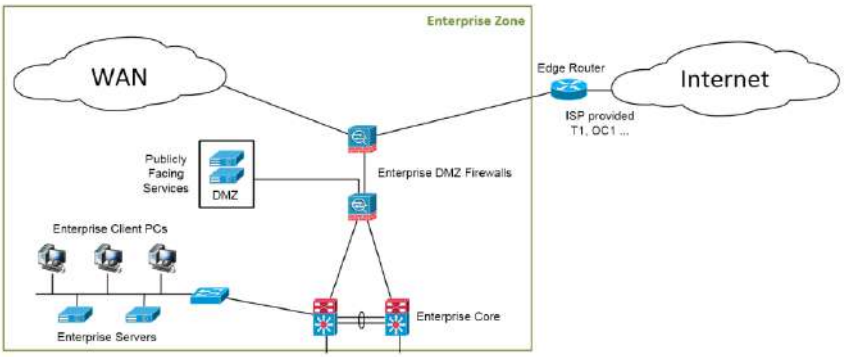


Рисунок 1.15 – Корпоративна зона АСУТП

Корпоративна зона – це частина АСУТП, де зазвичай знаходяться такі бізнес-системи, як ERP і SAP.

ERP (Enterprise Resource Planning) – це система планування ресурсів підприємства, яка інтегрує та автоматизує основні бізнес-процеси, такі як фінансове управління, виробництво, постачання, логістика, людські ресурси тощо. В контексті АСУТП, ERP виконує важливу функцію, з’єднуючи технологічні процеси з бізнес-процесами підприємства для оптимізації загальної продуктивності та ефективності.

ERP системи збирають інформацію з АСУТП, наприклад, про кількість виготовленої продукції, використання ресурсів, стан обладнання, і інтегрують ці дані з іншими бізнес-процесами, такими як управління фінансами, постачанням і продажами.

ERP дозволяє автоматизувати планування виробництва на основі даних, які надходять з АСУТП. Модуль управління виробництвом в ERP системі допомагає оптимізувати процеси, зменшити простої та ефективніше використовувати матеріали і ресурси. Інформація, зібрана з АСУТП (наприклад, про використання сировини або готової продукції), інтегрується в модулі ERP для управління запасами. Це дозволяє автоматично підтримувати оптимальний рівень матеріалів на складах, попереджуючи нестачу чи надлишок.

Дані про стан обладнання, зібрані через АСУТП, можуть передаватися в ERP для управління технічним обслуговуванням. Це допомагає планувати профілактичне обслуговування, ремонти та модернізацію обладнання.

Також, ERP системи надають можливість створювати звіти про виробничі процеси на основі даних з АСУТП. Це допомагає керівництву оцінювати ефективність виробництва, якість продукції та інші ключові показники. ERP системи можуть використовувати дані з АСУТП для контролю якості продукції на всіх етапах виробничого циклу. Це дозволяє відстежувати продуктивність і якість в реальному часі, а також швидко реагувати на відхилення від стандартів.

АСУТП автоматизує управління технологічними процесами на рівні обладнання, датчиків і виробничих ліній, збираючи інформацію в режимі реального часу і контролюючи виконання технологічних операцій. ERP працює на більш високому рівні, забезпечуючи управління ресурсами підприємства, фінансами, постачаннями і стратегічним плануванням. Інтеграція ERP та АСУТП дозволяє створити єдину систему управління підприємством, де дані з технологічних процесів використовуються для прийняття бізнес-рішень, що дозволяє оптимізувати всі аспекти діяльності.

SAP (Systems, Applications, and Products in Data Processing – системи, застосунки та продукти в обробці даних) – це програмне забезпечення, яке використовується для управління бізнес-процесами, включаючи виробничі, фінансові та логістичні аспекти підприємств. Хоча сама SAP не є традиційною системою автоматизації, вона може бути інтегрована з АСУТП для ефективного управління виробничими процесами та оптимізації ресурсів.

SAP може бути інтегрована з АСУТП для отримання даних з виробничих систем і забезпечення більш високого рівня управління підприємством, наприклад, управління виробничими ланцюгами, закупівлями, складським обліком тощо.

Система SAP дозволяє синхронізувати реальні виробничі процеси з бізнес-операціями, забезпечуючи точне планування і виконання виробничих завдань.

Через модуль SAP PM (Plant Maintenance), SAP може допомагати в управлінні технічним обслуговуванням виробничого обладнання. АСУТП передає дані про стан обладнання в систему SAP, яка, в свою чергу, автоматично планує технічне обслуговування і ремонти. За допомогою модуля SAP PP (Production Planning) можна автоматизувати планування виробничих процесів, оптимізуючи використання ресурсів, матеріалів та часу.

Модуль SAP MM (Materials Management) дозволяє керувати запасами матеріалів на виробництві. Інтеграція з АСУТП дозволяє автоматично відстежувати використання матеріалів у процесі виробництва та керувати закупівлями.

1.3.2 Рівень 5 – Корпоративна мережа

Системи корпоративної мережі зазвичай розташовані на корпоративному рівні та охоплюють кілька об'єктів або підприємств. Вони беруть дані з підлеглих систем на окремих підприємствах і використовують накопичені дані для звітування про загальний стан виробництва, запаси та попит. Будучи технічно не частиною АСУТП, корпоративна зона покладається на підключення до мереж АСУТП для передачі даних, які керують бізнес-рішеннями.

1.3.3 Рівень 4 – Бізнес-планування та логістика виробництва

Рівень 4 є осередком для всіх систем інформаційних технологій (IT), які підтримують виробничий процес на підприємстві. Ці системи повідомляють про виробничу статистику, таку як час безвідмовної роботи та одиниці, вироблені для корпоративних систем, і приймають замовлення та бізнес-дані з корпоративних систем для розподілу між системами операційної технології (OT) або системами АСУТП.

Системи, які зазвичай знаходяться на рівні 4, включають сервери баз даних, сервери додатків (веб, звіти, MES), файлові сервери, клієнти електронної пошти, робочі столи супервізора тощо.

MES (Manufacturing Execution System) – це система управління виробничими процесами на підприємстві, яка забезпечує виконання, моніторинг та оптимізацію операцій на виробництві в режимі реального часу. В контексті АСУТП, MES є важливим рівнем, що інтегрує дані з виробничого обладнання та автоматизованих систем для покращення контролю та управління виробництвом. MES є проміжною ланкою між АСУТП і ERP. Вона отримує дані з АСУТП для моніторингу виробництва в режимі реального часу та передає інформацію на рівень ERP для управління бізнес-процесами (закупівля, постачання, фінанси).

MES отримує дані безпосередньо з обладнання та систем АСУТП, що дозволяє контролювати всі етапи виробничого процесу в режимі реального часу. Вона забезпечує відстеження стану виробничих ліній, контролює

продуктивність, використання ресурсів і відповідність технологічним стандартам.

MES координує виконання виробничих планів, перетворюючи стратегічні завдання, задані на рівні ERP (Enterprise Resource Planning), в конкретні операції для виробництва. Система може автоматично розподіляти завдання між різними виробничими одиницями, забезпечуючи оптимізацію використання ресурсів та виконання виробничих планів.

Система управління виробничими процесами дозволяє відстежувати рух матеріалів та компонентів протягом усього виробничого процесу, від сировини до готової продукції. Це включає управління партіями продукції, відстеження дефектів та контроль якості на кожному етапі виробництва.

MES допомагає керувати використанням матеріалів, робочої сили та обладнання. Вона може інтегруватися з ERP-системою для оптимізації постачання ресурсів та планування виробництва. Система дозволяє автоматично коригувати роботу виробничих ліній на основі наявності ресурсів і умов на виробництві.

Також, MES генерує звіти про виробничі операції, що допомагає керівництву оцінити ефективність процесів, продуктивність обладнання та якість продукції. Інформація може бути використана для довгострокового планування, оптимізації процесів і прийняття управлінських рішень.

1.3.4 Промислова демілітаризована зона

На рис. 1.16 подана структура демілітаризованої зони підприємства.

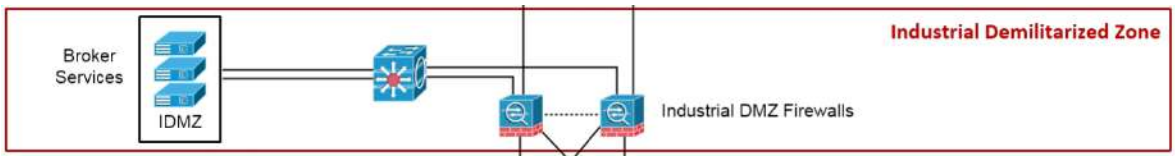


Рисунок 1.16 – Структура демілітаризованої зони підприємства

Між корпоративною зоною підприємства та промисловою зоною розташована промислова демілітаризована зона або IDMZ. Подібно до традиційної (IT) DMZ, орієнтована на OT IDMZ дозволяє безпечно з’єднувати мережі з різними вимогами безпеки.

IDMZ є результатом зусиль, спрямованих на створення стандартів безпеки, таких як NIST Cybersecurity Framework та NERC CIP. IDMZ є шаром обміну

інформацією між бізнесовими або IT-системами на рівнях 4 і 5 та виробничими або OT-системами на рівнях 3 і нижче. Запобігаючи прямому зв'язку між IT та OT системами та використовуючи брокерську службу в IDMZ для передачі комунікацій, додається додатковий шар розділення та інспекції до загальної архітектури. Системи на нижчих рівнях не піддаються прямому впливу атак або компрометації. Якщо якась система в IDMZ буде скомпрометована, IDMZ може бути вимкнена, компрометація може бути локалізована, і виробництво може продовжуватися.

Системи, які зазвичай знаходяться в Промисловій демілітаризованій зоні, включають (веб) проксі-сервери, сервери реплікації баз даних, контролери доменів Microsoft тощо.

1.3.5 Виробнича зона

Виробнича зона – це місце виконання промислових операцій, це ядро виробничої системи де виконується технологічний процес. Виробнича зона підрозділяється на чотири рівні [1]:

– рівень 3: управління виробництвом;

– рівень 2: зона наглядового контролю;

– рівень 1: базовий контроль;

– рівень 0: виробничі процеси.

На рис. 1.17 подані різні виробничі зони.

1.3.6 Рівень 3 – управління виробництвом

Рівень 3 – це місце, де знаходяться системи, які підтримують функції управління та моніторингу в масштабах підприємства. На цьому рівні оператор взаємодіє з усіма виробничими системами. Централізовані диспетчерські пульти з HMI та операторськими терміналами забезпечують огляд усіх систем, які керують процесами на заводі чи об'єкті. Оператор використовує ці системи HMI для виконання таких завдань, як перевірка якості, керування часом безвідмовної роботи та моніторинг сигналів тривоги, подій і тенденцій.

Рівень 3, управління виробництвом, також є місцем, де працюють системи OT, які звітують до IT-систем рівня 4. Системи на нижчих рівнях надсилають виробничі дані на сервери збору та агрегації даних на цьому рівні, які потім можуть надсилати дані на вищі рівні або запитувати їх від системи на вищих рівнях (операції push проти pull).

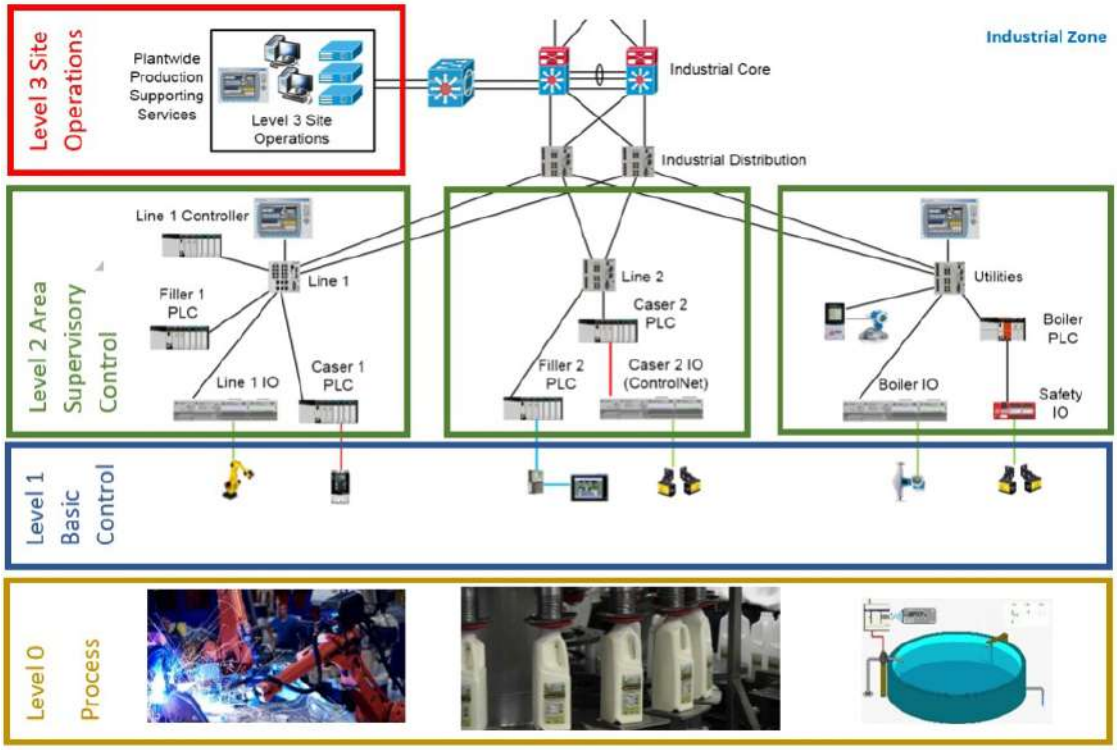


Рисунок 1.17 – Виробнича зона АСУТП

Системи, які зазвичай знаходяться на рівні 3, включають сервери баз даних, сервери додатків (веб-сервери), файлові сервери, контролери домену Microsoft, інженерні робочі станції серверів HMI тощо.

1.3.7 Рівень 2 – зона наглядового контролю

Багато функцій і систем на рівні 2 такі самі, як і на рівні 3, але спрямовані на меншу частину задач управління ТП або область загальної автоматизованої системи. На цьому рівні певні частини системи контролюються та керуються системами HMI. Мова йде про певну машину чи верстат з ЧПУ, де за допомогою сенсорного екрану HMI, запускаються або зупиняються певні функції виробничого обладнання, переглядаються деякі робочі параметри та виконується маніпулювання пороговими значеннями та заданими параметрами для верстатів з ЧПУ, або іншого промислового обладнання.

Системи, які зазвичай знаходяться на рівні 2, включають: HMI (автономні або системні клієнти), системи диспетчерського керування, такі як ПЛК керування лінією, інженерні робочі станції тощо.

1.3.8 Рівень 1 – базовий контроль

Рівень 1 – це місце, де знаходиться все контрольне обладнання. Головне призначення пристроїв на цьому рівні – управління виконавчими пристроями (відкривати клапани, рухати приводи, запускати двигуни тощо). Як правило, до рівня 1 входять ПЛК, частотно-регулюючі приводи (VFD), спеціальні пропорційно-інтегрально-диференціальні (PID) контролери тощо. Хоча можна знайти ПЛК на рівні 2, його функція є наглядовою, а не контрольною.

1.3.9 Рівень 0 – виробничі процеси

Рівень 0 – це місце, де знаходиться фактичне технологічне обладнання, яке контролюється з вищих рівнів. Даний рівень, також відомий як EUC (обладнання під контролем), відрізняється тим, що тут ми можемо знайти такі пристрої, як двигуни, насоси, клапани та датчики, які вимірюють швидкість, температуру чи тиск. Оскільки на рівні 0 виконується фактичний процес і виготовляється продукт, вкрай важливо, щоб усе відбувалося плавно та без перерв. Найменший збій в роботі одного пристрою може спричинити збій у роботі всіх операцій.

1.4 Контрольні запитання та завдання

1. Що таке АСУТП? Які основні складові входять до структури АСУТП?
2. Яку роль виконують програмовані логічні контролери (ПЛК) в АСУТП?
3. У чому полягає різниця між функціями спостереження, моніторингу та контролю?
4. Яке призначення панелі оператора в системі управління?
5. Для чого використовується система диспетчерського контролю та збору даних (SCADA)?
6. Чим розподілена система управління (DCS) відрізняється від системи на базі ПЛК?
7. Що таке система безпеки в структурі АСУТП?
8. У чому полягає сутність моделі Пердью для промислових систем управління?
9. Які рівні входять до виробничої зони згідно з моделлю Пердью?

2 ОГЛЯД МОЖЛИВИХ СЦЕНАРІЇВ ВТРУЧАННЯ У ДІЇ АСУТП

2.1 Спрощена структура мережі Пердью

Розглянемо спрощену структуру мережі Пердью для виконання аналізу можливих сценаріїв втручання в роботу АСУТП (рис. 2.1).

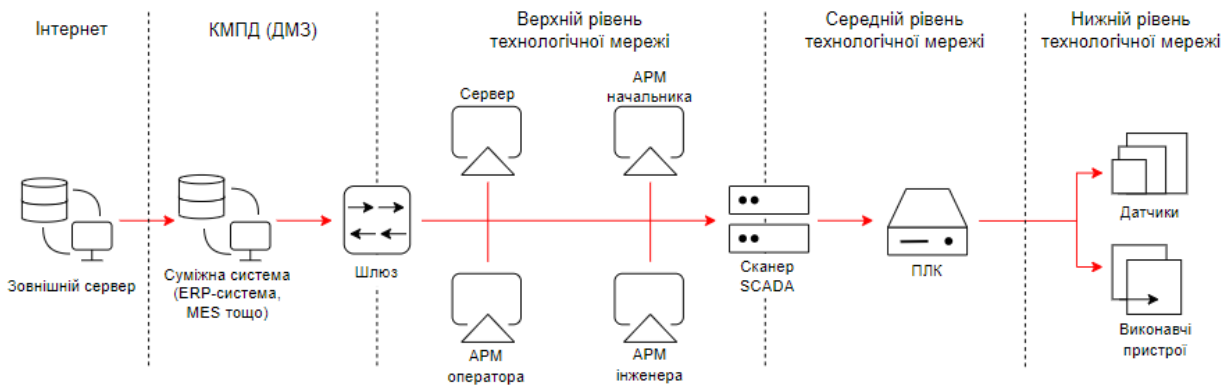


Рисунок 2.1 – Спрощена мережа АСУТП

У сегменті «Інтернет» розташований зовнішній сервер. Доступ в інтернет із корпоративної мережі є завжди. Іноді трапляються випадки, коли з технологічного сегменту мережі також є необмежений доступ до інтернету.

Наступний елемент – корпоративна мережа передачі даних (КМПД). Замість неї може бути демілітаризована зона (ДМЗ), яка знаходиться між корпоративним сегментом та технологічною мережею. У цьому сегменті розташована суміжна система. Фактично стандартом вважається ERP-система або MES.

Корпоративну та технологічні мережі поділяє шлюз.

АСУТП зазвичай складається з трьох рівнів технологічної мережі. На верхньому рівні знаходиться SCADA-система, включаючи сервер SCADA, автоматизовані робочі місця операторів, інженерів та начальника. Крім того, там знаходяться різні сервери, у тому числі сервери єдиного часу, повідомлень та тривог, історичних даних та інколи веб-сервери. Крім того, слід враховувати не лише ті ресурси, які з’явилися в процесі розгортання АСУТП, а й ті, що були додані до мережі пізніше. Сервер SCADA зазвичай має доступ до верхнього

рівня технологічної мережі через один інтерфейс, а через інший – у середній рівень, де розташовані програмовані логічні контролери (ПЛК). У ПЛК закладається основна логіка технологічних процесів.

На нижньому рівні технологічної мережі розташовуються два типи пристроїв: датчики (наприклад, тиску, температури тощо) та виконавчі пристрої (двигуни, роутери, заслінки).

2.2 Атака на технологічний сегмент через суміжні системи

На рис. 2.2 подана схема атаки на АСУТП через суміжні системи.

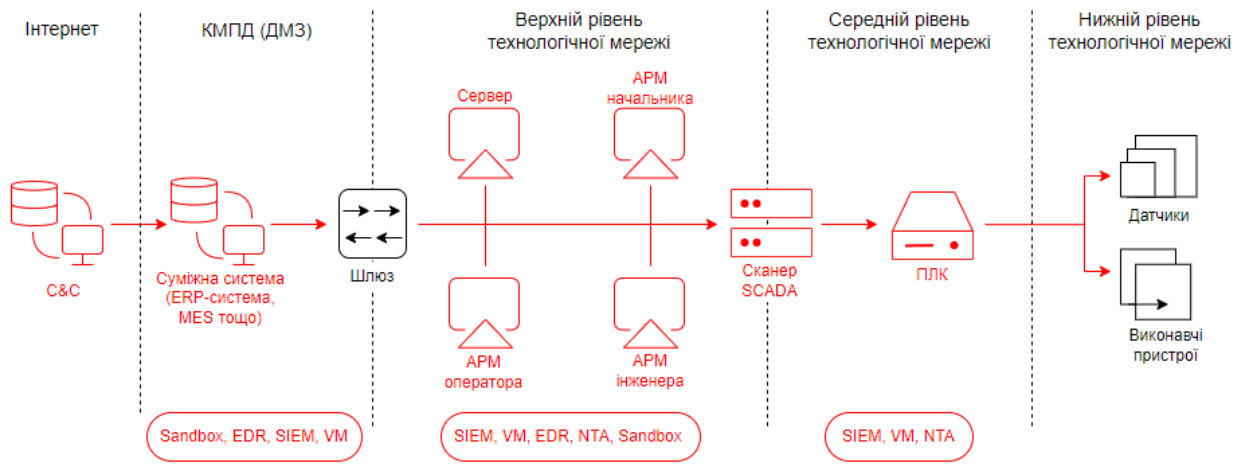


Рисунок 2.2 – Схема атаки на АСУТП через суміжні системи

Як видно з рис. 2.2, в сегменті «Інтернет» замість зовнішнього сервера з’явився командний центр атакуючого (C&C). Розглянемо варіанти компрометації суміжної системи корпоративної мережі.

Як правило, будь-яка атака на корпоративну мережу включає дві основні стадії:

– переміщення атакуючого від вузла до вузла підвищення привілеїв. Класичний приклад – отримання облікових даних адміністратора домену або підприємства Active Directory;

– переміщення атакуючого по мережі у спробі реалізувати власні завдання. Проводячи розвідку на вузлах, аналізуючи, які системи є та як їх вигідно можна використовувати, підвищуючи в них привілеї, зловмисник може атакувати

технологічний сегмент через корпоративну систему, яка має туди доступ, тобто суміжна.

Розглянемо класи засобів захисту, які виявлять компрометацію суміжної системи.

2.2.1 EDR система

Endpoint Detection and Response (EDR) – просунута система безпеки, що є інтегрованим рішенням для забезпечення безпеки кінцевих точок (комп'ютерних апаратних пристроїв) від потенційних загроз. Технологія поєднує в собі безперервний моніторинг та збір даних про кінцеві точки в режимі реального часу і вирішує проблему постійного моніторингу та реагування на складні загрози.

Антивірусне програмне забезпечення дозволяє вирішити ряд типових завдань – це усунення масових загроз [2]. Тоді як EDR-інструменти застосовуються для визначення цільових загроз і атак. Іншими словами, EDR це інструмент вузької спеціалізації, призначений для виявлення цілеспрямованих атак і складних загроз, які не детектуються звичайним антивірусним програмним забезпеченням.

EDR постійно контролює кінцеві точки, забезпечуючи швидке реагування та негайну відповідь. Рішення здійснює моніторинг подій кінцевої точки та мережі. Інформація записується в центральну базу даних, де відбувається аналіз, складання звітності та подальші дослідження.

Система для збору даних про кінцеві точки підвищує рівень захисту користувачів, що працюють за межами захисту периметра компанії, а також захищає від безфайлових та інших видів атак і навіть заражених USB-накопичувачів. Щоб усунути недоліки в безпеці корпоративної мережі, організації розгортають рішення для виявлення та реагування кінцевих точок (EDR) як доповнення до існуючих засобів захисту. Розглянемо приклади програмного забезпечення, що реалізує функції EDR.

Рішення Endpoint Threat Defense and Response від розробника Mcafee підійде для підприємців. Цей програмний комплекс полегшує роботу фахівців, що займаються питаннями інформаційної безпеки. Сучасні технології, просунуті інструменти і простий інтерфейс для адміністрування: все це є в Endpoint Protection від Malwarebytes. Ключовими перевагами можна вважати:

– можливість централізованого управління і реагування;

– вбудоване ядро для нейтралізації загроз;

– інтелектуальний алгоритм обміну інформацією між різними рівнями;

– простоту використання;

– Epolicy Orchestrator.

Крім того у Mcafee є цілий набір інформаційних панелей, що дозволяють швидко виявити загрозу або вразливість. Участі кінцевого користувача в забезпеченні безпеки практично немає, все здійснюється в автоматичному режимі.

Основною перевагою рішення SentinelOne можна вважати можливість дешифрування файлів після атаки програми-вимагача. Це єдине рішення на ринку, що дозволяє зробити щось подібне. Серед переваг цього рішення можна також виділити:

– ефективність розгортання в системі;

– кросплатформеність (можна розгорнути ПЗ практично на будь-якому пристрої);

– механізм глибинної аналітики.

Рішення вже зарекомендувало себе, і активно використовується великими компаніями по всьому світу.

Intercept X від Sophos це програмний продукт, що поєднує в собі багатий функціонал і тонке налаштування механізму детектування. Ключовими перевагами цього рішення можна вважати:

– функціонал, який об’єднує всі потоки даних (від телеметрії до мережевої інформації користувача і, навіть, його e-mail);

– систему ефективного виявлення загроз;

– алгоритм, що відслідковує сліди різних атак;

– протидія програмам-вимагачам.

Більш того, за допомогою Sophos можна отримувати файли з віддалених пристроїв, щоб піддати їх аналізу.

Однак EDR має проблемні зони:

– EDR не може забезпечити видимість кінцевої точки без агента EDR;

– для EDR потрібен персонал служби безпеки, навчений виявленню та реагуванню;

– EDR не забезпечує прозорість мережі. Внутрішні загрози можуть переміщатися через мережу в поперечному напрямку і непомітно спілкуватися з віддаленим сервером.

Згідно з Osterman Research [3], найбільш важливими причинами для розгортання рішення EDR є його здатність до захисту від безфайлових шкідливих програм, можливість покращення відновлення після злому та забезпечення покращеної телеметрії загроз у порівнянні зі звичайними рішеннями безпеки.

Типовий варіант використання EDR – коли активна загроза проявляється у кількох формах на кінцевій точці, дивлячись на шаблони дій, а не на прості сигнали, такі як конкретний вірус або порушення брандмауера. Наприклад, зловмисник, який краде дійсні облікові дані за допомогою фішингової атаки, може увійти в систему у звичайному режимі, не викликаючи жодних сигналів тривоги або використовуючи зловмисне програмне забезпечення.

EPP (Endpoint Protection Platform) виявляє погрози, але не вторгнення та приховані атаки [4]. Зловмисники можуть проникати в мережі непоміченими та залишатися в корпоративній мережі протягом тривалого часу у пошуках конфіденційних даних. Також кіберзлочинці навчилися уникати виявлення антивірусу, і вони можуть керувати скомпрометованим активом, не викликаючи підозри. Ключовою проблемою, яку необхідно вирішити підприємствам, є відображення критичних попереджень для легкого розуміння проблеми або загрози та розставлення пріоритетів для наступних кроків.

При використанні традиційних рішень та засобів захисту мережі аналітикам безпеки часто не вистачає інформації, необхідної для повного розуміння порушення. Співробітникам також бракує можливості аналізувати загрози та проводити ретельні розслідування. Через відсутність необхідних інструментів, усунення загроз триває тривалий час.

Під час використання багатьох традиційних рішень безпеки не записується достатній обсяг інформації про активність кінцевих точок. Ця інформація не зберігається або недоступна, а інструменти аналізу відсутні. Рішення EDR усувають недостатню прозорість, забезпечуючи можливості аудиту та пошуку загроз, а також глибше розуміння тактики, методів та процедур зловмисників.

EDR ґрунтується на можливостях традиційних рішень безпеки, допомагаючи з проблемами:

– відсутності виявлення загроз після злому;

– відсутності можливості глибокого огляду та аналізу.

Рішення EDR вирішують кожну з перерахованих проблем завдяки моніторингу всіх дій кінцевих точок для виявлення: цільових та складних загроз,

горизонтального переміщення в мережі, використання крадених облікових даних, інсайдерської активності та інших аномальних дій з боку зловмисників.

Endpoint Detection and Response записує всі події, що відбуваються на кінцевих точках та в мережі, для надання вичерпних даних для подальших розслідувань та виправлення інцидентів безпеки. Система EDR вирішує не лише технічні проблеми, пов'язані з моніторингом та аналізом загроз, але й забезпечує низку переваг для бізнесу.

Рішення EDR пропонують важливі переваги:

– безперервний моніторинг широкого діапазону кінцевих пристроїв у корпоративних мережах та за їх межами дозволяє організаціям відстежувати не тільки зловмисні атаки із зовнішніх джерел (наприклад, постійні загрози APT, які можуть призвести до витоків даних), але також для відстеження аномальної активності всередині організації (наприклад, видобутку криптовалюти чи крадіжки даних співробітниками);

– застосування штучного інтелекту для моніторингу систем на предмет зловмисної активності з метою попередження атак на етапі підготовки та під час їх реалізації;

– запис великих обсягів активності в мережі, на відміну від інших інструментів, таких як SIEM та платформи захисту кінцевих точок, які не записують або зберігають малу кількість інформації;

– інтеграція з розширеними функціями, такими як пісочниця (Sandbox), для пошуку сплячих загроз;

– увімкнення запобіжного пошуку індикаторів атаки, щоб побачити загрози, які ще не були виявлені.

– розширені можливості аналізу, що дозволяють командам безпеки швидше оцінювати та блокувати наступні атаки. До них належать: пошук індикаторів компрометації; випереджальний пошук індикаторів атаки та визначення першопричини зараження; кіберінцидент та увімкнення захисту від нього;

– виправлення наслідків здійснених атак із можливістю відкату кінцевих точок до раніше відомого справного стану;

– створення деталізованих політик для обробки USB-пристроїв з метою блокування невідомих та потенційно шкідливих USB-ключів;

– увімкнення захисту віддалених співробітників, які не можуть покладатися на захист периметра.

Крім того, рішення EDR можуть забезпечити бізнес-переваги, переконавши регулюючі органи, співробітників відділу нормативно-правової відповідності, клієнтів та інших осіб, що організація, яка розгортає рішення EDR, серйозно ставиться до своєї безпеки. Рішення EDR може продемонструвати, що погрози будуть ретельно відстежуватися, дуже докладна інформація про події кінцевих точок зберігатиметься протягом відповідного періоду часу, а усунення погроз безпеки буде відбуватися якнайшвидше.

Активне впровадження EDR обумовлюється необхідністю доповнення традиційних антивірусних і EPP-рішень, інтеграція з якими дозволяє забезпечити ефективніший захист, покращену звітність і розширені функціональні можливості.

2.2.2 Мережева пісочниця

Основним рішенням для захисту від складних цілеспрямованих атак (Advanced Persistent Threats, APT) є мережева пісочниця (Sandbox) [5]. Принцип її роботи полягає в емуляції на віртуальній машині, образі операційної системи, тобто робочому середовищі, куди поміщається підозрілий файл та аналізується його поведінка.

Вона допомагає виявити в файлі або трафіку невідповідності, аналізуючи його дії в системі, і видає вердикт [6]. Всі програми будуть працювати всередині антивіруса, при цьому будь-які зміни залишаться там, не торкнувшись загальної системи. Користувач може виконувати звичні функції, пов’язані з відвідуванням порталів, посилань, завантаженням файлів і багато іншого у звичайному режимі. Якщо з’являються шкідливі ознаки, достатньо видалити вміст утиліти.

Якщо після аналізу з’ясується, що файл має потенційну загрозу, тобто є шкідливим, то інформація передається на всі пристрої безпеки, щоб мінімізувати шкідливий вплив.

Ключові особливості пісочниці полягають в комплексній перевірці файлів, що включає в себе сигнатурний, статичний і динамічний аналіз. Слід виділити такі основні функції утиліти:

– виявлення і запобігання дій шкідливих програм;

– перевірка на наявність загроз з урахуванням можливостей хмарного сервісу;

– швидкий обмін відомостей про небезпечні об’єкти;

– відстеження загроз і підготовка звіту.

Функціонування пісочниці також полягає у результаті аналізу, генеруванні антивірусної сигнатури з блокуванням всього сімейства шкідливих об'єктів у віртуальному середовищі, захисті персональних даних, виведення результатів на мережевий або міжмережевий екран. Застосування даного продукту безпеки пов'язано з такими перевагами, як захист інтернет-серфінгу, поліпшення приватності, безпечне листування, запуск будь-яких клієнтських додатків, месенджерів. Програма здійснює контроль потенційно незахищених джерел.

2.2.3 Система класу SIEM

Система класу SIEM виявляє підозрілу активність користувачів та програм усередині кожного активу та являється ядром будь-якого SOC.

Система або процес управління вразливістю (Vulnerability Management) дозволяє виявляти критично небезпечні вразливості в інфраструктурі (для нашого випадку – в суміжній системі) та усувати їх.

2.3 Контрольні запитання та завдання

1. Які основні сценарії втручання в роботу АСУТП розглядаються в розділі?
2. У чому полягає сутність моделі мережі Пердью та які її рівні?
3. Яке призначення спрощеної структури мережі Пердью в контексті кібербезпеки?
4. Які вразливості можуть виникати на різних рівнях архітектури Пердью?
5. Як може здійснюватися атака на технологічний сегмент через суміжні інформаційні системи?
6. Яку роль відіграє система класу EDR у виявленні та запобіганні кібератакам?
7. Що таке мережева пісочниця та які її функції в системах захисту АСУТП?
8. Які можливості надають системи класу SIEM для моніторингу та аналізу подій безпеки?
9. Як взаємодіють між собою EDR, мережева пісочниця та SIEM у процесі виявлення загроз?

3 ПРОТОКОЛИ ПРОМИСЛОВОГО ІНТЕРНЕТУ РЕЧЕЙ

3.1 Рівні архітектури IIOT

В процесі розробки засобів автоматизації, що використовуються в мережі IIOT пристроїв, необхідно враховувати особливості побудови даної мережі в концепції Інтернету речей. На рис. 3.1 подані відмінності між традиційними інтернет протоколами та тими, що застосовуються в IIOT.

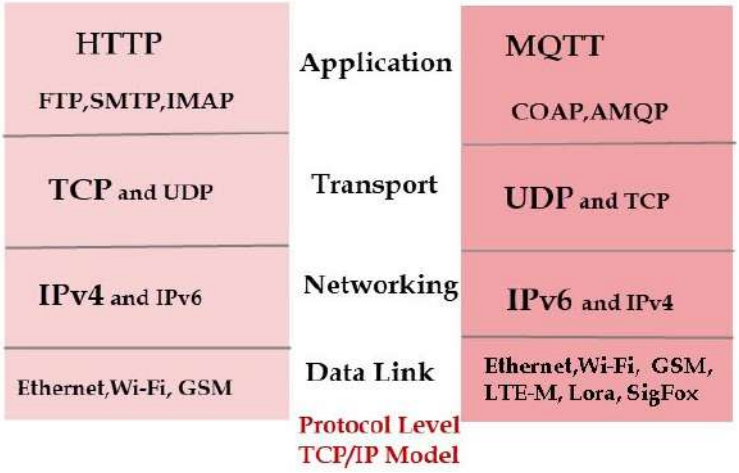


Рисунок 3.1 – Відмінності між традиційними інтернет протоколами та тими, що застосовуються в IIOT

На відміну від типової WEB-системи, що працює за принципом клієнт-серверної програми, модель IIoT-рішень набагато складніша. Аналогічно клієнт-серверному принципу в IIoT-архітектурі можна виділити дві різні за фізичним розташуванням групи обов'язкових компонентів:

– периферія (Edge) – це кінцеві smart-пристрої, розташовані на технологічному обладнанні, за яким здійснюється віддалений моніторинг та керування;

– потужні Big Data інструменти, що розгорнуті у центрі обробки даних на серверах чи в хмарі (Backend).

Тим не менш, через особливості використання IIoT-рішень, пов'язаних з умовами експлуатації та прикладної специфіки, наприклад, екстремальні температури, вібрації, опади, велика довжина каналів передачі даних тощо,

в архітектурі Industrial Internet of Things можна виділити цілих 12 рівнів (шарів) (рис. 3.2).

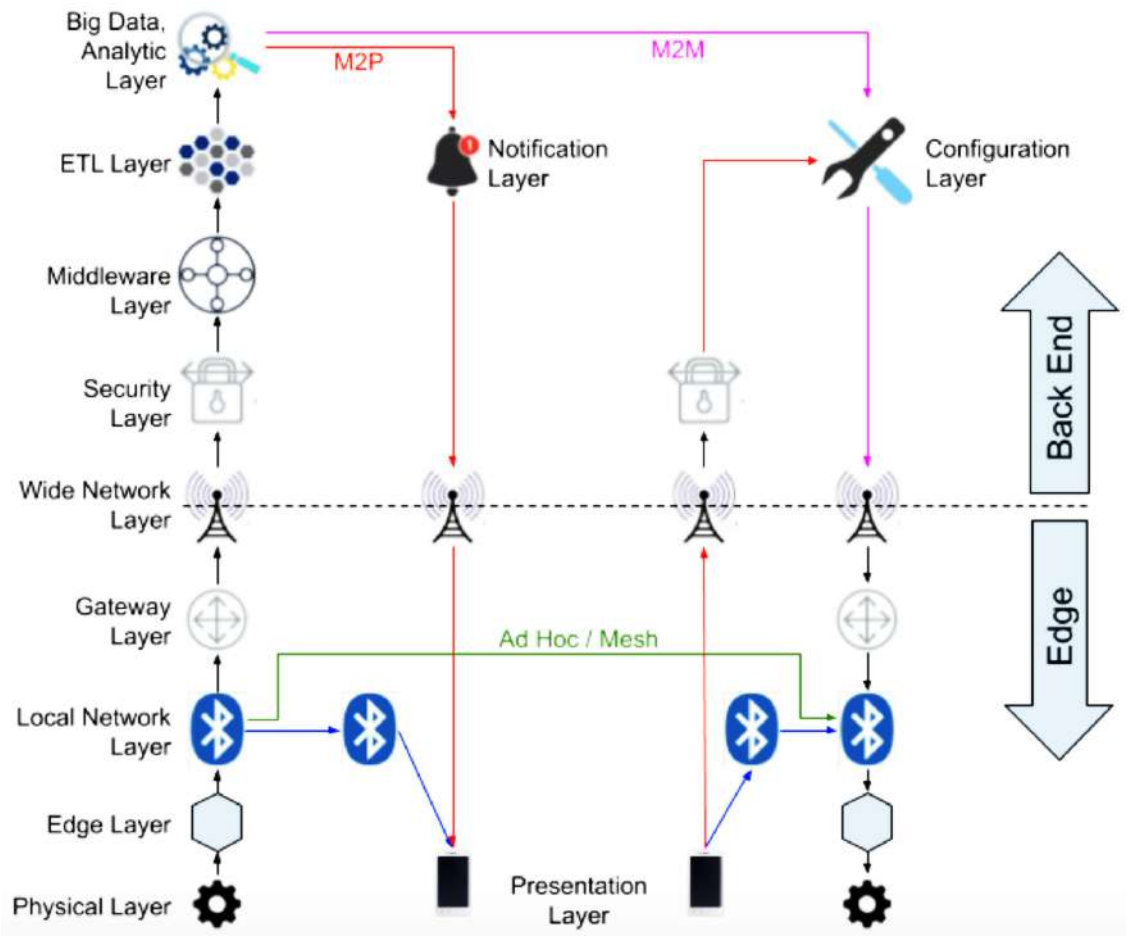


Рисунок 3.2 – Рівні архітектури IIOT

На фізичному рівні (Physical Layer) розташовані кінцеві пристрої – датчики та сенсори, камери та інше обладнання (таймери, п’єзоелементи, мікрофони, фотодіоди / транзистори / резистори, вимикачі, акселерометри, GPS-трекери, баркод-рідери тощо), яке збирає інформацію та надсилає її далі, на рівень периферійної обробки даних (Edge Layer).

Периферійні обчислення (Edge Layer) забезпечують мінімальну обробку даних, зокрема перетворення між аналоговим та цифровим поданням інформації (АЦП / ЦАП). Як правило, ці функції реалізуються за допомогою мікроконтролерів Raspberry Pi, Arduino та інших однокристальних систем SoC-модулів (System-on-a-Chip) – електронні схеми, що виконують функції цілого пристрою, включаючи бездротові мережі на кристалі (Wireless network-on-chip, WNOC).

Основною вимогою до пристроїв фізичного та периферійного рівнів є низьке енергоспоживання (живлення від батарейки), низька вартість закупівлі та експлуатації (безперебійна робота без обслуговування від 1 до 10 років, мінімальні витрати на придбання, встановлення та обслуговування). Для зниження енергоспоживання периферійні пристрої зазвичай мають чотири режими роботи: сон, режим вимірювання та збору інформації з датчиків, режим зв'язку, передачі та отримання інформації, а також режим встановлення та підключення.

Периферійна комунікація (Local Network Layer) застосовується коли дані після первинної обробки вирушають на шлюз (Gateway), який може бути дуже далеко (але не більше кількох кілометрів). Для надійної передачі даних на далекі відстані застосовуються бездротові динамічні мережі, що самоорганізуються: однорангова Ad Hoc і ієрархічна Mesh. Зазвичай при цьому використовуються протоколи ZigBee/Zwave, BLE, LoRa, Proprietary low band, WiFi, Ethernet, CAN, LTE, PLC (рис. 3.3).

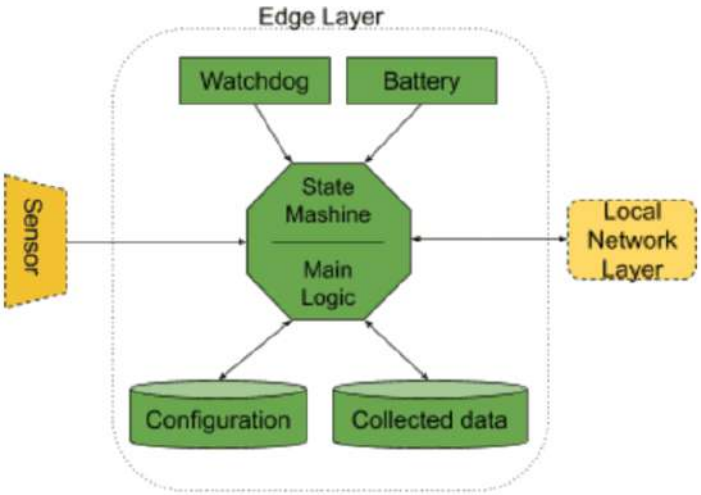


Рисунок 3.3 – Схема периферійного IIoT-пристрою

Шлюз (Gateway Layer), який виконує ETL-операції (Extract, Transform, Load) – отримання, перетворення та збереження інформації з периферійних пристроїв, забезпечує важливі дії у разі критичної ситуації навіть без зв'язку з Backend, а також спілкується із сервером з використанням мобільного бездротового зв'язку (4G / 5G) або дротового доступу до Інтернету, надсилаючи туди оброблену інформацію та одержуючи дані конфігурації для кінцевих пристроїв.

Шлюз необхідний IIoT-рішенням, тому що якщо Backend отримуватиме необроблену інформацію, це збільшить його потужність та комунікаційні витрати. Крім того, віддалений Backend-сервер не може гарантувати реакцію в реальному часі для великої кількості периферійних пристроїв, наприклад, камери вуличного спостереження тощо.

Рівень зовнішнього зв'язку (Wide Network Layer) розділяє периферію та BackEnd. Тут використовується стандартизований протокол для IoT-рішень LwM2M (Lightweight M2M, легковажний міжмашинний протокол), розроблений для доступу до кожного периферійного пристрою. Якщо периферійний пристрій не підтримує інтерфейс LwM2M, шлюз вирішить цю проблему, забезпечивши зв'язок з периферією. Рівень зовнішнього зв'язку містить також комунікаційні послуги та моделі ISO, включаючи служби балансування та визначення розташування, засновані на DNS-сервісі, транспортний протокол COAP, протокол шифрування з ключами безпеки та сесією встановлення з'єднання DTLS та багато інших компонентів. При цьому DNS використовується як балансувальник навантаження, періодично генеруючи DNS-запит для отримання нової IP-адреси екземпляра рівня безпеки. Крім LwM2M лише на рівні зовнішнього зв'язку також використовуються протоколи PLC, Ethernet, LTE, FOTA/SOTA, Radio.

Рівень безпеки (Security Layer) забезпечує автентифікацію, авторизацію, облік та шифрування / дешифрування на основі ролей та дозволів. Як правило, тут використовуються засоби інформаційної безпеки від хмарних провайдерів, у яких розгорнуті компоненти IIoT-рішень, що агрегують і аналізують, наприклад, Azure Cloud, AWS R53/IAM/EC2 тощо.

Рівень внутрішньосерверного зв'язку (Middleware Layer), що забезпечує внутрішню функціональність балансування навантаження у хмарі, черги повідомлень та передачі потокової інформації на основі мікросервісів або PaaS-рішень від хмарних провайдерів. Компоненти цього шару повинні бути дубльовані та автоматично масштабуватися у зв'язку з нестаціонарним характером передачі даних, щоб забезпечити асинхронну передачу повідомлень з буферизацією та перерозподілом навантаження. Саме на цьому рівні працюють Big Data брокери повідомлень та управління чергами, такі як Apache Kafka або RabbitMQ. Для швидкого завантаження даних із периферії часто використовується платформа обробки подій (повідомлень) Apache NiFi або її спрощена модифікація Apache MiNiFi.

Рівень збору, обробки та зберігання даних (ETL), що реалізується не тільки в периферійних пристроях та шлюзах, а й у Backend, за допомогою збору даних, їх агрегації, уніфікації подання, збереження для подальшого використання, повідомлення інших сервісів про надходження нових даних та загального управління життєвим циклом інформації, у тому числі архівування та знищення. Тут також працюють Big Data-засоби управління чергами повідомлень (Apache Kafka або RabbitMQ) та потокове завантаження даних (Apache NiFi та MiNiFi).

Big Data аналітика, машинне навчання та інші методи штучного інтелекту (Artificial Intelligence, AI). Інструментальні засоби цього рівня найменш стандартизовані та залежать від конкретного випадку. Наприклад, для прогнозування можливих відмов обладнання може використовуватися бібліотека MLLib фреймворків Spark або Flink, для генерації OLAP-кубів та складних SQL-запитів – Spark SQL та FlinkQL відповідно.

Рівень повідомлень (Notification layer), що повідомляє про настання якоїсь події, наприклад, у формі листа електронною поштою, SMS-повідомлення або телефонного дзвінка (у крайньому, екстреному випадку). Для зниження енергоспоживання мобільні програми переходять у сплячий режим, але iOS та Android мають механізм повідомлень, що вказує на прибуття нових даних. Алгоритмічно таке M2P-спілкування (machine to person) реалізується у вигляді моделі «видавець-передплатник», коли клієнтська програма підписується на необхідні події і, у разі їх настання, отримує інформаційний сигнал – повідомлення.

Рівень представлення (Presentation Layer) застосовується коли оброблена інформація від периферійних пристроїв презентується кінцевому користувачеві за допомогою UI/UX-інтерфейсів на екрані комп’ютерного монітора, планшета або мобільного телефону. Рівень подання також відповідає за обслуговування, конфігурацію та зміни стану кожного компонента IIoT-системи, дозволяючи віддалено керувати навіть периферійними пристроями. Поки не існує стандартизованих UI/UX-засобів для рівня представлення IIoT-системи, незважаючи на наявність міжнародних та національних стандартів, які регламентують поняття людино-машинного інтерфейсу, що широко використовуються в SCADA-системах.

Рівень конфігурацій (Configuration Layer) забезпечує сховище статусів периферійних пристроїв (актуальний стан, новий стан, який буде завантажено та проміжний статус, що вказує на процес оновлення стану). Це необхідно через

особливості зв’язку периферії та шлюзу з Backend-сервером, тому що комунікація кінцевих пристроїв з хмарою тримається не завжди, а періодичними сеансами, під час яких відбувається відправлення та отримання даних.

3.2 Основні протоколи організації зв’язку в IIOT

Серед багатьох протоколів, що застосовуються для організації обміну повідомленнями в IIOT можна виділити декілька тих, що застосовуються найчастіше. До них відносяться:

– DDS OMG;
– AMQP;
– MQTT;
– JMS;
– REST;
– CoAP;
– XMPP.

Data Distribution Service (DDS) – це протокол проміжного програмного забезпечення та стандарт API для з’єднання, орієнтованого на дані, від Object Management Group (OMG). Він об’єднує компоненти системи разом, забезпечуючи підключення до даних з низькою затримкою, надзвичайну надійність і масштабовану архітектуру, яка потрібна для бізнесу та критично важливих додатків Інтернету речей (IoT). Це протокол мережевої комунікації, розроблений для використання в розподілених системах реального часу (Real-Time Distributed Systems). DDS забезпечує механізми розподілу даних та спільного використання даних між різними системами та процесами в реальному часі.

Даний протокол був розроблений з урахуванням вимог, що ставляться до систем реального часу, де час від реакції на події може залежати від мікросекунд до мілісекунд. DDS надає механізми забезпечення надійності та якості обслуговування (Quality of Service), таких як маршрутизація повідомлень, контроль доступу та механізми синхронізації.

DDS використовує поняття тем, що дозволяє підписникам підписуватися на теми, які вони хочуть отримувати, і видавцям використовувати ці теми для відправлення повідомлень (рис. 3.4). Протокол також дозволяє підписникам створювати власні теми та публікувати на них повідомлення.

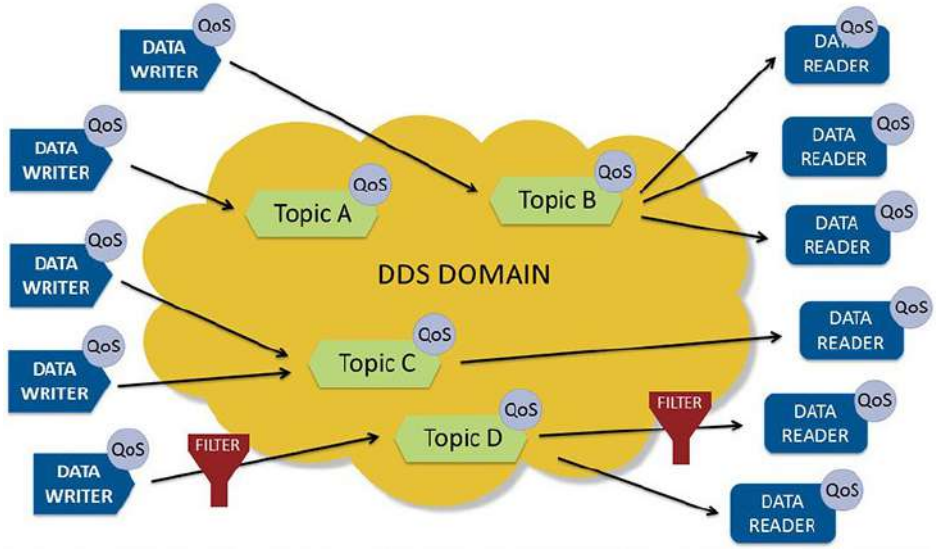


Рисунок 3.4 – Контрольований обмін даними за допомогою протоколу DDS

DDS забезпечує контрольований Qos обмін даними. Програми спілкуються, публікуючи теми та підписуючись на них. У підписках можна вказати фільтри часу та вмісту та отримати лише частину даних, які публікуються в темі. Різні домени DDS повністю незалежні один від одного. Немає спільного використання даних між доменами DDS.

DDS широко використовується в різних галузях, включаючи авіацію, оборонну та космічну промисловість, медицину, телекомунікації та виробничі системи.

AMQP (Advanced Message Queuing Protocol) – це відкритий стандарт протоколу передачі повідомлень, який призначений для взаємодії між різними програмними системами. Протокол забезпечує безпеку, надійність та масштабованість передачі повідомлень.

Цей протокол був створений для вирішення проблеми інтеграції між різними протоколами, платформами та мовами програмування. Він дозволяє відправляти та отримувати повідомлення з одного додатка в інший, що дозволяє програмістам легко створювати розподілені системи.

AMQP пропонує поведінку провайдера повідомлень і клієнта в тій мірі, що рішення від різних постачальників справді сумісні. Він є двійковим протоколом рівня програми, призначеним для ефективної підтримки широкого спектру програм для обміну повідомленнями та патернів (шаблонів). Він забезпечує потік контрольованих повідомлень з гарантіями доставки

повідомлень, такі як AT-MOST-ONCE (де кожне повідомлення доставляється один раз або ніколи), AT-LIST-ONCE (де кожне повідомлення буде доставлено, але може виявитися так, що це відбудеться декілька разів) і EXACTLY-ONCE (коли повідомлення буде завжди доставлено тільки один раз) та автентифікацію або шифрування на основі SASL та TLS. Це передбачає використання надійного протоколу транспортного рівня, як-от протокол управління передачею (TCP).

Брокер (або сервер) відіграє вирішальну роль у активації протоколу AMQP. Він відповідає за створення з'єднань, які забезпечують кращу маршрутизацію даних і чергування на стороні клієнта. Робота з формування черг і підтвердження повідомлень виконується споживачем.

Спрямування даних, отриманих з бірж, і розміщення їх у чергах здійснюється виробником (рис. 3.5).

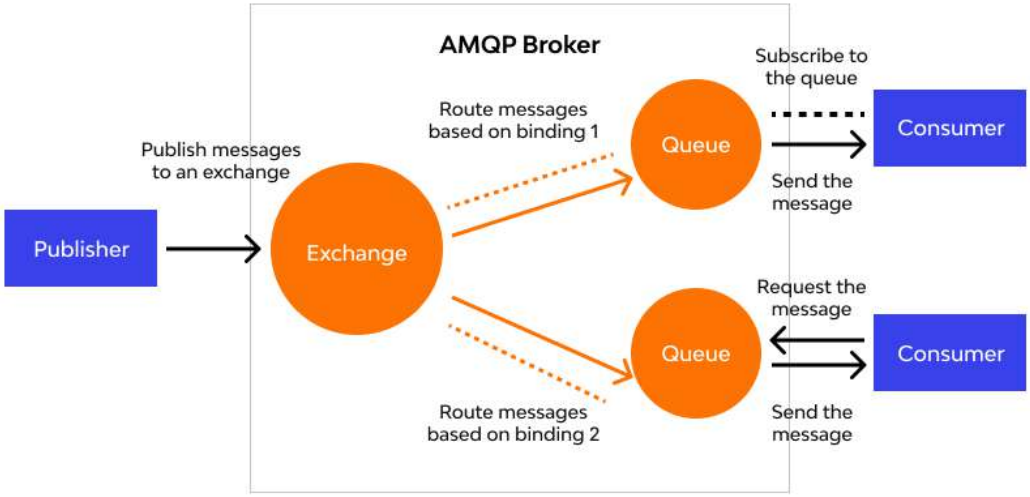


Рисунок 3.5 – Принцип взаємодії компонентів за протоколом AMQP

Протокол AMQP складається з наступних компонентів:

– клієнт – програма, що відправляє та отримує повідомлення;

– брокер повідомлень – посередник, який приймає повідомлення від клієнтів та передає їх до інших клієнтів. Брокер може зберігати повідомлення у черзі до того часу, поки вони не будуть оброблені;

– повідомлення – носій інформації, який передається між клієнтами;

– канали – пов'язують клієнтів з брокером та дозволяють розділити потік повідомлень на різні теми;

– правила маршрутизації – правила, що вказують як повідомлення має бути доставлено до клієнта. Можна налаштувати правила маршрутизації, щоб направляти повідомлення на певний канал або до певного клієнта;

– моделі повідомлень – моделі, що вказують як повідомлення має бути оброблено. Наприклад, можна використовувати модель «request-response», щоб відправити запит та отримати відповідь від іншого клієнта.

Ці компоненти дозволяють програмістам легко створювати розподілені системи, які можуть передавати повідомлення між різними компонентами. Крім того, AMQP є відкритим стандартом, що дозволяє різним системам взаємодіяти між собою без обмежень з боку вендора.

Протокол MQTT (Message queuing telemetry support) визначається як міжмашинний протокол з низьким споживанням пропускної здатності, який допомагає пристроям IoT спілкуватися один з одним, з мінімальними вимогами до коду та площею мережі (рис. 3.6).

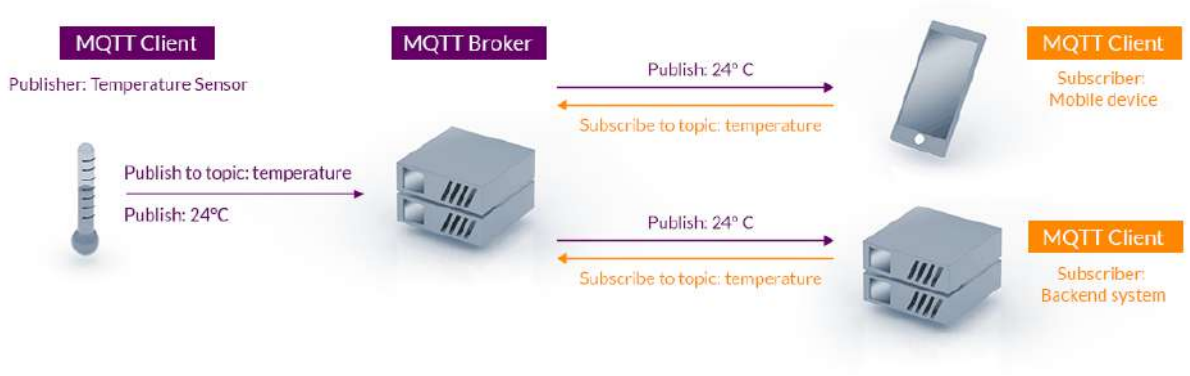


Рисунок 3.6 – Принцип взаємодії між пристроями за протоколом MQTT

MQTT – це розумне рішення для бездротових з’єднань із різними рівнями затримки через періодичні обмеження широкосмугового зв’язку або ненадійні з’єднання. Він підходить для підключення пристроїв з невеликим кодом. Стандарт використовується в багатьох галузях промисловості, включаючи автомобілі, енергетику та зв’язок.

Message Queuing Telemetry Transport підходить для зв’язку між машинами (M2M) через його оптимізацію для середовищ із низькою пропускною здатністю та високою затримкою.

MQTT – це протокол видавця / передплатника, яким керує основний брокер. Це означає, що між передавачем і отримувачем немає прямого зв’язку.

Незважаючи на те, що MQTT часто згадують як телеметричний транспорт із чергою повідомлень, спілкування MQTT не використовує черги повідомлень. Оскільки всі отримувачі, зацікавлені в конкретних повідомленнях («позначених темою»), зараховуються як користувачі, джерела інформації публікують її, і всі отримувачі, які бажають одержати конкретні повідомлення («позначені темою»), отримують доступ до цієї інформації. MQTT використовується для всього, від IoT та IIoT до зв'язування хмарних середовищ в IoT та IIoT.

Java Message Service (JMS) – це стандарт проміжного програмного забезпечення для розсилки повідомлень, що дозволяє standalone або веб-додаткам на платформі Java створювати, надсилати, отримувати та читати повідомлення в рамках інтеграції інформаційних систем. При цьому відправлення повідомлень виконується в асинхронному режимі, тобто, відправник не чекає на відповідь від отримувача (рис. 3.7).

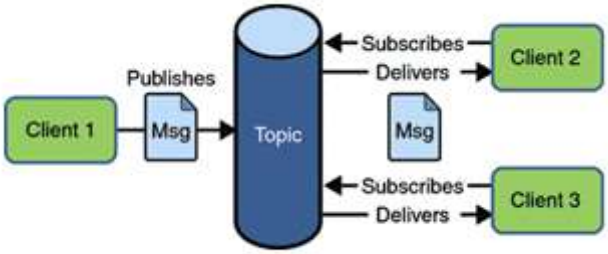


Рисунок 3.7 – Принцип взаємодії між пристроями за протоколом JMS

Отримувачі можуть бути таких типів:

– MQ (Message Queue) – проміжне програмне забезпечення для повідомлення (Message Oriented Middleware), яке дозволяє незалежним і не обов'язково синхронним додаткам у розподіленій системі обмінюватися даними один з одним через канал обміну повідомленнями;

– WEB-контейнер типу JBoss, GlassFish або іншого J2EE-сервера програм з відкритим вихідним кодом, який забезпечує обмін повідомленнями між програмами контейнера або мережею за допомогою JNDI (Java Naming and Directory Interface), API для доступу до служб імен та каталогів.

Крім Rabbit MQ, з яким найчастіше порівнюють Apache Kafka, до JMS-брокерів відносяться Open MQ, Apache ActiveMQ, OpenJMS, JBoss Messaging, Glassfish, TIBCO EMS, Sonic MQ, IBM MQ та інші відкриті та пропрієтарні рішення.

JMS підтримує дві моделі обміну повідомленнями:

– модель «черга» (point-to-point), коли кожне повідомлення має лише одного адресата. Повідомлення надсилається до черги, яка виконує роль своєрідної «поштової скриньки», і може бути зчитане отримувачем у будь-який зручний час. Якщо адресат не був активним у момент надсилання, повідомлення зберігається в черзі та не втрачається. Після отримання повідомлення система може сформувати підтвердження про його доставку. Для приймання повідомлень отримувач повинен періодично підключатися до відповідної черги та зчитувати наявні повідомлення;

– модель «видавець-підписувач» (publish-subscribe), коли підписник оформлює підписку на певний топік, до якого видавець публікує повідомлення. Опубліковане повідомлення отримують усі активні підписники цього топіка. У такій моделі програма-отримувач має бути запущеною та підписаною на топік у момент надсилання повідомлення.

REST – мігрував в IOT як модель проєктування Web API. Архітектура REST умовно складається з клієнтів і серверів. Клієнти ініціюють запити до серверів, сервери обробляють запити та повертають відповідні відповіді. Запити та відповіді будуються навколо передачі представлених ресурсів (рис. 3.8).

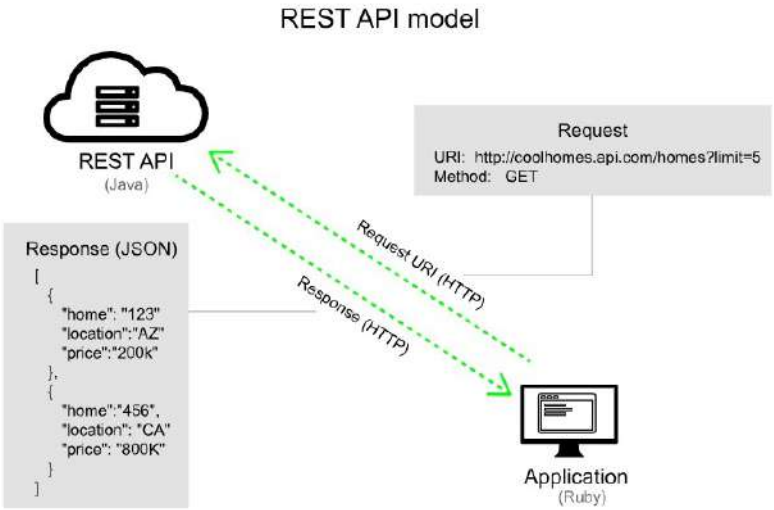


Рисунок 3.8 – Принцип обміну повідомленнями за протоколом REST

API, що використовують протокол HTTP для передавання запитів і відповідей, належать до вебсервісів. У таких системах клієнт, який звертається до ресурсу, і сервер API, що формує відповідь, можуть бути реалізовані

з використанням будь-яких мов програмування та платформ. Це не має принципового значення, оскільки взаємодія між сторонами здійснюється через універсальний веб-протокол HTTP, який забезпечує сумісність і незалежність від технологічної реалізації.

Веб-протокол є частиною веб-сервісів: вони незалежні від мови і тому сумісні між різними платформами та системами. При документуванні REST API не має значення, чи будують інженери API за допомогою Java, Ruby, Python або іншої мови. Запити виконуються через HTTP і відповіді повертаються через HTTP.

Constrained Application Protocol (CoAP) – це протокол, який було розроблено для передачі даних у мережах з обмеженими ресурсами, таких як IoT. CoAP базується на протоколі UDP і забезпечує передачу даних між клієнтом та сервером.

CoAP – звичайний клієнт-серверверний протокол IoT. Він дозволяє клієнтам робити запити на веб-перекази відповідно до потреби. З іншого боку, це також дозволяє допоміжним серверам відповідати на запити, що надходять. Таким чином, вузли пристроїв в екосистемі IoT можуть взаємодіяти лише через CoAP.

CoAP і HTTP працюють за подібною концепцією запит-відповідь. Проте CoAP реалізує свою функціональність через асинхронні транзакції з використанням протоколу UDP, що робить його більш придатним для ресурсно-обмежених пристроїв. Як і HTTP, CoAP підтримує методи POST, GET, PUT і DELETE. Підвищений рівень безпеки при використанні CoAP забезпечується механізмами захисту, зокрема підтримкою RPK (Raw Public Key) та PSK (Pre-Shared Key), які застосовуються для автентифікації та шифрування переданих даних.

CoAP має чотири типи обміну інформацією:

– запит (Request) – це повідомлення від клієнта до сервера, яке містить запит на певну інформацію;

– відповідь (Response) – це повідомлення від сервера до клієнта, яке містить відповідь на запит;

– сповіщення (Notification) – це повідомлення від сервера до клієнта, яке містить оновлення стану або даних, що стосуються клієнта;

– підписка (Subscription) – це повідомлення від клієнта до сервера, що запускає механізм підписки на сповіщення від сервера.

За допомогою цих типів обміну інформацією, CoAP забезпечує зв'язок між обмеженими пристроями та хмарами та дозволяє передавати дані в обмеженому мережевому середовищі з обмеженими ресурсами.

Архітектура CoAP складається з декількох компонентів (рис. 3.9):

– клієнт (Client) – пристрій, який ініціює запити до сервера;

– сервер (Server) – пристрій, який надає відповіді на запити від клієнта;

– Proxy – пристрій, який здійснює пересилання запитів від клієнта до сервера або від сервера до клієнта;

– Resource Directory – директорія ресурсів, яка зберігає інформацію про доступні ресурси на серверах.

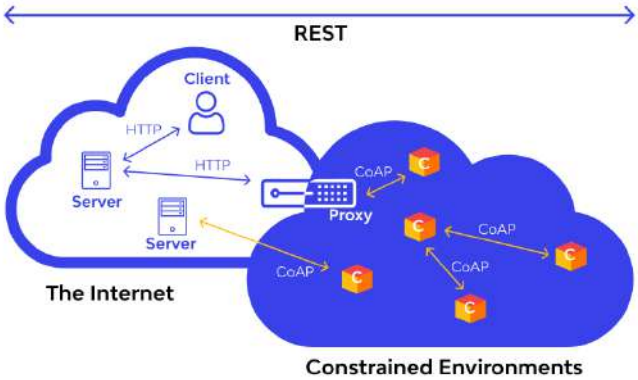


Рисунок 3.9 – Архітектура CoAP

Основним елементом архітектури CoAP є ресурс (Resource). Ресурс може бути створений на сервері і мати унікальну адресу (URI). Клієнт може звертатися до цього ресурсу за допомогою запиту GET або POST і отримувати відповідь від сервера. Ресурс може мати додаткові атрибути, такі як назва, тип та стан, які дозволяють клієнту дізнатися більше про ресурс і його поточний стан.

Оскільки CoAP є протоколом з обмеженими ресурсами, він має декілька обмежень. Наприклад, він не підтримує безпеку на рівні протоколу, але це можна вирішити за допомогою протоколів, таких як DTLS або IPSec. Крім того, він не підтримує повідомлення більшого розміру, ніж MTU мережі.

Протокол XMPP (Extensible Messaging and Presence Protocol) – це відкритий стандарт комунікації в режимі реального часу для обміну повідомленнями, створення мережевих служб та роботи з мережевими пристроями. Він базується на протоколі XML і зазвичай використовується для

створення мережевих додатків для обміну повідомленнями в режимі реального часу.

Основні характеристики протоколу XMPP:

– відкритий стандарт, що дає змогу кожному створювати власні мережеві додатки та розширювати функціонал протоколу;

– підтримує різні методи автентифікації, такі як ім’я користувача та пароль, SSL-сертифікати, SASL (Simple Authentication and Security Layer) та інші;

– дає змогу користувачам керувати своєю присутністю в мережі та обмінюватися цією інформацією з іншими користувачами;

– дозволяє користувачам обмінюватися повідомленнями в режимі реального часу;

– дозволяє розширювати функціонал протоколу за допомогою стандартів, які приймаються спільнотою розробників;

– може бути використаний в мережах будь-якого розміру, від невеликих локальних мереж до глобальних мереж з мільйонами користувачів;

– підтримує шифрування за допомогою TLS (Transport Layer Security) та SSL (Secure Sockets Layer), що забезпечує безпеку обміну повідомленнями;

– підтримує різні методи автентифікації, такі як ім’я користувача та пароль, механізм SASL (Simple Authentication and Security Layer) та інші;

– підтримує маршрутизацію повідомлень, що дозволяє користувачам встановлювати зв’язок з іншими користувачами через різні мережі та протоколи;

– дозволяє користувачам об’єднуватися в групові чати з можливістю передачі повідомлень в реальному часі.

Архітектура XMPP ґрунтується на розподіленій архітектурі клієнт-сервер (Client-Server Architecture), що дозволяє розробляти масштабовані системи з безліччю користувачів та серверів (рис. 3.10).

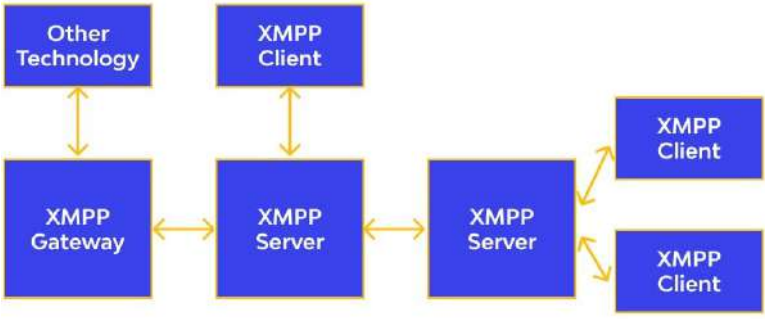


Рисунок 3.10 – Архітектура XMPP

XMPP складається з двох основних компонентів: клієнти та сервери. Клієнти – це програми для обміну повідомленнями у реальному часі, а сервери – це програми, що забезпечують мережеву інфраструктуру для обробки повідомлень та керування присутністю.

XMPP використовує модель взаємодії «запит-відповідь» (Request-Response Interaction Model) для обміну повідомленнями та станом присутності. Клієнти та сервери взаємодіють за допомогою XML-повідомлень, що передаються через мережу. XMPP також містить протоколи для обміну файлами, автентифікації, шифрування та інших операцій.

Однією з особливостей XMPP є можливість використання різних серверів у мережі, що дозволяє користувачам взаємодіяти зі своїми контактами з різних доменів та мереж. XMPP також підтримує групові розмови та відеозв'язок, а також можливість використання різних клієнтів на різних пристроях (наприклад, комп'ютерах, смартфонах тощо).

3.3 Особливості та принцип використання протоколу MQTT

3.3.1 Основні характеристики протоколу MQTT

Комунікаційна стратегія публікації / передплати (pub/sub) MQTT, що спрямована на максимізацію використання пропускної здатності, є заміною традиційної споживчої архітектури, яка безпосередньо взаємодіє з кінцевою точкою. Однак у парадигмі pub/sub клієнт, який передає новини (видавець), відокремлений від клієнтів, які отримують інформацію (або передплатників). Оскільки ані автори, ані клієнти не спілкуються один з одним одразу, їхньою взаємодією в них керують треті сторони, які називаються брокерами (рис. 3.11).

MQTT забезпечує можливість формування ієрархічної структури каналів зв'язку (топіків), подібної до дерева з гілками та листям. Кожного разу, коли видавець передає нові дані, повідомлення супроводжується параметрами контролю доставки (QoS), що визначають рівень надійності його передавання.

Клієнти, підписані на топіки вищого рівня ієрархії, можуть отримувати всі повідомлення відповідної гілки, тоді як клієнти нижчого рівня отримують лише ті повідомлення, що стосуються конкретних підлеглих топіків, розташованих у нижній частині структури. Це полегшує обмін інформацією розміром від двох байт до 256 Мб.

Основні риси протоколу MQTT:

– обмін повідомленнями відбувається за принципом видавець-передплатник (Pub-Sub);

– розмір заголовка повідомлення становить 2 байти, а корисне навантаження може змінюватись від 1 байта до 256 Мб;

– у протоколі закладено можливість вибору одного із трьох рівнів обслуговування.

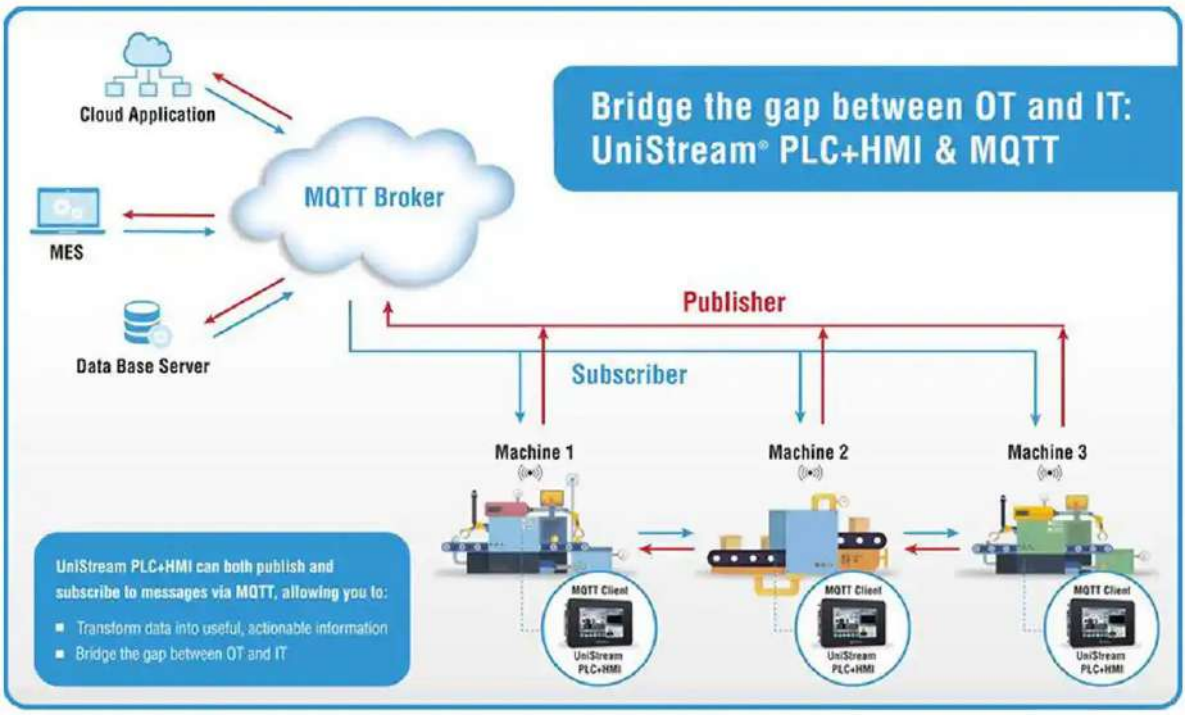


Рисунок 3.11 – Архітектура «видавець-передплатник» у протоколі MQTT

Відмінною особливістю принципу «видавець-передплатник» від клієнт-серверного підходу є те, що клієнти, які надсилають повідомлення (видавці, Publisher), та клієнти, які приймають повідомлення (підписники, Subscriber), як правило, розподілені. Розподіл може бути організований у трьох площинах:

– простір – видавець і передплатник нічого не повинні знати один про одного;

– час – видавець і передплатник не повинні бути увімкнені в один і той самий час;

– синхронізація – операції на обох сторонах не повинні припинятися протягом публікації чи отримання інформації.

Видавець та передплатник не передають одне одному повідомлення безпосередньо, не встановлюють прямий контакт, можуть не знати про існування одне одного. Координує та керує передачею повідомлень від видавця до передплатника та від передплатника до видавця брокер (Broker).

Паралельне виконання операцій на брокері є другою важливою особливістю принципу взаємодії «видавець-передплатник».

MQTT-клієнт – це, зазвичай, пристрій, оснащений мікроконтролером, який підтримує стек TCP/IP. Клієнтські бібліотеки MQTT доступні для багатьох мов програмування, наприклад Android, Arduino, C, C++, C#, Go, iOS, Java, JavaScript, NET.

Брокер є основним елементом системи «видавець-передплатник». Він відповідає за прийом усіх повідомлень, їх фільтрацію, ухвалення рішення про те, кому цікаві ці повідомлення, і, зрештою, за пересилання повідомлень усім клієнтам-передплатникам.

Серед серверних реалізацій брокера можна назвати:

– IBM WebSphere MQ;

– відкрите програмне забезпечення Eclipse Mosquitto;

– рішення, що базується на хмарному сервісі Eurotech Everywhere Device Cloud;

– відкритий високопродуктивний та масштабований MQTT-брокер EMQX (раніше відомий як emqttd), який підтримує кластерну архітектуру та забезпечує обслуговування мільйонів одночасних з’єднань із можливістю горизонтального масштабування;

– брокер HiveMQ, орієнтований на корпоративне застосування, що забезпечує підвищений рівень безпеки, надійності та масштабованості.

Спрощений процес обміну інформацією можна описати так (рис. 3.12):

– видавець передає повідомлення з даними (наприклад, інформація з датчиків температури) на брокер, вказуючи при цьому тему (Topic), до якої ці дані належать (наприклад, «Temp»);

– брокер аналізує, які з передплатників мають передплату на певні теми, в даному випадку – на тему «Temp»;

– передплатникам, які підписані на тему «Temp», брокером буде надіслано повідомлення з інформацією від датчиків температури.

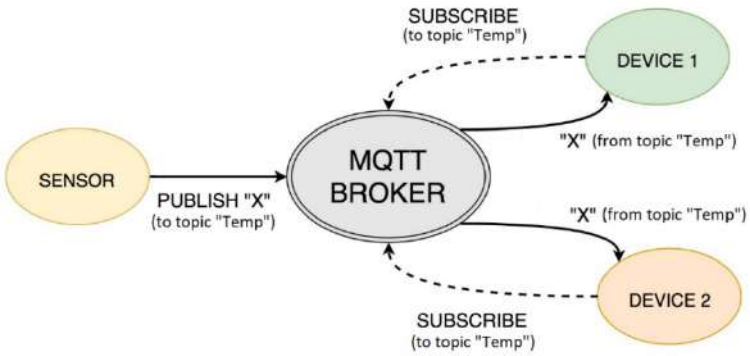


Рисунок 3.12 – Процес обміну інформацією за протоколом MQTT

Таким чином, безліч передплатників може бути підписано на різноманітні теми та, залежно від цих передплат, отримувати необхідну їм інформацію, не спілкуючись безпосередньо з видавцем.

3.3.2 Формат повідомлень за протоколом MQTT

Центральним елементом комунікації є брокер (Broker), що виконує роль сервера, який відповідає за обробку та надсилання всіх повідомлень.

Кожен клієнт, який хоче надіслати повідомлення через сервер, в термінології протоколу MQTT, є видавцем (Publisher).

Брокер фільтрує вхідні повідомлення та перенаправляє клієнтам, які приймають повідомлення. Клієнти, зареєстровані в брокері та підписані на відповідні топіки, називаються передплатники (Subscriber).

Безпосередньо видавець та передплатник повідомлення не передають і між собою ніяк не пов’язані. Для взаємодії з брокером пристрої використовують стандартизований набір повідомлень:

– Connect – встановити з’єднання з брокером;

– Disconnect – розірвати з’єднання з брокером;

– Publish – опублікувати дані в топік на брокері;

– Subscribe – підписатися на топік на брокері;

– Unsubscribe – відписатися від топіка.

Сервер MQTT відповідає за автентифікацію та авторизацію клієнтів. Після успішної авторизації та автентифікації клієнти зможуть стати видавцями та передплатниками.

Спрощено зв’язок між усіма учасниками можна подати наступним чином. Видавець відправляє дані MQTT брокеру, вказуючи при цьому в повідомленні

певну тему – топік, до якої ці дані відносяться. Далі брокер аналізує отримане повідомлення та пересилає його тим клієнтам, які підписані на цю тему. Таким чином, на стороні брокера відбувається фільтрація, що дозволяє передплатникам отримувати лише інформацію, у якій вони зацікавлені. При цьому клієнт може бути підписаний одразу на кілька топіків.

Будь-який IoT-пристрій може виконувати функції як відправника, так і одержувача, тобто кожен передплатник може також виступати у ролі видавця.

Протокол MQTT дозволяє пристроям спілкуватися та взаємодіяти з різними типами серверів. При плануванні та розробці системи необхідно заздалегідь уточнити, які можливості пропонують доступні на ринку сервери та вибрати той, який найкраще задовольняє вимогам.

У протоколі MQTT існує 15 типів повідомлень, які можуть складатися з кількох частин (рис. 3.13):

– фіксований заголовок;

– змінний заголовок;

– корисне навантаження (дані, що передаються).

Рисунок 3.13 – Загальна структура повідомлення MQTT

Фіксований заголовок є у всіх типах повідомлень. В цьому заголовку містяться такі поля (рис 3.14):

– Message Type (тип повідомлення) – наприклад: CONNECT, SUBSCRIBE, PUBLISH тощо;

– прапори, специфічні для кожного пакета MQTT – ці 4 біти використовуються для допоміжних прапорів, наявність і статус яких залежить від типу повідомлення;

– Remaining Length (довжина, що залишається) – поточна довжина повідомлення (заголовок змінної + дані), розмір від 1 до 4 байт.

Bit	7	6	5	4	3	2	1	0
Byte 1	Message Type				Flags specific to each MQTT packet			
Byte 2	Remaining Length							

Рисунок 3.14 – Поля фіксованого заголовка

Всього в протоколі MQTT існує 15 типів повідомлень (табл. 3.1).

Таблиця 3.1 – Типи повідомлень протоколу MQTT

Тип повідомлення	Значення	Напрям передачі	Опис
Зарезервовано	0000 (0)	Заборонено	Зарезервовано
CONNECT	0001 (1)	Клієнт → Сервер	Запит клієнта на підключення до сервера
CONNACK	0010 (2)	Клієнт ← Сервер	Підтвердження підключення
PUBLISH	0011 (3)	Клієнт ← Сервер Клієнт → Сервер	Опублікувати повідомлення
PUBACK	0100 (4)	Клієнт ← Сервер Клієнт → Сервер	Опублікувати підтвердження
PUBREC	0101 (5)	Клієнт ← Сервер Клієнт → Сервер	Публікацію отримано
PUBREL	0110 (6)	Клієнт ← Сервер Клієнт → Сервер	Публікацію випущено
PUBCOMP	0111 (7)	Клієнт ← Сервер Клієнт → Сервер	Публікацію завершено
SUBSCRIBE	1000 (8)	Клієнт → Сервер	Запит клієнта на підписку
SUBACK	1001 (9)	Клієнт ← Сервер	Підписка підтверджена
UNSUBSCRIBE	1010 (10)	Клієнт → Сервер	Запит на скасування підписки
UNSUBACK	1011 (11)	Клієнт ← Сервер	Підтвердження скасування підписки
PINGREQ	1100 (12)	Клієнт → Сервер	Запит PING
PINGRESP	1101 (13)	Клієнт ← Сервер	Відповідь PING
DISCONNECT	1110 (14)	Клієнт → Сервер	Відключення клієнту
Зарезервовано	1111 (15)	Заборонено	Зарезервовано

Чотири старші біти першого байту використовуються як специфічні прапори (рис. 3.15):

– DUP – встановлюється, коли клієнт або брокер MQTT затвердить повторне відправлення пакета (використовується в PUBLISH, SUBSCRIBE, UNSUBSCRIBE, PUBREL). Якщо прапорець встановлено, заголовок змінної має містити ідентифікатор повідомлення;

– QoS – якість обслуговування (0, 1, 2);

– RETAIN – в процесі публікації даних із встановленим прапором retain, брокер збереже його. З наступною передплатою на цей топік брокер негайно відправить повідомлення з цим прапором. Використовується лише у повідомленнях із типом PUBLISH.

Bit	7	6	5	4	3	2	1	0
Byte 1	Message type				DUP	QoS	QoS	Retain
Byte 2	Remaining Length							

Рисунок 3.15 – Специфічні прапори повідомлення

Змінний заголовок є у деяких типах заголовків і містить такі дані:

– Packet identifier (ідентифікатор пакета) – є у більшості типах повідомлень;

– Protocol name (назва протоколу) – відбувається лише у типі повідомлення CONNECT;

– Protocol version (версія протоколу) – відбувається лише у типі повідомлення CONNECT;

– Connect flags (прапори підключення) – прапори, які визначають поведінку клієнта під час підключення.

На рис. 3.16 подана структура змінного заголовка.

Bit	7	6	5	4	3	2	1	0
Byte 8	User name	Password	Will Retain	Will QoS		Will Flag	Clean Session	Reserved

Рисунок 3.16 – Структура змінного заголовка

Байт, що становить заголовок має такі поля:

– User name – якщо прапор встановлений, тоді ім’я користувача має бути в «корисному навантаженні»;

– Password – якщо прапор встановлений, то пароль має бути вказаний в «корисному навантаженні»;

– Will Retain – якщо прапор встановлений, брокер зберігає повідомлення;

– Will QoS – визначає якість послуги повідомлення;

– Will Flag – якщо встановлений прапор, клієнт відключиться від сервера без надсилання команди DISCONNECT; брокер сповіщає всіх підключених клієнтів про цю подію за допомогою Will Message;

– Clean Session – якщо прапор не встановлений, брокер зберігає сеанс, а також усі підписки клієнта, а при наступному підключенні із замовником відправляє всі повідомлення з QoS1 та QoS2, отримані через брокера, коли клієнт був вимкнений. Якщо прапорець встановлений, під час спроби встановлення наступного з’єднання клієнт повинен підписатися на всі теми.

Розмір даних або «корисного навантаження», можна обчислити, віднявши довжину змінного заголовка від решти довжини. Зміст і формат даних, які надсилаються за допомогою повідомлень MQTT, генеруються у програмі, що використовує цей протокол.

QoS є технологією надання різним класам трафіку різних пріоритетів в обслуговуванні. Специфікація протоколу MQTT визначає три рівні якості обслуговування, що визначається спеціальним прапорцем QoS. Як зазначалося вище, основною відмінністю протоколу MQTT є можливість використання різних рівнів обслуговування, які задаються значенням даного прапора. Це робить MQTT більш гнучким, на відміну від протоколу CoAP (Constrained Application Protocol), повідомлення якого можуть підтверджуватись або оброблятися без підтвердження.

QoS 0: (At most once) – не більше одного. Видавець публікує повідомлення (PUBLISH) на брокері, а брокер публікує його для передплатника. Проте видавець не вимагає, щоб це повідомлення було гарантовано передано передплатнику. Іншими словами, передплатник може і не одержати це повідомлення, але це не відслідковується видавцем (рис. 3.17).

Описаний сценарій використовується для випадків, коли втрати даних не критичні. Наприклад, під час постійного моніторингу температури, коли втрата одиничного вимірювання не відіграє істотну роль загальної картини.

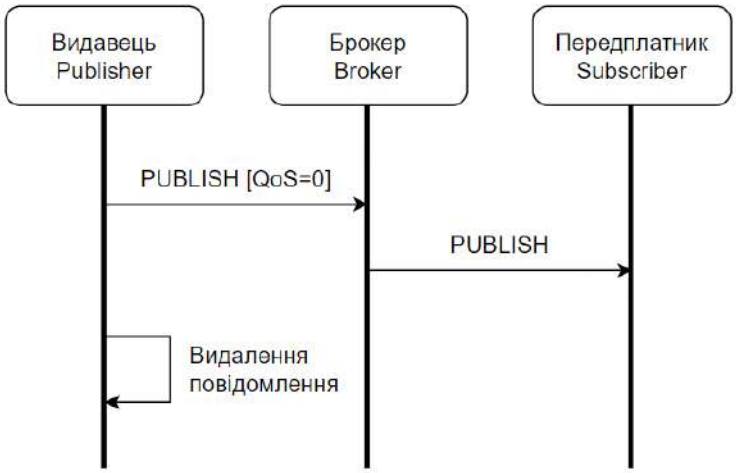


Рисунок 3.17 – Сценарій QoS 0: (At most once) – не більше одного

QoS 1: At least once – хоча б один. Видавець публікує повідомлення на брокері (PUBLISH). Брокер зберігає це повідомлення та публікує його для передплатника. Тільки після того, як повідомлення буде опубліковано для передплатника, брокер надсилає підтвердження публікації видавцю (PUBACK). Сценарій такої взаємодії подано на рис. 3.18.

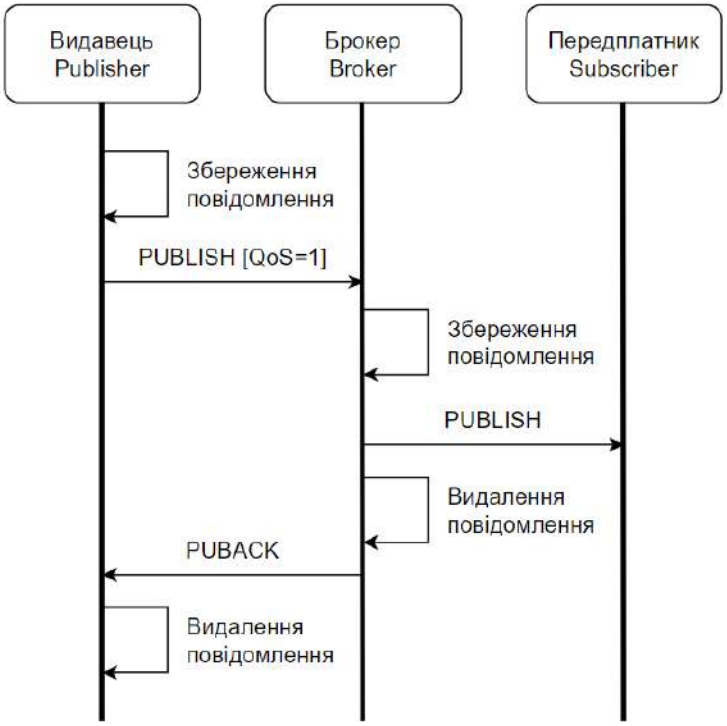


Рисунок 3.18 – Сценарій QoS 1: At least once – хоча б один

Доки видавець не отримає підтвердження публікації передплатнику, дана публікація надсилатиметься брокеру і далі передплатнику. Таким чином, передплатник повинен отримати це повідомлення як мінімум один раз.

QoS 2: Exactly one – гарантовано один. Даний рівень QoS забезпечує найвищу гарантію доставки повідомлень за рахунок використання додаткових процедур підтвердження та завершення публікації (PUBREC, PUBREL, PUBCOMP). Сценарій представлено на рис. 3.19 і застосовується для ситуацій, коли потрібно виключити будь-які втрати та дублювання даних від датчиків. Наприклад, коли від отриманого повідомлення спрацьовує сигналізація, виклик екстрених служб.

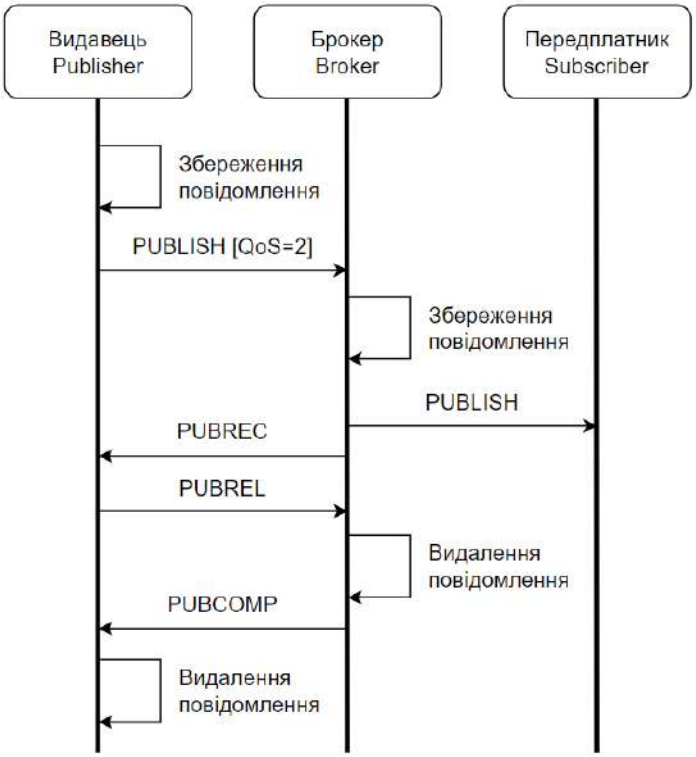


Рисунок 3.19 – Сценарій QoS 2: Exactly one – гарантовано один

Спеціальний прапор RETAIN служить для індикації збереження останнього прийнятого брокером повідомлення (рис. 3.20). Тобто прапор RETAIN = 1 у повідомленні PUBLISH від видавця повідомляє брокеру про те, що повідомлення на цю тему потрібно зберегти і, коли новий передплатник приєднається до теми, надіслати йому це повідомлення.

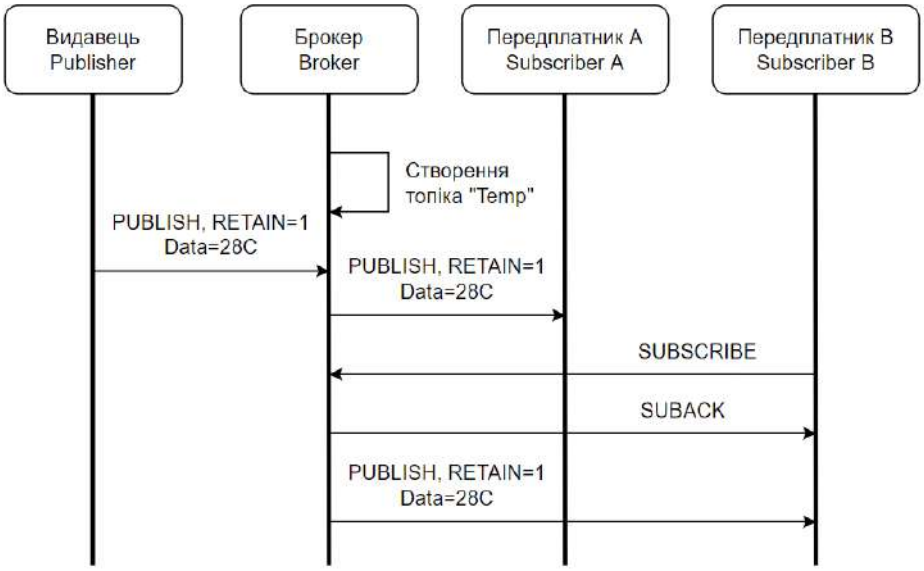


Рисунок 3.20 – Сценарій для прапорця RETAIN

Процес встановлення з’єднання подано на рис. 3.21.

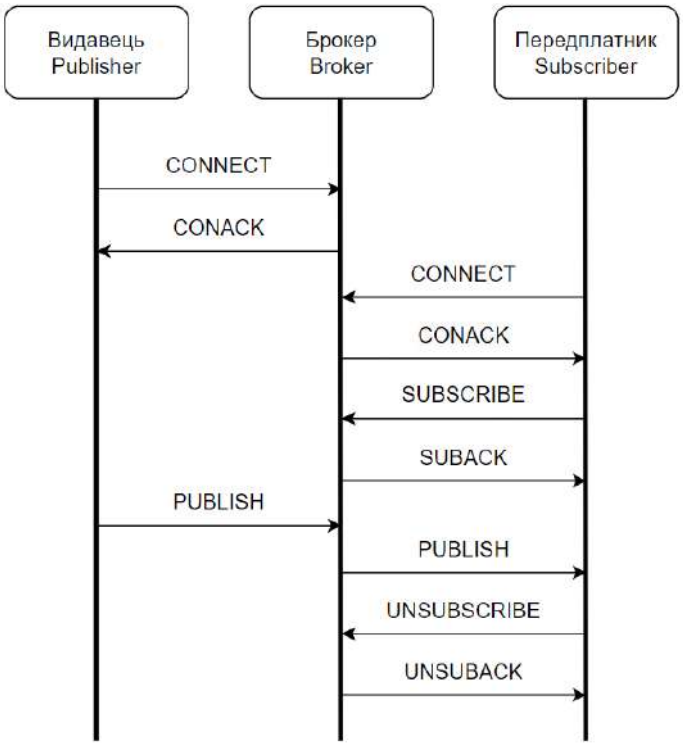


Рисунок 3.21 – Процес встановлення з’єднання

Встановлення з’єднання починається з передачі від клієнта брокеру повідомлення CONNECT, в якому зазначаються:

– ClientId – унікальний ідентифікатор кожного клієнта, що підключається до брокера;

– CleanSession – прапор видалення збережених повідомлень із попередніх сесій для даного клієнта;

– Username/Password – ім’я користувача та пароль для ідентифікації та авторизації клієнта;

– KeepAlive – тимчасовий інтервал, що регулює передачу ping-запитів та ping-відповідей для контролю відключення однієї зі сторін.

Брокер у відповідь надсилає клієнту повідомлення CONACK, що складається з:

– Session Present Flag – вказує чи існують для даного клієнта сесії, що діють, від попередніх підключень;

– Connect Acknowledge Flag – повідомляє клієнта про успішне підключення або про будь-які помилки.

Після того, як клієнта MQTT було підключено до брокера, він може публікувати повідомлення. Публікація відбувається шляхом відправки брокеру від клієнта повідомлення PUBLISH, де зазначаються:

– Topic Name – назва теми, до якої належить це повідомлення. Це поле є обов’язковим, тому що MQTT-брокер приймає рішення про пересилання того чи іншого повідомлення клієнту, виходячи з тем, на які клієнт підписаний;

– спеціальні прапори – QoS, DUP та RETAIN;

– корисне навантаження, де передаються самі дані.

Таким чином, після отримання повідомлення PUBLISH брокер відправляє підтвердження прийому публікації (якщо це задано QoS) і надсилає отримане повідомлення всім клієнтам, які підписані на цю тему.

Щоб отримувати повідомлення з необхідними даними, клієнт MQTT повинен спочатку підписатися на їх отримання за допомогою повідомлення SUBSCRIBE. Це повідомлення складається з двох частин:

– Packet Identifier – необхідно для QoS 1 та QoS 2;

– List of Subscriptions – назви тем, які клієнт хоче передплатити, та необхідне значення QoS.

Слід зазначити, що у протоколі MQTT прийнята ієрархічна структура побудови тем, тому для зручності застосовуються wildcard-символи, завдяки яким передплатник може передплатити всі підтеми цієї теми (символ #) чи теми певного рівня (символ +).

У відповідь на повідомлення SUBSCRIBE брокер відправляє клієнту підтвердження SUBACK, в якому повідомляє результат підписки (успішна

чи ні). Також клієнт може відписатися від теми, яка більше не представляє для нього інтересу, відправивши брокеру повідомлення UNSUBSCRIBE, у якому буде зазначена дана тема. Брокер підтверджує відмову від інформації на цю тему повідомленням UNSUBACK.

IIoT містить взаємозалежні датчики, прилади та інші пристрої, об'єднані в мережу з комп'ютерами за допомогою промислових програм. Таке об'єднання дозволяє збирати та аналізувати дані, обмінюватися ними, що потенційно сприяє підвищенню продуктивності праці та ефективності виробництва, а також дає інші економічні переваги. IIoT – це еволюція розподіленої системи управління (DCS), яка дозволяє підвищити рівень автоматизації за рахунок використання хмарних обчислень для уточнення та оптимізації управління процесом.

IIoT використовує технології кібербезпеки, хмарні обчислення, граничні обчислення, мобільні технології, міжмашинну взаємодію, 3D-друк, передову робототехніку, аналіз великих обсягів даних, Інтернет речей, технологію RFID та когнітивні обчислення. П'ять найважливіших з них описані нижче.

1. Кіберфізичні системи (англ. Cyber-physical systems) – це інформаційно-технологічна концепція, що передбачає інтеграцію обчислювальних ресурсів у фізичні процеси. Такі системи складаються з пов'язаних фізичних та обчислювальних компонентів і функціонують на стику реального та віртуального світів, забезпечують їхню взаємодію та ефективне управління різними технологіями – розумними містами, автоматизованими системами управління виробництвом, енергетикою, Big Data, IoT, IIoT, штучним інтелектом та іншими.

2. Хмарні обчислення дозволяють надавати IT-послуги, ресурси при цьому завантажуються та вилучаються з Інтернету без прямого підключення до сервера. Файли можуть зберігатися у хмарних системах зберігання, а не на локальних пристроях.

3. Крайові обчислення – це парадигма розподілених обчислень, що наближає комп'ютерне сховище даних до місця, де вони необхідні. На відміну від хмарних обчислень, крайові (англ. Edge Computing), або граничні, периферійні обчислення належать до децентралізованої обробки даних на межі мережі. У середовищі IIoT більш затребуваною є архітектура «edge + cloud», ніж підхід, що ґрунтується виключно на централізованих хмарних обчисленнях, оскільки вона сприяє підвищенню продуктивності праці, збільшенню обсягів товарів і послуг в індустріальному секторі та покращенню їхньої якості.

4. Аналітика великих даних – це процес вивчення великих та різноманітних наборів даних або великих обсягів даних.

5. Штучний інтелект (ШІ) – це область комп'ютерних наук, в якій створюються інтелектуальні машини, що працюють і реагують подібно до людей. Машинне навчання є основною частиною ШІ, воно дозволяє програмному забезпеченню стати точнішим у прогнозуванні результатів без явного програмування.

Системи IIoT часто сприймаються як багаторівнева модульна архітектура цифрових технологій. Рівень пристрою визначається фізичним компонентом: CPS, датчиком чи машиною. Мережевий рівень включає фізичні мережеві шини, хмарні обчислення та протоколи зв'язку, що агрегують та транспортують дані на сервісний рівень. Сервісний рівень складається з додатків, які маніпулюють даними та об'єднують їх у інформацію, що відображається на панелі приладів драйвера.

Платформи IoT допомагають підтримувати взаємодію між речами та створювати складніші структури, такі як розподілені обчислення та розподілені додатки. Розвиток цифрових технологій у цьому напрямку може спричинити створення середовища розробки програмного забезпечення, призначеного спеціально для IoT. Компанії розробляють технологічні платформи, щоб забезпечити цей тип функціональності для IoT. Розробляються нові платформи, які ширше використовують штучний інтелект.

IIoT успішно застосовується як у соціальній сфері (транспорт, охорона здоров'я, житлово-комунальне господарство), так і в різних галузях промисловості. За допомогою IIoT створюються нові бізнес-моделі, підвищується продуктивність праці, перетворюється робоча сила.

3.4 Розгортання брокера MQTT Eclipse Mosquitto

3.4.1 Основні відомості про Eclipse Mosquitto

Eclipse Mosquitto є популярним брокером повідомлень MQTT, який забезпечує зв'язок між сенсорами та іншими IoT-пристроями. Він був розроблений за допомогою мови програмування C і може працювати як на Windows, так і на Linux.

Eclipse Mosquitto простий у налаштуванні та використанні. Він має низькі вимоги до ресурсів, що дозволяє йому працювати на вбудованих пристроях та

малих комп'ютерах. Eclipse Mosquitto є повноцінним брокером MQTT. Він підтримує версії 5.0, 3.1 та 3.1.1 протоколу MQTT.

За технологією pub/sub Eclipse Mosquitto підтримує можливість підписки, які дозволяють отримувати повідомлення тільки на певні теми.

Eclipse Mosquitto підтримує три рівні якості обслуговування (QoS):

– QoS 0 (кожен раз надсилається тільки один раз);

– QoS 1 (надсилається щонайменше один раз);

– QoS 2 (надсилається рівно два рази).

Даний брокер підтримує автентифікацію та шифрування. Він може використовувати базу даних автентифікації для перевірки користувачів та паролів. Крім того, він підтримує SSL / TLS для шифрування з'єднань. Також, він підтримує мости між різними брокерами MQTT. Це дозволяє повідомленням з одного брокера MQTT бути переданими на інший брокер MQTT.

3.4.2 Порядок встановлення брокера Eclipse Mosquitto на ОС Linux

Розглянемо порядок дій для встановлення брокера Eclipse Mosquitto на ОС Linux. Першим кроком, необхідно виконати команду:

```
sudo apt update
```

Дана команда виконує оновлення списку доступних пакетів для встановлення з віддалених репозиторіїв. Це дозволяє системі знайти та завантажити оновлення для встановлених пакетів та інші корисні програми. Оновлення списку доступних пакетів дозволяє зберегти систему оновленою та забезпечити стабільну роботу пакетного менеджера під час встановлення нових програм. Перед встановленням будь-якої програми на ОС Linux, зазвичай рекомендується виконати команду sudo apt update для забезпечення актуальності списку пакетів.

Наступним кроком, встановлюємо Eclipse Mosquitto за допомогою команди (рис. 3.22):

```
sudo apt install mosquitto
```

В результаті виконання команди на ОС буде встановлено необхідне програмне забезпечення для роботи з брокером Eclipse Mosquitto.

```
serg@serg-virtual-machine:~$ sudo apt install mosquitto
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  libcjson1 libdlt2 libev4 libmosquitto1 libwebsockets16
The following NEW packages will be installed:
  libcjson1 libdlt2 libev4 libmosquitto1 libwebsockets16 mosquitto
0 upgraded, 6 newly installed, 0 to remove and 21 not upgraded.
Need to get 575 kB of archives.
After this operation, 1 664 kB of additional disk space will be used.
Do you want to continue? [Y/n]
```

Рисунок 3.22 – Встановлення брокеру Mosquitto

В версії ОС, наприклад Ubuntu 22.04 LTS крім базової команди встановлення брокера, необхідно додатково виконати перевірку та встановлення додаткових бібліотек. Таким чином, якщо попередньо не було встановлено менеджер пакетів Python pip, його необхідно додати до системи за допомогою команди:

```
sudo apt install python3-pip
```

Далі встановлюємо бібліотеку Paho MQTT для Python за допомогою pip (рис. 3.23):

```
pip3 install paho-mqtt
```

```
serg@serg-virtual-machine:~$ pip3 install paho-mqtt
Defaulting to user installation because normal site-packages is not writeable
Collecting paho-mqtt
  Downloading paho-mqtt-1.6.1.tar.gz (99 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 99.4/99.4 KB 1.2 MB/s eta 0:00:00
  Preparing metadata (setup.py) ... done
Building wheels for collected packages: paho-mqtt
  Building wheel for paho-mqtt (setup.py) ... done
  Created wheel for paho-mqtt: filename=paho_mqtt-1.6.1-py3-none-any.whl size=62
133 sha256=a707f11d80067f4ac79bbbd8b3d0515cdae09d63dbf896ecf618e2d94fca14c6
  Stored in directory: /home/serg/.cache/pip/wheels/8b/bb/0c/79444d1dee20324d442
856979b5b519b48828b0bd3d05df84a
Successfully built paho-mqtt
Installing collected packages: paho-mqtt
Successfully installed paho-mqtt-1.6.1
```

Рисунок 3.23 – Встановлення бібліотеки Paho MQTT для Python

Бібліотека Paho MQTT для Python необхідна для підтримки MQTT-протоколу в програмах на мові програмування Python. Вона містить функції для підключення до брокера MQTT, публікації повідомлень та підписки на топіки. Встановлення цієї бібліотеки дозволить створювати програми, які взаємодіють з іншими пристроями за допомогою MQTT-протоколу.

Щоб перевірити, чи встановлено Eclipse Mosquitto правильно, необхідно запустити службу Eclipse Mosquitto та перевірити її стан за допомогою таких команд:

```
sudo systemctl start mosquitto
sudo systemctl status mosquitto
```

На рис. 3.24 подано результат виконання цих команд.

```
serg@serg-virtual-machine:~$ sudo systemctl start mosquitto
serg@serg-virtual-machine:~$ sudo systemctl status mosquitto
● mosquitto.service - Mosquitto MQTT Broker
     Loaded: loaded (/lib/systemd/system/mosquitto.service; enabled; vendor pre>
     Active: active (running) since Sun 2023-02-26 12:19:37 EET; 12min ago
       Docs: man:mosquitto.conf(5)
             man:mosquitto(8)
    Process: 3005 ExecStartPre=/bin/mkdir -m 740 -p /var/log/mosquitto (code=ex>
    Process: 3006 ExecStartPre=/bin/chown mosquitto /var/log/mosquitto (code=ex>
    Process: 3007 ExecStartPre=/bin/mkdir -m 740 -p /run/mosquitto (code=exited>
    Process: 3008 ExecStartPre=/bin/chown mosquitto /run/mosquitto (code=exited>
   Main PID: 3009 (mosquitto)
      Tasks: 1 (limit: 4520)
     Memory: 1.4M
        CPU: 454ms
     CGroup: /system.slice/mosquitto.service
             └─3009 /usr/sbin/mosquitto -c /etc/mosquitto/mosquitto.conf

Feb 26 12:19:37 serg-virtual-machine systemd[1]: Starting Mosquitto MQTT Broker>
Feb 26 12:19:37 serg-virtual-machine systemd[1]: Started Mosquitto MQTT Broker.
```

Рисунок 3.24 – Результат запуску Eclipse Mosquitto та перевірки його стану

Якщо служба Eclipse Mosquitto запускається успішно, перевіримо, чи можна підключитись до неї з використанням інструментів, таких як mosquitto_sub та mosquitto_pub. Якщо ці команди планується запускати на тій самій операційній системі, що й сам сервер, то необхідно додати до неї програму-клієнт. Також це необхідно зробити й на іншому ПК, де планується виконувати задачі підключення до брокера за допомогою вищевказаних команд.

Mosquitto-clients – це пакет програм для командного рядка, які дозволяють взаємодіяти з брокером Eclipse Mosquitto через протокол MQTT. Встановлення mosquitto-clients дозволяє виконувати наступні операції:

– перевірка підключення до брокера Eclipse Mosquitto;

– підписка на теми для отримання повідомлень;

– публікація повідомлень на теми;

– тестування і налагодження роботи з брокером Eclipse Mosquitto.

Встановлення mosquitto-clients є корисним для тих, хто працює з MQTT-брокером Eclipse Mosquitto та бажає взаємодіяти з ним через командний рядок.

Виконаємо встановлення клієнта за допомогою наступної команди:

```
sudo apt install mosquitto-clients
```

Перевірка роботи брокера виконується в декілька кроків (рис. 3.25):

– запускаємо два окремих термінали;

– в першому терміналі вводимо та виконуємо наступну команду (рис. 3.25, 1)

```
mosquitto_sub -h localhost -t test
```

для передплати на тему «test», після чого термінал переходить в режим очікування;

– в другому терміналі виконаємо публікацію даних в темі «test» (рис. 3.25, 2):

```
mosquitto_pub -h localhost -t test -m "hello world"
```

– після публікації даних в першому терміналі повинні з’явитись такі самі дані, які були відправлені (рис. 3.25, 3).

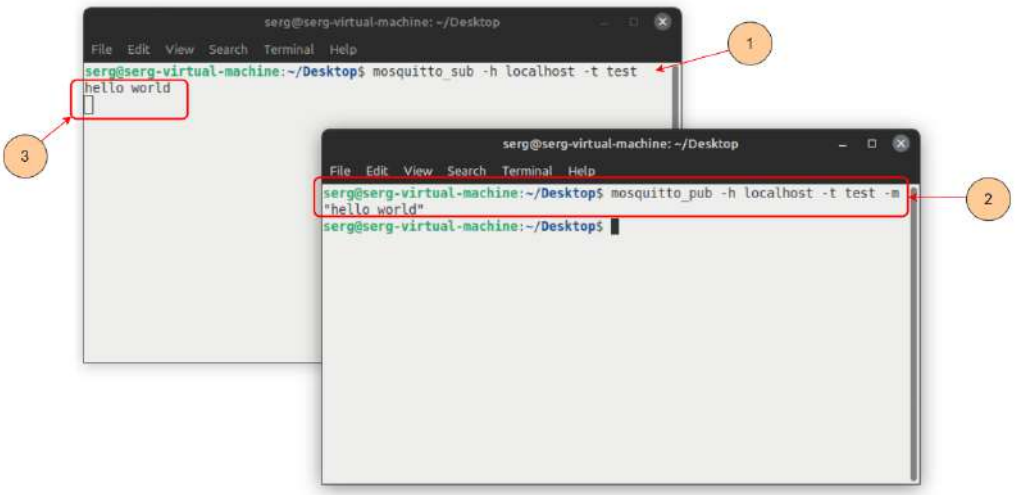


Рисунок 3.25 – Перевірка роботи брокера Eclipse Mosquitto

В більшості випадках практичного застосування брокера Eclipse Mosquitto використовується автентифікація піл час виконання передплати на теми та публікації самих тем.

Для увімкнення підтримки автентифікації в Eclipse Mosquitto потрібно налаштувати файл конфігурації mosquitto.conf. Відкриємо цей файл командою:

```
sudo nano /etc/mosquitto/mosquitto.conf
```

Виконаємо додавання наступних рядків в кінець файлу:

```
# Enable authentication
allow_anonymous false
password_file /etc/mosquitto/passwd
```

На рис. 3.26 подано вміст файлу конфігурації після додавання необхідних налаштувань. Перший рядок «allow_anonymous false» вимикає можливість анонімного підключення до брокера. Другий рядок «password_file …» вказує на шлях до файлу з паролями користувачів. Після додавання нових рядків необхідно зберегти зміни та закрити файл конфігурації.

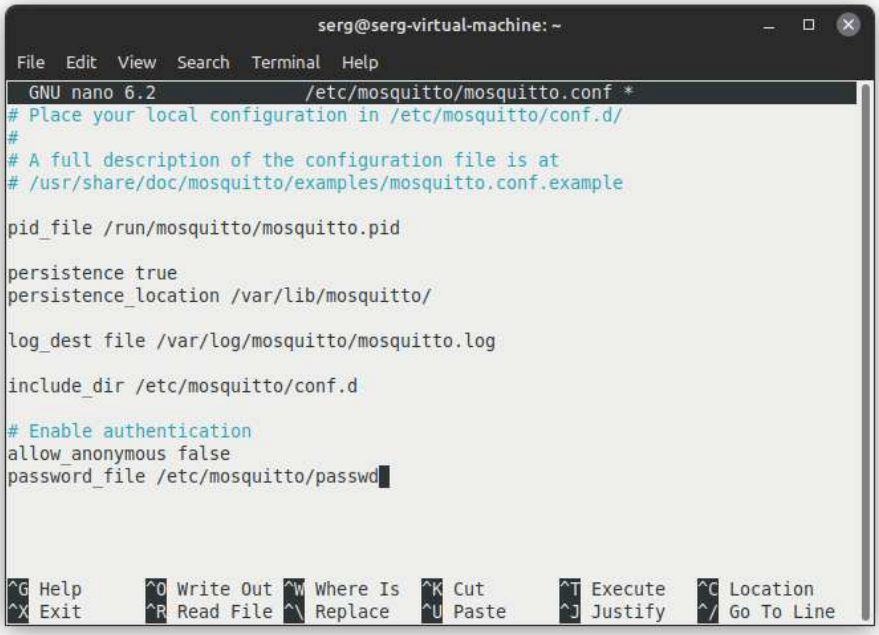


Рисунок 3.26 – Вміст файлу конфігурації

Далі необхідно створити файл з паролями користувачів командою:

```
sudo mosquitto_passwd -c /etc/mosquitto/passwd <username>
```

Замість <username> необхідно вказати ім’я користувача, якому потрібно створити пароль. Після цього система попросить ввести пароль для користувача (рис. 3.27). Цей крок необхідно повторити для кожного користувача, якому потрібно створити пароль.

Після створення паролів необхідно перезавантажити службу Eclipse Mosquitto, щоб зміни набрали чинності:

```
sudo systemctl restart mosquitto
```

```
serg@serg-virtual-machine:~$ sudo mosquitto_passwd -c /etc/mosquitto/passwd testMQTT
Password:
Reenter password:
```

Рисунок 3.27 – Створення пароля для користувача testMQTT

Тепер при спробі підключитись до Eclipse Mosquitto буде необхідно ввести ім’я користувача та пароль. Для підключення до Eclipse Mosquitto з ім’ям користувача та паролем, необхідно використовувати параметри -u та -P під час використання командного рядка mosquitto_sub або mosquitto_pub.

Наприклад, якщо ми хочемо підписатись на повідомлення топіка «example» з ім’ям користувача «testMQTT» та відповідним паролем «testMQTT», ми можемо виконати наступну команду:

```
mosquitto_sub -h <hostname> -t example -u testMQTT -P testMQTT
```

Аналогічно, якщо ми хочемо надіслати повідомлення до топіка «example» з ім’ям користувача «testMQTT» та паролем «testMQTT», ми можемо виконати наступну команду:

```
mosquitto_pub -h <hostname> -t example -m "Hello, World!" -u testMQTT -P testMQTT
```

Необхідно звернути увагу, що необхідно замінити <hostname> на ім’я або IP-адресу хоста, на якому працює Eclipse Mosquitto. Приклад використання автентифікації подано на рис. 3.28.

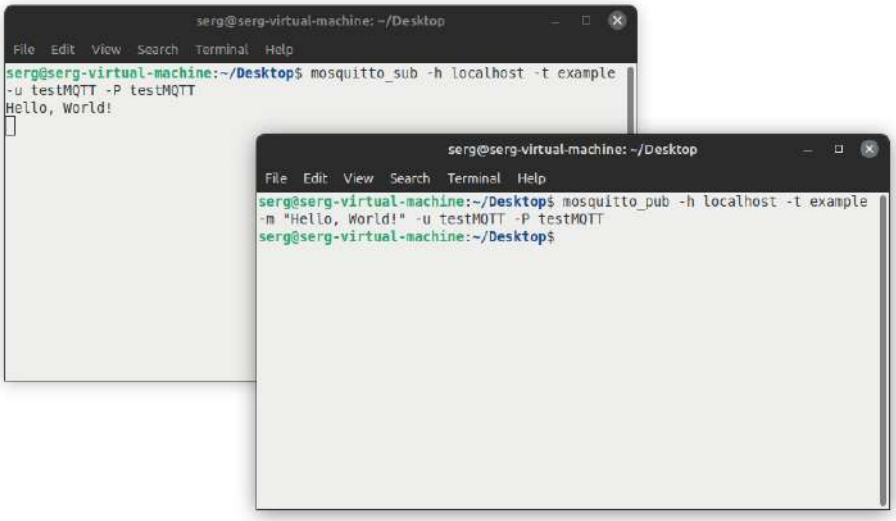


Рисунок 3.28 – Приклад використання автентифікації під час роботи з брокером Eclipse Mosquitto

3.5 Використання графічної платформи Cedalo для управління брокером Eclipse Mosquitto

Cedalo є платформою для розробки та розгортання рішень Інтернету речей, яка надає простий та інтуїтивно зрозумілий інтерфейс, що дозволяє швидко створювати та налаштовувати IIoT-проєкти без спеціалізованих знань.

Cedalo надає широкий спектр інструментів для розробки та інтеграції рішень Інтернету речей, таких як Node-RED, Apache Kafka та MQTT. Ця платформ має вбудовану підтримку захисту від кібератак, а також забезпечує захист персональних даних користувачів.

Cedalo може бути використана для створення як малих, так і великих IIoT-проєктів, які можуть масштабуватися залежно від потреб користувачів. Даний інструмент може бути використаний на різних платформах, включаючи Linux, Windows та MacOS.

Також, Cedalo має активну спільноту користувачів та розробників, які надають допомогу та розробляють нові функції та плагіни для платформи.

Наряду з платною, існує безкоштовна версія платформи управління з відкритим кодом. Її можна використовувати для керування клієнтами, групами та ролями, що доступно у новому плагіні Dynamic Security. Щоб встановити та налаштувати цю версію платформи, потрібна платформа Docker на робочій версії Linux для доступу до графічного інтерфейсу керування.

Встановлення Docker розпочинається з оновлення списку доступних пакетів для встановлення з віддалених репозиторіїв:

```
sudo apt update.
```

Наступною командою є:

```
sudo apt install ca-certificates curl apt-transport-https
```

Ця команда встановлює необхідні залежності для того, щоб здійснити успішний доступ до репозиторію Cedalo, зокрема:

– ca-certificates – пакет з сертифікатами довірених організацій, що використовуються для перевірки підписів цифрових сертифікатів під час з'єднання з захищеними ресурсами через HTTPS;

– curl – утиліта для отримання вмісту з інтернету за допомогою протоколу HTTP або HTTPS;

– apt-transport-https – модуль для підтримки передачі даних через HTTPS у менеджері пакетів apt.

Щоб завантажити пакети Docker на Ubuntu, нам потрібно додати ключ GPG, який використовується розробником для підпису пакетів Docker, інакше система поверне помилку та не зможе використовувати репозиторій:

```
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -
```

Далі, ми можемо встановити докер за допомогою стандартного системного репозиторію Ubuntu Jammy, однак доступна версія не буде останньою. Отже, необхідно додати офіційне сховище Docker вручну за допомогою поданого нижче блоку команд:

```
echo "deb [arch=amd64] https://download.docker.com/linux/ubuntu focal stable" \
| sudo tee /etc/apt/sources.list.d/docker.list > /dev/null
```

Нарешті ми налаштували всі необхідні речі. Тепер необхідно запустити команду системного оновлення, щоб оновити кеш сховища та оновити вже встановлені пакети (рис. 3.29):

```
sudo apt update
```

Після цього скористаємось пакетами APT, щоб отримати всі інструменти Docker, необхідні для створення контейнерів:

```
sudo apt-get install docker-ce docker-ce-cli containerd.io docker-compose
```

```
serg@serg-virtual-machine:~/Cedalo$ sudo apt update
Hit:1 http://security.ubuntu.com/ubuntu jammy-security InRelease
Hit:2 http://archive.ubuntu.com/ubuntu jammy InRelease
Hit:3 https://deb.nodesource.com/node_19.x jammy InRelease
Get:4 https://download.docker.com/linux/ubuntu focal InRelease [57,7 kB]
Hit:5 http://archive.ubuntu.com/ubuntu jammy-updates InRelease
Ign:6 http://packages.linuxmint.com vera InRelease
Hit:7 http://archive.ubuntu.com/ubuntu jammy-backports InRelease
Hit:8 http://packages.linuxmint.com vera Release
Get:10 https://download.docker.com/linux/ubuntu focal/stable amd64 Packages [24,7 kB]
Fetched 82,4 kB in 1s (57,4 kB/s)
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
21 packages can be upgraded. Run 'apt list --upgradable' to see them.
W: https://download.docker.com/linux/ubuntu/dists/focal/InRelease: Key is stored in lega
cy trusted.gpg keyring (/etc/apt/trusted.gpg), see the DEPRECATION section in apt-key(8)
 for details.
```

Рисунок 3.29 – Оновлення кешу репозіторію

На рис. 3.30 подано результат виконання даної команди.

```
serg@serg-virtual-machine:~/Cedalo$ sudo apt-get install docker-ce docker-ce-cli contain
erd.io docker-compose
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  docker-buildx-plugin docker-ce-rootless-extras docker-compose-plugin
  docker-scan-plugin git git-man liberror-perl libslirp0 pigz python3-attr
  python3-docker python3-dockerpty python3-docopt python3-dotenv python3-jsonschema
  python3-pyrsistent python3-texttable python3-websocket slirp4netns
Suggested packages:
  aufs-tools cgroupfs-mount | cgroup-lite git-daemon-run | git-daemon-sysvinit git-doc
  git-email git-gui gitk gitweb git-cvs git-mediawiki git-svn python-attr-doc
  python-jsonschema-doc
Recommended packages:
  docker.io
The following NEW packages will be installed:
  containerd.io docker-buildx-plugin docker-ce docker-ce-cli docker-ce-rootless-extras
  docker-compose docker-compose-plugin docker-scan-plugin git git-man liberror-perl
  libslirp0 pigz python3-attr python3-docker python3-dockerpty python3-docopt
  python3-dotenv python3-jsonschema python3-pyrsistent python3-texttable
  python3-websocket slirp4netns
```

Рисунок 3.30 – Встановлення пакетів, що необхідні для роботи Docker на Ubuntu

Ця команда встановлює пакети, необхідні для роботи Docker на Ubuntu:

– docker-ce: це пакет Docker Community Edition, який містить основні компоненти Docker Engine, необхідні для запуску контейнерів;

– docker-ce-cli: це пакет Docker CLI (Command Line Interface), який містить команди для управління Docker з командного рядка;

– containerd.io: це пакет Containerd, який забезпечує рівень абстракції над Docker Engine і дозволяє управляти процесами контейнерів;

– docker-compose: це пакет Docker Compose, який дозволяє визначати та запускати багатоконтейнерні застосунки з конфігурацією в YAML-форматі.

Після встановлення перевіряємо роботу служби Docker:

```
systemctl status docker
```

Результат перевірки подано на рис. 3.31.

```
serg@serg-virtual-machine:~/Cedalo$ systemctl status docker
● docker.service - Docker Application Container Engine
     Loaded: loaded (/lib/systemd/system/docker.service; enabled; vendor preset: enable>
     Active: active (running) since Sun 2023-02-26 15:43:42 EET; 11s ago
TriggeredBy: ● docker.socket
       Docs: https://docs.docker.com
   Main PID: 7031 (dockerd)
      Tasks: 8
     Memory: 26.4M
        CPU: 377ms
     CGroup: /system.slice/docker.service
             └─7031 /usr/bin/dockerd -H fd:// --containerd=/run/containerd/containerd.s>

Feb 26 15:43:42 serg-virtual-machine dockerd[7031]: time="2023-02-26T15:43:42.076898006>
Feb 26 15:43:42 serg-virtual-machine dockerd[7031]: time="2023-02-26T15:43:42.076976504>
Feb 26 15:43:42 serg-virtual-machine dockerd[7031]: time="2023-02-26T15:43:42.151787167>
Feb 26 15:43:42 serg-virtual-machine dockerd[7031]: time="2023-02-26T15:43:42.393502542>
Feb 26 15:43:42 serg-virtual-machine dockerd[7031]: time="2023-02-26T15:43:42.581362987>
Feb 26 15:43:42 serg-virtual-machine dockerd[7031]: time="2023-02-26T15:43:42.640982409>
Feb 26 15:43:42 serg-virtual-machine dockerd[7031]: time="2023-02-26T15:43:42.641163126>
Feb 26 15:43:42 serg-virtual-machine dockerd[7031]: time="2023-02-26T15:43:42.676962409>
Feb 26 15:43:42 serg-virtual-machine systemd[1]: Started Docker Application Container E>
Feb 26 15:43:42 serg-virtual-machine dockerd[7031]: time="2023-02-26T15:43:42.687211589>
lines 1-22/22 (END)
```

Рисунок 3.31 – Перевірка роботи служби Docker

Якщо служба не запущена, запустити її можна за допомогою команди:

```
sudo systemctl start docker
```

Після встановлення та запуску Docker, можна виконати команду:

```
docker run -it -v ~/cedalo_platform:/cedalo cedalo/installer:2-linux
```

Ця команда запускає Docker-контейнер з ім’ям «cedalo/installer:2-linux» і виконується в інтерактивному режимі з підключенням терміналу (-it). Також вказується опція -v, яка монтує локальну директорію ~/cedalo_platform до директорії /cedalo в контейнері.

Команда використовується для запуску інсталятора Cedalo Platform в Docker-контейнері і монтування директорії, яка містить налаштування та дані для платформи. Команда дозволяє інсталювати та налаштовувати платформу в середовищі, що ізольоване від решти системи, що дозволяє зберегти стабільність роботи та безпеку. Результат роботи команди подано на рис. 3.32.

```
serg@serg-virtual-machine:~/Cedalo$ sudo docker run -it -v ~/cedalo_platform:/cedalo ced
alo/installer:2-linux
Unable to find image 'cedalo/installer:2-linux' locally
2-linux: Pulling from cedalo/installer
a0d0a0d46f8b: Pull complete
4684278ccdc1: Downloading  3.757MB/36.51MB
cb39e3b315fc: Download complete
90bb485869f4: Download complete
a9b6e68f09a6: Download complete
da18d27ebad3: Download complete
9f4029d62cf1: Download complete
c01a6f0e2030: Waiting
92f5561ced99: Waiting
b2a26f7dd1b0: Waiting
2a40628157d4: Waiting
b09ba9722997: Waiting
cc699223b320: Waiting
56b108e4e071: Waiting
```

Рисунок 3.32 – Запуск інсталятора Cedalo Platform в Docker-контейнері

Коли докер завантажить платформу Cedalo, він запитає, які компоненти необхідно встановити. Можна погодитись із попереднім вибором та просто натиснути клавішу Enter (рис. 3.33).

```
alo/installer:2-linux
Unable to find image 'cedalo/installer:2-linux' locally
2-linux: Pulling from cedalo/installer
a0d0a0d46f8b: Pull complete
4684278ccdc1: Pull complete
cb39e3b315fc: Pull complete
90bb485869f4: Pull complete
a9b6e68f09a6: Pull complete
da18d27ebad3: Pull complete
9f4029d62cf1: Pull complete
c01a6f0e2030: Pull complete
92f5561ced99: Pull complete
b2a26f7dd1b0: Pull complete
2a40628157d4: Pull complete
b09ba9722997: Pull complete
cc699223b320: Pull complete
56b108e4e071: Pull complete
Digest: sha256:d37638ce9aec2f3cd2925f864929b5b656cc9ada51282467ededd5f2ab8968ff
Status: Downloaded newer image for cedalo/installer:2-linux
? Select what to install › - Space to select. Return to submit
◉   Management Center for Eclipse Mosquitto
◉   Eclipse Streamsheets
◉   Eclipse Mosquitto 2.0
○   Eclipse Mosquitto 1.6
```

Рисунок 3.33 – Підтвердження вибору компонентів

Якщо виконується тільки встановлення графічного інтерфейсу керування, то не слід вибирати встановлення Eclipse Mosquitto, оскільки його вже

встановлено. В іншому випадку можна залишити налаштування за замовчуванням без змін (рис. 3.34).

```
✓ Select what to install › Management Center for Eclipse Mosquitto, Eclipse Streamsheets
, Eclipse Mosquitto 2.0
Successfully installed the following services: Management Center for Eclipse Mosquitto,
Eclipse Streamsheets, Eclipse Mosquitto 2.0
Navigate to the installation directory (e.g. C:cedalo_platform or ~/cedalo_platform) and
 run the start.bat or start.sh script
```

Рисунок3.34 – Результат вибору та інсталяції пакетів

Після завершення процесу встановлення ми маємо каталог cedalo_platform у нашому домашньому кталозі /home з усіма необхідними файлами, які потрібні для запуску цього інтерфейсу керування GUI Mosquitto.

Переходимо в цей каталог:

```
cd ~/cedalo_platform
```

та запускаємо платформу Cedalo командою:

```
sudo ./start.sh
```

На рис. 3.35 подано приклад запуску платформи Cedalo.

```
serg@serg-virtual-machine:~/cedalo_platform$ sudo ./start.sh
streamsheets-data
Creating network "cedalo-platform" with driver "bridge"
Pulling mosquitto (eclipse-mosquitto:2-openssl)...
2-openssl: Pulling from library/eclipse-mosquitto
ef5531b6e74e: Pull complete
cb9518cf6acf: Pull complete
636eec3ce25f: Pull complete
Digest: sha256:9a7eec108c926a5310f4de3c32a51b2bc665ce87d015a5059f8a1a25842e5cfd
Status: Downloaded newer image for eclipse-mosquitto:2-openssl
Pulling management-center (cedalo/management-center:2)...
2: Pulling from cedalo/management-center
cbdbe7a5bc2a: Pull complete
4c504479294d: Downloading [=========================>                        ]  17.79MB
/36MBb93d557: Download complete
227291017118: Download complete
d49291481acb: Download complete
3207412a4cc3: Download complete
26a8f922ed6b: Download complete
0d318063ee6f: Download complete
/5.649MB209f: Download complete
465f7242c085: Download complete
```

Рисунок 3.35 – Приклад запуску платформи Cedalo

Докер виконає деякі завантаження, якщо сценарій запускається вперше. Після завершення завантаження та запуску служби не треба закривати термінал, оскільки це також закриє й веб-інтерфейс керування.

За замовчуванням веб-інтерфейс центру керування працює на порту 8088, таким чином, щоб отримати доступ до нього, необхідно відкрити браузер і ввести:

```
http://localhost:8088
```

Приклад першого запуску подано на рис. 3.36.

Коли система запитує ім’я користувача та пароль, вводяться облікові дані за замовчуванням:

– користувач: cedalo;

– пароль: mmcisawesome.

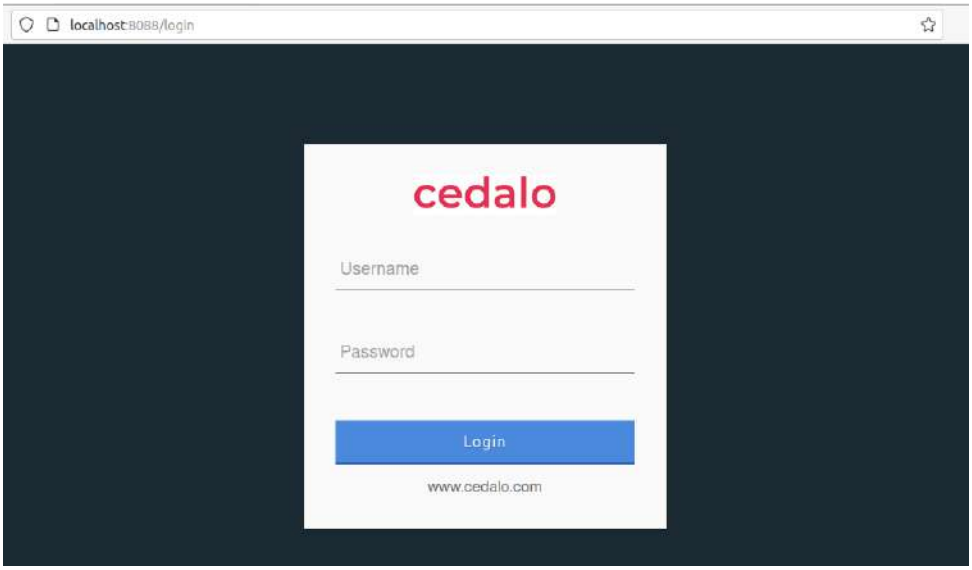


Рисунок 3.36 – Вхід в систему Cedalo

Щоб змінити дані користувача за замовчуванням, можна відредагувати файл docker-compose.yml. Крім того, URL-адреса підключення Eclipse Mosquitto:

```
mqtt://localhost:1883
```

В наступний раз вхід в систему буде виглядати таким чином (рис. 3.37).

Рисунок 3.37 – Вхід в систему за допомогою логіну та пароля

На рис. 3.38 подано приклад графічного інтерфейсу програми Cedalo.

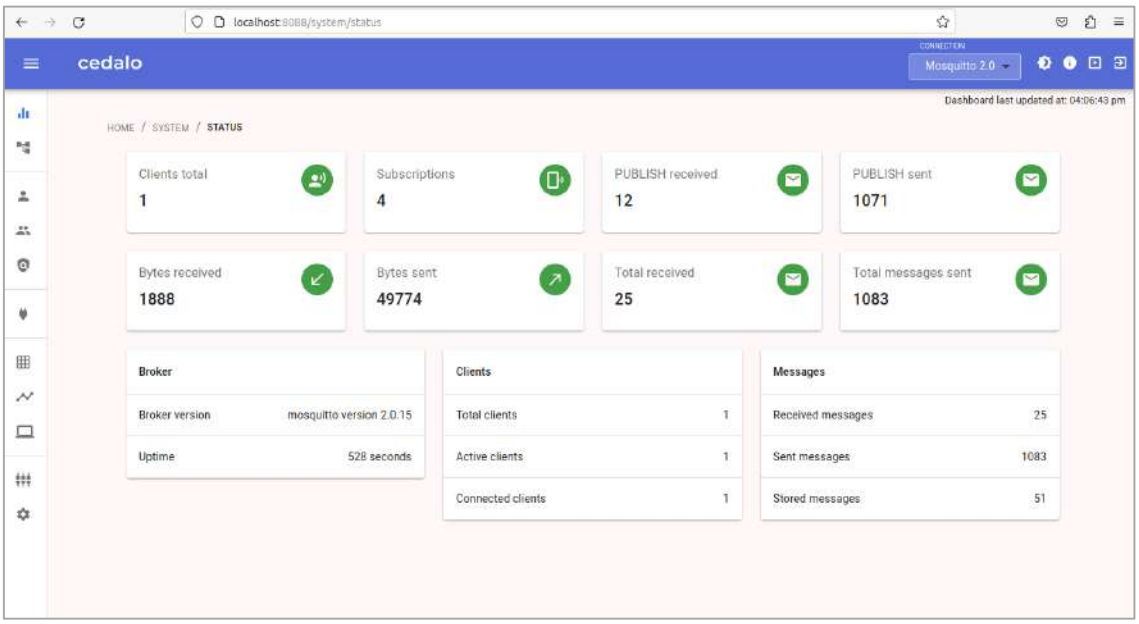


Рисунок 3.38 – Приклад графічного інтерфейсу програми Cedalo

Графічний інтерфейс надає можливість створювати нового користувача системи. Для цього потрібно в головному меню обрати пункт «Client» (1) та заповнити потрібні поля (2 та 3) (рис. 3.39).

В якості прикладу створимо користувача з ім’ям «testCedalo» та паролем «pas_testCedalo». Після заповнення полів потрібно зберегти інформацію, натиснувши кнопку «Save» (рис. 3.39, 4).

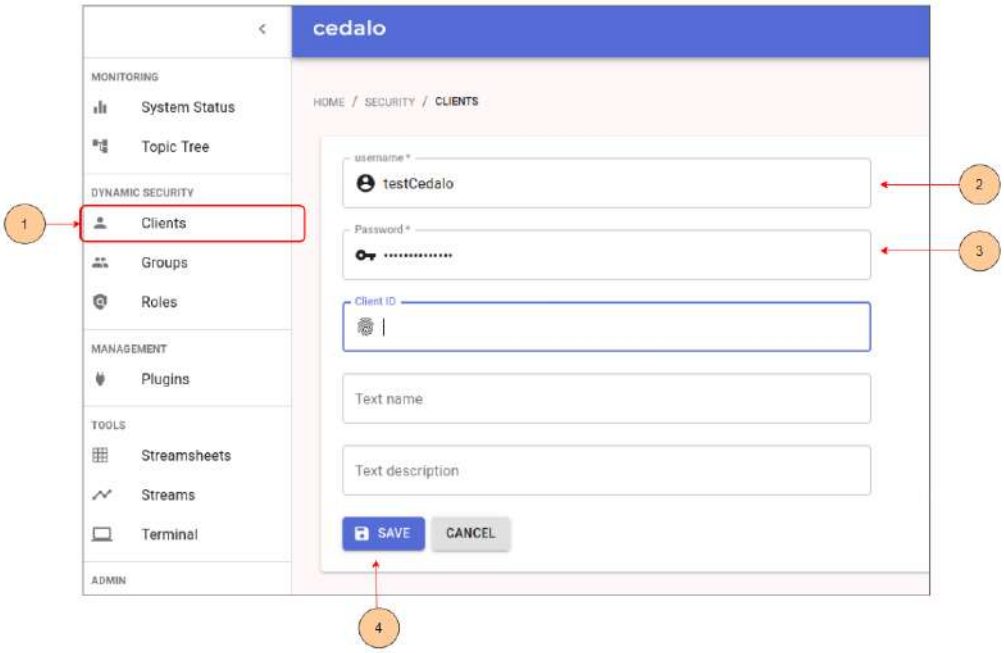


Рисунок 3.39 – Створення нового користувача

Після створення нового клієнта необхідно обрати для нього одну або декілька ролей (рис. 3.40).

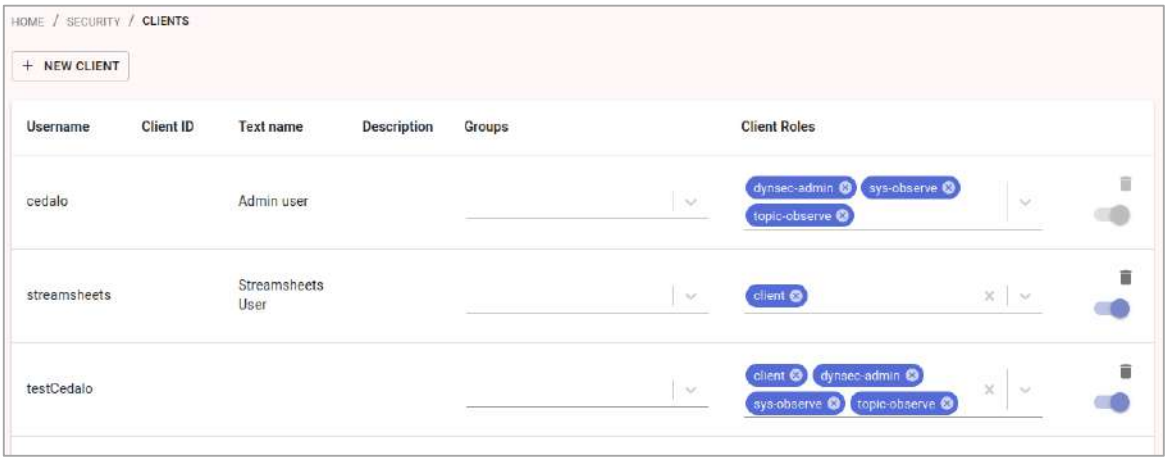


Рисунок 3.40 – Вибір необхідної ролі для користувача

Роль «Client» є обов’язковою для реалізації можливості створення та передплати на теми. Виконаємо перевірку правильності створення користувача, створивши нову тему та підписавшись на неї (рис. 3.41).

Підписка на тему відбувається за допомогою команди:

```
mosquitto_sub -h localhost -t example -u testCedalo -P pas_testCedalo
```

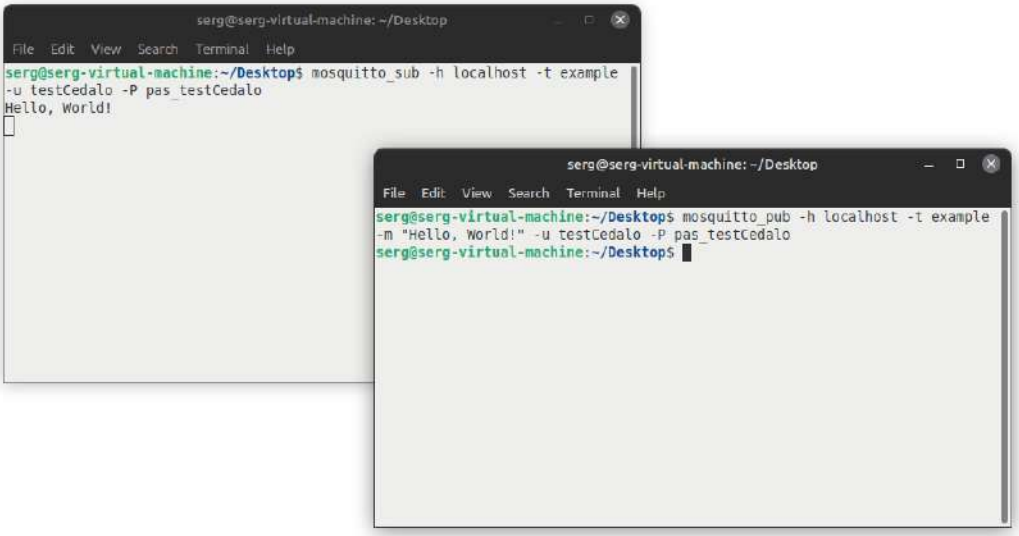


Рисунок 3.41 – Створення нової теми та передплата на неї

Публікація даних виконується наступним чином:

```
mosquitto_pub -h localhost -t example -m "Hello, World!" -u testCedalo -P pas_testCedalo
```

Поданий на рис. 3.41 приклад показав, що повідомлення доходять до підписника після публікації.

За допомогою графічного інтерфейсу Cedalo можна дізнатися інформацію про поточні теми та їх наповнення (рис. 3.42).

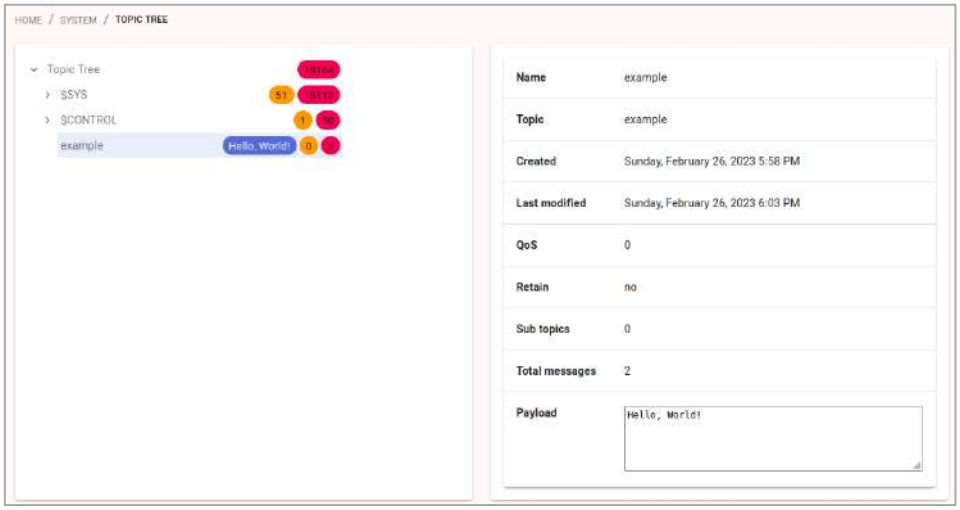


Рисунок 3.42 – Детальна інформація про поточні теми в брокері Eclipse Mosquitto

Кожна тема має бути досліджена за допомогою графічного інструменту. Можна дізнатись інформацію про назву теми, вміст останнього повідомлення, кількість підтем в темі та кількість повідомлень, що були надіслані в межах конкретної теми. На рис. 3.43 подана інформація про структуру теми.

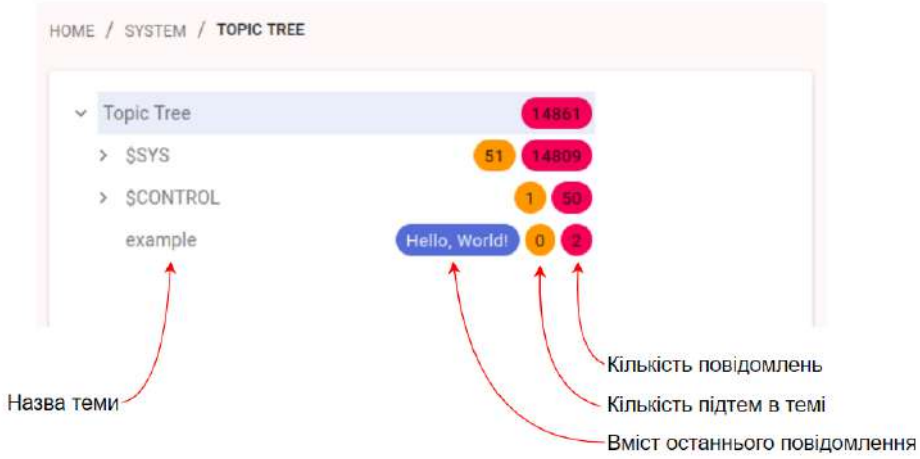


Рисунок 3.43 – Інформація про структуру теми

Для створення підтеми (subtopic) в Eclipse Mosquitto необхідно вказати її ім’я у підписці (subscribe) на відповідну тему (topic) з використанням символу «+» або «#» для вказання більш загальних або конкретних підтем.

Наприклад, якщо необхідно створити підтему «sensor1» в темі «example», то можна підписатися на тему «example/sensor1» з використанням символу «/» для розділення назви теми та підтеми.

Таким чином, всі повідомлення, які будуть публікуватися в темі «example/sensor1» або будь-якій її підтемі, будуть отримуватися підписаним на цю тему.

Наприклад, щоб підписатися на всі підтеми «sensors», можна використовувати підписку на тему «sensors/#». Або для підписки на всі підтеми для певного датчика, наприклад «sensor1», можна використовувати підписку на тему «example/sensor1/#».

Щоб створити публікацію повідомлення в певну підтему, необхідно вказати її ім’я у темі (topic) при публікації повідомлення. Наприклад:

```
mosquitto_pub -t " example/sensor1/temperature" -m "25"
```

Ця команда публікує повідомлення з темою «example/sensor1/temperature» і текстом «25». Всі підписані на цю підтему клієнти отримають це повідомлення (рис. 3.44).

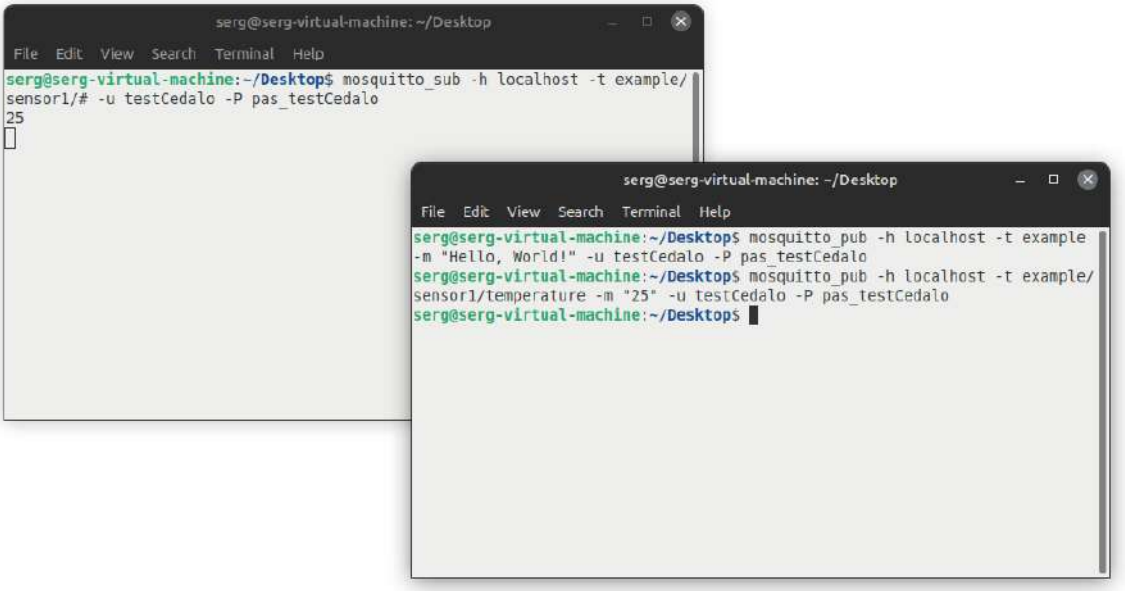


Рисунок 3.44 – Підписка та публікація в підтемі sensor1 теми example

Відповідно до доданої інформації зміниться вікно графічного інтерфейсу в Cedalo (рис. 3.45).

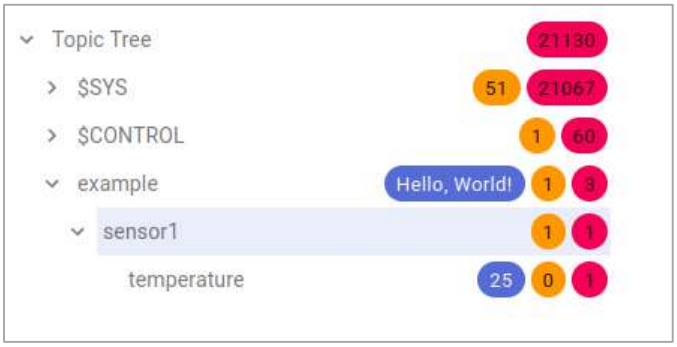


Рисунок 3.45 – Представлення інформації про підтеми та отримані повідомлення за допомогою Cedalo

Інформація в панелі керування оновлюється автоматично в реальному часі без необхідності перезавантаження інтерфейсу.

3.6 Безпека Інтернету речей

В даний час спостерігається збільшення кількості інцидентів (злочинів) у сфері інформаційної безпеки та інформаційних технологій. Цьому сприяє поширення мережевих технологій зберігання даних і IoT: у 2018 році кількість підключених пристроїв оцінювалася в 22 мільярди з перспективою зростання приблизно до 40 мільярдів до 2025 року (дані дослідницької компанії Strategy Analytics). Ці пристрої можуть містити вразливості, якими можуть скористатися кіберзлочинці і, в результаті, поставити під загрозу конфіденційність користувача та громадську безпеку.

Причиною цієї проблеми є той факт, що технології IoT, як і більшість споживчих технологій, розроблені без урахування вимог безпеки, оскільки основним завданням виробників було мінімізувати собівартість та час розробки, здешевити виробництво та збільшити обсяг продукції, що випускається. У результаті подібної політики розумні пристрої відчувають нестачу ресурсів. Через ці недоліки більшість інструментів безпеки не можуть бути встановлені в пристроях IoT, що робить пристрої легкою мішенню для кіберзлочинів.

Проблеми екосистеми IoT полягають у наступному:

– дуже велика площа атаки. Загрози та ризики, пов’язані з IoT, різноманітні та швидко розвиваються. З огляду на їх вплив на здоров’я, безпеку та конфіденційність користувачів, їх небезпеку не можна ігнорувати. Користувачі можуть не знати, що IoT значною мірою ґрунтується на зборі та обробці великих обсягів даних з різних джерел, включаючи конфіденційні дані, та обмін ними;

– складність екосистеми IoT. IoT слід розглядати не як сукупність незалежних пристроїв, а скоріше як багату, різноманітну і широку екосистему, яка включає пристрої, комунікації, інтерфейси і людей;

– відсутність законодавчих актів, норм, стандартів та правил. Фрагментоване та повільне прийняття стандартів та правил для впровадження заходів безпеки та передового досвіду у сфері IoT, а також постійна поява нових технологій ще більше ускладнюють відповідні проблеми;

– широке впровадження у критично важливі системи. Проникнення технологій IoT у критично важливу інфраструктуру, що має стратегічне значення для держави, є загрозою безпеці;

– складність інтеграції систем безпеки. Інтеграція систем безпеки – дуже складне завдання через наявність потенційно суперечливих точок зору та вимог усіх зацікавлених сторін. Наприклад, різні пристрої та системи IoT можуть використовувати різні рішення автентифікації, при цьому їх необхідно інтегрувати та зробити сумісними;

– економія виробників безпеки. Стрімке впровадження технологій IoT та розширення функціональних можливостей розумних пристроїв у низці критично важливих галузей створюють значний потенціал для скорочення витрат завдяки використанню потоків даних, розширеного моніторингу, інтеграції та інших можливостей. Водночас відносно низька вартість, якою зазвичай характеризуються IoT-пристрої та системи, має і негативні наслідки з погляду безпеки. З метою здешевлення продукції виробники нерідко обмежують захисні функції, унаслідок чого система безпеки пристрою може виявитися недостатньою для протидії окремим типам хакерських атак;

– нестача досвідчених фахівців. Це досить нова область, і тому не вистачає людей з відповідними навичками та досвідом у галузі безпеки IoT;

– складнощі із забезпеченням безпеки під час оновлення IoT-пристроїв. Забезпечити безпеку при встановленні оновлень до IoT надзвичайно складно, оскільки специфіка доступних користувачам інтерфейсів не дозволяє використовувати традиційні механізми оновлення. Забезпечення безпеки цих механізмів є непростим завданням, особливо з урахуванням використання бездротової мережі;

– складнощі із забезпеченням захисту на всіх етапах створення ПЗ. Оскільки кількість рішень для IoT стрімко зростає, виробники приділяють недостатньо часу забезпеченню безпеки та конфіденційності на етапі розробки. З цієї причини, а іноді і через фінансові проблеми, компанії, що розробляють продукти IoT, зазвичай звертають більше уваги на функціональність та зручність використання пристрою, ніж на його безпеку;

– відсутність чітко сформульованої відповідальності. Відсутність чіткого розподілу відповідальності може призвести до двозначності та конфліктів у разі інциденту, пов’язаного з безпекою, особливо з урахуванням великого та складного ланцюжка, характерного для виробництва IoT-пристроїв. Крім того, залишається без відповіді питання про те, як забезпечити безпеку, якщо у виробництві одного пристрою брало участь кілька різних сторін (юридичних

та/або фізичних осіб). Забезпечення відповідальності є ще однією важливою проблемою.

Екосистема IoT-технологій є комбінацією різних технологічних зон: зона IoT-пристроїв, мережева зона і хмарна зона. Ці зони можуть бути джерелом цифрових даних. Тобто дані можна збирати з розумного пристрою або датчика із внутрішньої мережі, такого як брандмауер або маршрутизатор, або із зовнішніх мереж (хмара або програма). Ці технологічні зони є об'єктом кримінального інтересу кіберзлочинців.

Для боротьби з кіберзлочинами був створений спеціальний розділ криміналістики – комп'ютерна криміналістика, або форензика (англ. Computer forensics), – прикладна наука про розкриття злочинів, пов'язаних з комп'ютерною інформацією, про дослідження доказів у вигляді комп'ютерної інформації, методи пошуку. IoT-криміналістика займається розслідуванням кіберзлочинів у системі IoT, дослідженням цифрових доказів.

Під час розслідування комп'ютерних інцидентів, пов'язаних із шахрайством або несанкціонованими кібератаками, у яких використовувалися мережеві з'єднання, проводиться цифрова експертиза – спеціалізоване дослідження, що передбачає аналіз застосування мережевих технологій та засобів передавання даних.

В залежності від місця зберігання даних у системі IoT експерти у сфері IoT-криміналістики виділяють три небезпечні ділянки у ландшафті кіберзагроз: хмара, мережа та пристрій, відповідно виділяються хмарна криміналістика, мережева та криміналістика на рівні пристрою IoT.

Оскільки цінні дані часто зберігаються у хмарі, хмарна інфраструктура є однією з найважливіших цілей зловмисників. Для проведення традиційної цифрової експертизи експерт-криміналіст спочатку отримує вилучене цифрове обладнання, а потім починає розслідування для отримання цифрових доказів (цифрових даних, які можна використовувати як доказ скоєння кіберзлочину). Однак, якщо дані зберігаються в хмарі, використовується інший сценарій, тому що цифрові докази можуть бути розміщені в хмарних сховищах на різних серверах і їх важко отримати звідти. Крім того, у хмарі обмежений доступ до інфраструктури та інформації про точне місце зберігання даних. Під час розслідування інциденту, що відбувся у хмарі, постачальник хмарних послуг може запросити інформацію про ім'я власника даних або місце зберігання відповідних даних.

Слід зазначити, що у хмарних сервісах, які використовують віртуальні машини як сервери, можуть зберігатися різні типи даних. Журнали подій, тимчасові інтернет-файли та інша службова інформація на сервері можуть бути втрачені або видалені, якщо вони не синхронізуються з постійними носіями даних, наприклад у разі перезапуску або вимкнення віртуального сервера.

Мережева криміналістика передбачає дослідження всіх типів мереж, що використовуються пристроями Інтернету речей для передавання та отримання даних. До таких мереж належать домашні, локальні, промислові мережі, а також мережі рівня MAN і WAN. У випадку інцидентів, пов'язаних з IoT-пристроями, потенційними джерелами доказів можуть бути журнали мережевого трафіку, зокрема записи брандмауерів, систем виявлення вторгнень (англ. Intrusion Detection System, IDS) та інших засобів моніторингу мережі.

Експертиза на рівні пристрою охоплює всі можливі цифрові докази, які можуть бути отримані безпосередньо з IoT-пристроїв. До таких доказів належать графічні, аудіо- та відеодані, наприклад записи з камер відеоспостереження, а також службові журнали роботи пристроїв.

Сучасні інструменти цифрової криміналістики не завжди повністю відповідають особливостям інфраструктури середовища IoT. Оскільки значна частина даних IoT зберігається у хмарних сервісах, саме вони стають одним із основних джерел цифрових доказів під час розслідування кіберінцидентів. Водночас отримання необхідної інформації з хмарного середовища є складним навіть для досвідчених експертів. Це пов'язано з тим, що на одному фізичному сервері можуть одночасно працювати кілька віртуальних машин, які належать різним користувачам, а також з тим, що доступ до хмарних сховищ може бути обмежений або втрачений після здійснення правопорушення.

Зазначені особливості ускладнюють проведення цифрової експертизи та зумовлюють необхідність розроблення нових методів і засобів розслідування кіберзлочинів у середовищі IoT.

3.7 Класифікація загроз промислового Інтернету речей

В таблиці 3.2 подано короткий опис можливих загроз промислового Інтернету речей.

Таблиця 3.2 – Класифікація загроз

Загроза	Опис
1. Умисні дії	
Шкідливе ПЗ	Програмне забезпечення, призначене для виконання небажаних та несанкціонованих дій у системі без згоди користувача Це ПЗ може призвести до пошкодження, модифікації або крадіжки інформації. Його небезпека може бути високою
Експлойт	Код, розроблений для використання вразливості для отримання доступу до системи. Цю загрозу важко виявити, і в середовищах IoT її небезпека варіюється від високої до критичної, залежно від порушених активів
Цільова атака	Атака, призначена для конкретної мети, яка проводиться протягом тривалого часу в кілька етапів. Основна мета злочинця – залишатися непоміченим і отримати якнайбільше конфіденційних даних, інформації чи контролю. Хоча небезпека цієї загрози є середньою, її виявлення, зазвичай, – дуже складний і тривалий процес
DDoS-атака	У процесі DDoS-атаки кілька систем атакують одну ціль, щоб навантажити її та призвести до збою. Це можна зробити шляхом створення низки з’єднань, переповнення каналу зв’язку або багаторазового повторного відтворення тих самих повідомлень
2. Перехоплення інформації	
Атака «людина посередині»	Активна атака підслуховування, за якої зловмисник передає повідомлення від однієї жертви іншій, щоб змусити їх повірити, що вони розмовляють безпосередньо один з одним
Підключення до активної сесії	Взяття під контроль активний сеанс зв’язку між двома елементами мережі. Зловмисник може отримати важливу інформацію, у тому числі конфіденційну
Перехоплення інформації	Несанкціоноване перехоплення та, іноді, модифікація приватної комунікації, наприклад телефонних дзвінків, миттєвих повідомлень, повідомлень електронної пошти
Мережева розвідка	Пасивне отримання внутрішньої інформації про мережу: підключені пристрої, протокол, відкриті порти, використовувані служби тощо
Перехоплення з’єднання	Крадіжка з’єднання для передачі даних, при цьому незаконний хост діє як законний з метою крадіжки, зміни або видалення даних, що передаються
3. Вимкнення	
Вимкнення живлення	Навмисне або випадкове переривання або збій у мережі. Залежно від порушеного сегмента мережі та часу, необхідного для відновлення, небезпека цієї загрози варіюється від високої до критичної
Збій пристрою	Несправність апаратного пристрою
Збій системи	Збій програмних служб або програм
Втрата сервісу підтримки	Недоступність послуг підтримки, необхідних для правильної роботи інформаційної системи

Продовження таблиці 3.2

Загроза	Опис
4. Технічний збій	
Уразливості на програмному рівні	Пристрої IoT часто вразливі через слабкі паролі, незмінені паролі, встановлені за замовчуванням, програмні помилки та помилки конфігурації
Сторонні помилки	Помилки в активному елементі мережі, викликані неправильним налаштуванням іншого елемента, що має до нього пряме відношення
5. Катастрофи	
Стихійні лиха	Повені, сильні вітри, сильні снігопади, зсуви ґрунту та інші стихійні лиха, які можуть пошкодити пристрої фізично
Аварії в середовищі IoT	Аварії в середовищі розгортання IoT-обладнання, що призводять до їх непрацездатності
6. Фізична атака	
Модифікація пристрою	Модифікація пристрою, внесення змін до пристрою (наприклад, використання поганої конфігурації портів, використання відкритих портів)
Знищення пристрою	Порча, крадіжка тощо

Класифікація загроз промислового Інтернету речей подана на рис. 3.46.

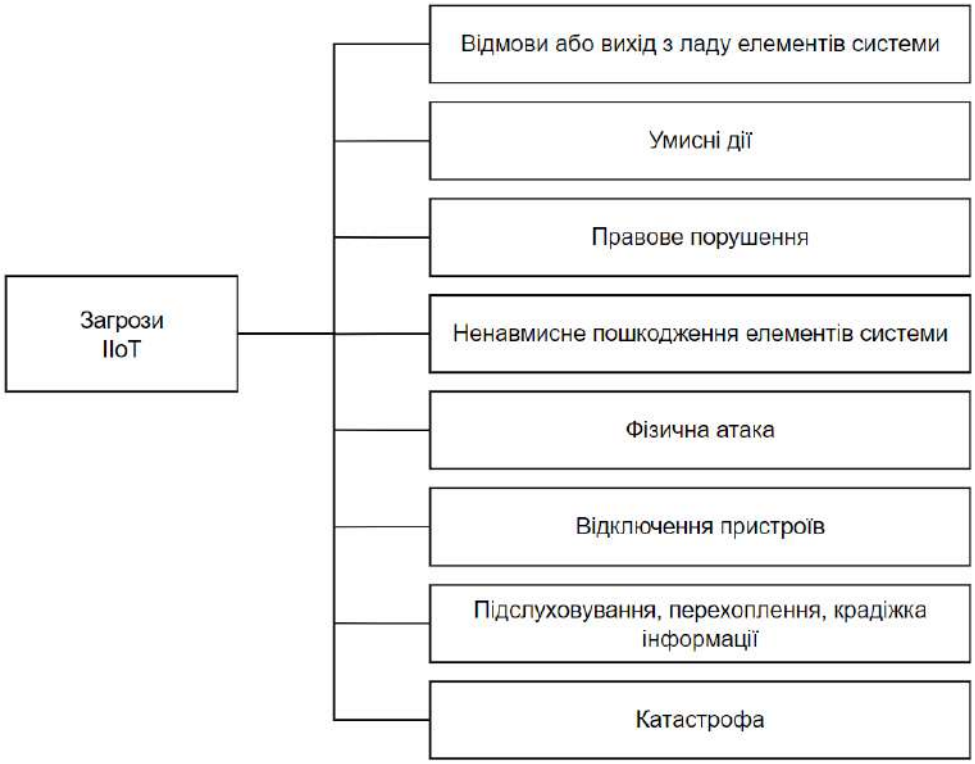


Рисунок 3.46 – Класифікація загроз промислового Інтернету речей

Розглянемо кожну із загроз більш докладніше.

3.7.1 Відмови або вихід з ладу елементів системи

Основні ознаки відмови або виходу з ладу елементів системи (рис. 3.47):

а) збій або вихід з ладу кінцевих IIoT-пристроїв виникає за умови неналежного обслуговування та недотримання посібників та інструкцій з експлуатації пристроїв;

б) відмова або вихід з ладу систем керування може статися, якщо не забезпечується належне обслуговування та дотримання посібників та інструкцій з експлуатації пристроїв;

г) експлуатація вразливості програмного забезпечення стає можливою через відсутність оновлень, використання слабких паролів або паролів за замовчуванням, а також неправильної конфігурації;

д) відмова або збій у постачальників послуг спричиняє порушення процесів, які залежать від сторонніх сервісів.

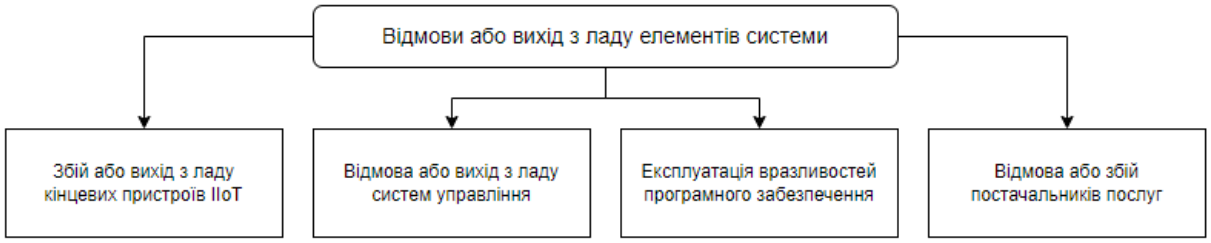


Рисунок 3.47 – Відмова або вихід з ладу елементів системи

3.7.2 Умисні дії

До умисних дій відноситься (рис. 3.48):

а) атака типу «відмова в обслуговуванні» – може мати двонаправлений характер. З одного боку, вона може бути спрямована безпосередньо на систему IIoT, коли на неї надсилається велика кількість запитів, що перевантажують ресурси та призводять до збоїв у роботі або повної недоступності сервісів (DoS-атака, від англ. Denial of Service – відмова в обслуговуванні). З іншого боку, зловмисник може використати значну кількість пристроїв IIoT у промисловому середовищі для формування бот-мережі та застосувати її як інструмент для здійснення атак на інші інформаційні системи. У такому випадку виконується розподілена атака типу відмови в обслуговуванні (DDoS-атака, від англ. Distributed Denial of Service);

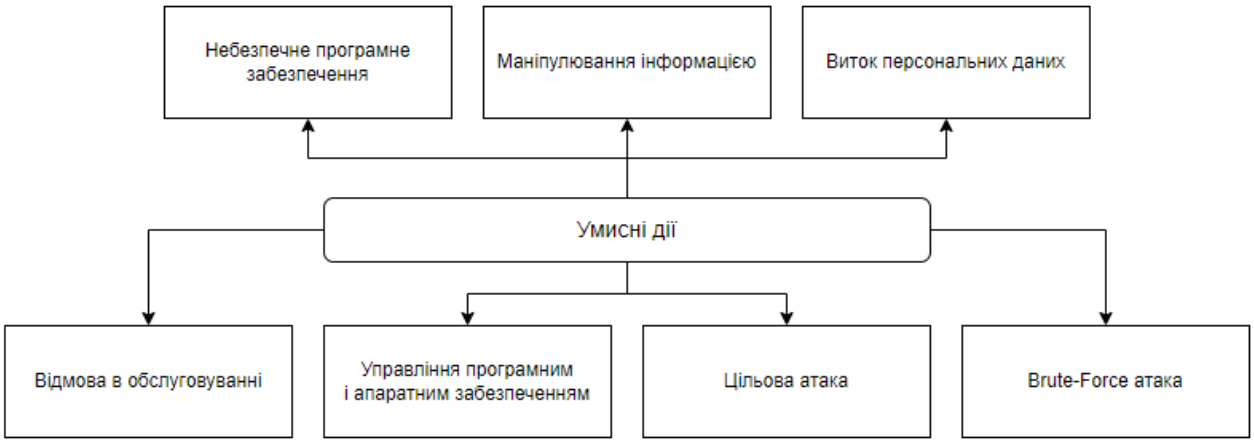


Рисунок 3.48 – Класифікація умисних дій під час атаки на IIoT інфраструктуру

б) шкідливе програмне забезпечення може проникати в систему IIoT з метою виконання несанкціонованих дій, що здатні порушити роботу системи, технологічних процесів або пов'язаних з ними даних. До типових прикладів таких загроз належать комп'ютерні віруси, троянські програми та шпигунське програмне забезпечення;

в) несанкціоноване керування програмним або апаратним забезпеченням, а також прикладними програмами пристроїв може здійснюватися зловмисником без відома користувача. У промислових системах IoT це може проявлятися у втручанні в роботу промислових роботів, зміні параметрів пристроїв або несанкціонованій модифікації їх конфігурації;

г) маніпулювання інформацією полягає у несанкціонованій зміні або підробці даних. Це може включати компрометацію операційних технологій (OT) або систем підтримки виробництва, таких як SCADA чи MES, а також зміну технологічних параметрів процесу. Наслідком може стати прийняття неправильних управлінських рішень на основі спотворених або сфальсифікованих даних;

д) цільова атака спрямована на конкретну організацію або окремих працівників з метою завдання шкоди чи отримання несанкціонованого контролю над системою. Такі атаки можуть здійснюватися шляхом компрометації ключових пристроїв, підміни телеметричних даних або введення в оману операторів. Додатковими ризиками є викрадення комерційної таємниці, технологічних формул, виробничих рецептів або іншої конфіденційної інформації. У сучасних умовах зловмисники можуть використовувати методи штучного інтелекту для реалізації персоналізованих атак, спрямованих на

окремих співробітників або обмежені групи користувачів. Такі атаки відрізняються від масових атак, які спрямовані на ураження великої кількості пристроїв через використання відомих вразливостей або шкідливих вебресурсів;

е) витік персональних даних може призвести до розголошення конфіденційної інформації, що зберігається на пристроях або у хмарних сервісах. Метою зловмисника є отримання несанкціонованого доступу до таких даних та їх подальше незаконне використання. У виробничих системах до подібної інформації можуть належати облікові записи користувачів, їхні ролі та права доступу. Хоча технологічні дані не завжди є конфіденційними, їх розголошення може створювати проблеми у випадку, якщо вони пов'язані з діяльністю конкретних працівників;

ж) атака типу brute-force (від англ. brute force – «груба сила») полягає у спробі отримання несанкціонованого доступу до ресурсів організації шляхом підбору паролів або ключів за допомогою перебору можливих комбінацій символів. Найбільш уразливими до таких атак є системи, у яких використовуються прості паролі або стандартні паролі за замовчуванням, особливо у промислових пристроях та системах керування.

3.7.3 Правове порушення

До правових порушень можна віднести (рис. 3.49):

а) порушення законодавства, норм, правил та зловживання персональними даними може призвести до юридичних проблем та фінансових втрат. Небезпека пов'язана з обробкою персональних даних, наприклад, при використанні кінцевих пристроїв IIoT без дотримання місцевих законів або норм;

б) невиконання вимог документації спричиняє порушення договірних вимог виробниками компонентів та постачальниками програмного забезпечення у разі неможливості забезпечити необхідні заходи безпеки.

Рисунок 3.49 – Класифікація правових порушень під час атаки на IIoT

3.7.4 Ненавмисне пошкодження елементів системи

До ненавмисного пошкодження елементів системи можна віднести (рис. 3.50):

а) ненавмисне зміна даних чи зміни у системі ОТ, виконане недостатньо навченим співробітником, може викликати порушення робочого процесса. Навіть із добрими намірами некваліфікований працівник, не підозрюючи про наслідки, може внести неналежні зміни до системи, особливо якщо він отримує повноваження, що перевищують необхідні;

б) некоректне використання або адміністрування пристроїв та систем IIoT /OT недостатньо навченим співробітником може призвести до порушення робочого процесу або фізичного пошкодження пристрою;

в) збитки, завдані третьою стороною, можуть призвести до пошкодження активів ОТ Якщо стороння організація має неконтрольований доступ до системи ОТ, наприклад з метою обслуговування або оновлення програмного забезпечення, порушення безпеки цією організацією можуть завдати шкоди компанії, яка отримує послугу.

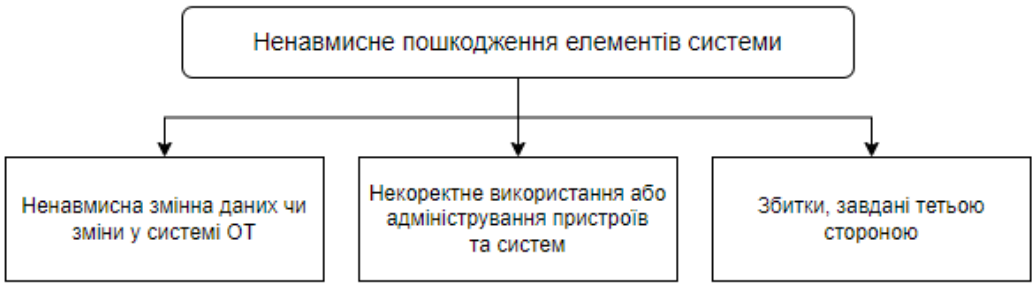


Рисунок 3.50 – Класифікація ненавмисних пошкоджень елементів системи

3.7.5 Фізична атака

До фізичної атаки на елементи системи можна віднести (рис. 3.51):

а) крадіжка та вандалізм можуть призвести до незапланованих простоїв виробництва, оскільки заміна пошкодженого або вкраденого пристрою потребує часу, іноді значного;

б) саботаж, диверсія можуть бути здійснені зловмисником при отриманні фізичного доступу до пристроїв внаслідок неправильної конфігурації портів та їх відкритості. Зловмисник також може використовувати доступ для виконання несанкціонованих дій оператора.

Рисунок 3.51 – Класифікація фізичної атаки на елементи системи

3.7.6 Вимкнення пристроїв

Вимкнення пристроїв характеризується наступними діями (рис. 3.52):

а) вимкнення мережі зв’язку може статися через проблеми з кабельною, бездротовою або мобільною мережею;

б) вимкнення електроживлення може стати результатом збою в роботі або виходу з ладу будь-якого джерела живлення та, у разі відсутності аварійного джерела живлення, призвести до серйозних наслідків через раптове припинення виробничих процесів;

в) втрата послуг підтримки відбувається внаслідок збою чи несправності систем, які підтримують виробництво чи логістику.

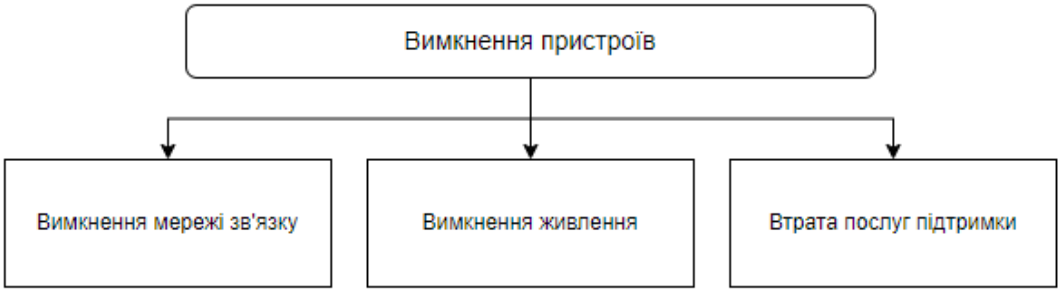


Рисунок 3.52 – Дії під час вимкнення пристроїв IIoT

3.7.7 Підслуховування, перехоплення, крадіжка інформації

Підслуховування, перехоплення, крадіжка інформації описується наступними діями (рис. 3.53):

а) «людина посередині» (MitM-атака, англ Man in the middle) – активна атака підслуховування, при якій зловмисник передає повідомлення від однієї жертви іншій, щоб змусити їх повірити, що вони розмовляють безпосередньо один з одним;

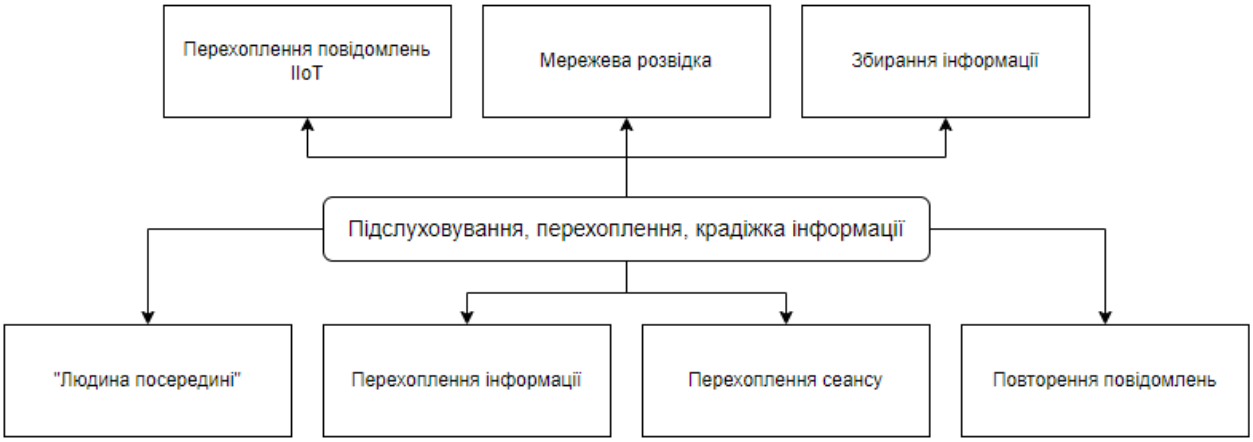


Рисунок 3.53 – Підслуховування, перехоплення, крадіжка інформації

б) перехоплення протоколу IIoT означає взяття під контроль існуючого сеансу зв'язку між двома елементами мережі. Зловмисник може прослуховувати цінну інформацію, зокрема паролі. У перехопленні можуть використовуватися агресивні методи, наприклад, примусове відключення або відмова в обслуговуванні;

в) перехоплення інформації включає несанкціоноване перехоплення (іноді модифікацію) особистих повідомлень, таких як телефонні дзвінки, миттєві повідомлення, повідомлення електронної пошти;

г) мережева розвідка передбачає пасивний і активний збір внутрішньої інформації про мережу: про підключені пристрої, протокол, відкриті порти, служби, що використовуються, тощо за допомогою загальнодоступних даних і додатків;

д) перехоплення сеансу (англ, session hijacking) передбачає перехоплення з'єднання для передачі даних і перемикання його на новий хост замість законного для крадіжки, зміни або видалення даних, що передаються;

є) збір інформації означає пасивне отримання внутрішньої інформації про мережу: про підключені пристрої, використовуваний протокол тощо;

ж) повторення повідомлень використовується як атака, щоб маніпулювати цільовим пристроєм або збивати його роботу за допомогою зловмисного використання допустимої передачі даних з багаторазовим відправленням або затримкою.

3.7.8 Катастрофа

Під час катастрофи виникають наступні процеси (рис. 3.54):

а) стихійне лихо, таке як повінь, удар блискавки, сильний вітер, дощ або снігопад, що може завдати фізичної шкоди компонентам довкілля ОТ;

б) екологічна катастрофа, наприклад пожежа, забруднення, вибух може призвести до фізичного пошкодження компонентів навколишнього середовища ОТ.

Рисунок 3.54 – Процеси, що виникають під час катастрофи

3.8 Контрольні запитання та завдання

1. Що таке промисловий Інтернет речей (IIoT) і які його основні відмінності від звичайного Інтернету речей?

2. Які рівні входять до архітектури IIoT та яке призначення кожного з них?

3. Які основні протоколи використовуються для організації зв'язку в системах IIoT?

4. Для чого використовується протокол MQTT у промисловому Інтернеті речей?

5. Які основні характеристики протоколу MQTT забезпечують його ефективність у системах IIoT?

6. Яку структуру має повідомлення у протоколі MQTT?

7. Яке призначення брокера MQTT та яку роль виконує Eclipse Mosquitto?

8. У чому полягає призначення графічної платформи Cedalo при роботі з брокером MQTT?

9. Які основні вимоги до забезпечення безпеки в системах Інтернету речей?

10. Які основні види загроз існують у промисловому Інтернеті речей та чим вони відрізняються?

4 МЕТОДИКА АНАЛІЗУ ТРАФІКУ ПРОМИСЛОВИХ МЕРЕЖ З МЕТОЮ ВИЯВЛЕННЯ НЕПЕРЕДБАЧУВАНОГО ВТРУЧАННЯ

4.1 Протокол Modbus. Базові поняття

Протокол Modbus та мережа Modbus є найпоширенішими у світі. Незважаючи на свій вік (стандартом Modbus став ще у 1979 році), Modbus не лише не застарів, але, навпаки, суттєво зросла кількість нових розробок та обсяг організаційної підтримки цього протоколу. Мільйони Modbus-пристроїв у всьому світі продовжують успішно працювати.

Однією з переваг Modbus є відсутність потреби у спеціальних інтерфейсних контролерах (Profibus та CAN вимагають для своєї реалізації замовні мікросхеми), простота програмної реалізації та елегантність принципів функціонування. Все це знижує витрати на опанування стандарту системними інтеграторами та розробниками контролерного обладнання.

Основним недоліком Modbus є мережевий обмін за типом «керуючий/підлеглий», що не дозволяє підлеглим пристроям передавати дані в міру їх появи і тому потребує інтенсивного опитування керуючими пристроями.

Різновидами Modbus є протоколи Modbus Plus – багатомайстерний протокол із кільцевою передачею маркера та Modbus TCP, розрахований на використання в мережах Ethernet та інтернет.

Протокол Modbus має два режими передачі: RTU (Remote Terminal Unit – «віддалений термінальний пристрій») та ASCII. Стандарт передбачає, що режим RTU в протоколі Modbus повинен бути обов’язково, а режим ASCII є опціонним. Користувач може вибирати будь-який з них, але всі модулі, включені в мережу Modbus, повинні мати той самий режим передачі.

Modbus RTU допускає один активний (ініціатор) пристрій в лінії (master), який може передавати команди одному або декільком пасивним пристроям (slave), звертаючись до них за унікальною в лінії адресою. Синтаксис команд протоколу дозволяє адресувати до 255 пристроїв на одній лінії зв’язку стандарту RS-232 або RS-485.

Ініціатива проведення обміну завжди виходить від активного (master) пристрою, наприклад ПК. Пасивні пристрої прослуховують лінію зв’язку.

Активний пристрій надсилає запит (посилка, послідовність байт) в лінію і переходить в стан прослуховування лінії зв'язку. Пасивний пристрій відповідає на запит, який прийшов на його адресу. Активний пристрій має встановлювати інтервал часу на приймання посилки даних. Якщо цей інтервал перевищив заданий час то генерується тайм-аут приймання даних.

Протокол забезпечує взаємодію у режимі Request/Response. Майстер ініціює запит до підлеглого пристрою, передаючи в PDU код функції та дані. Залежно від фізичного рівня мережі в пакеті можуть бути додаткові поля.

На рис. 4.1 подано принцип роботи протоколу Modbus у випадку відсутності помилок на підлеглому пристрої. Якщо обробка запита проходить без помилок, підлеглий пристрій повертає пакет, що містить вихідний код функції та запитані дані.

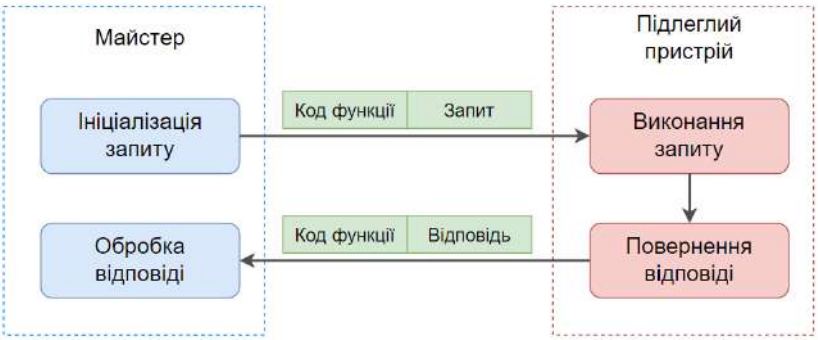


Рисунок 4.1 – Принцип роботи протоколу Modbus у випадку відсутності помилок на підлеглому пристрої

При виникненні помилки підлеглий пристрій повертає в якості даних код помилки, а замість вихідного коду функції – значення цієї помилки, збільшене на 128 (0x80 в шістнадцятковій системі HEX) (рис. 4.2).

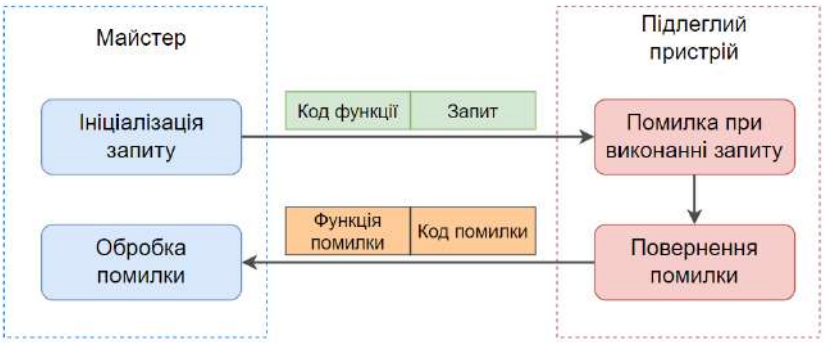


Рисунок 4.2 – Схема роботи Modbus у випадку помилок на підлеглому пристрої

Також передбачені тайм-аути з боку майстра, щоб уникнути тривалого очікування відповіді від пристроїв, що вийшли з ладу.

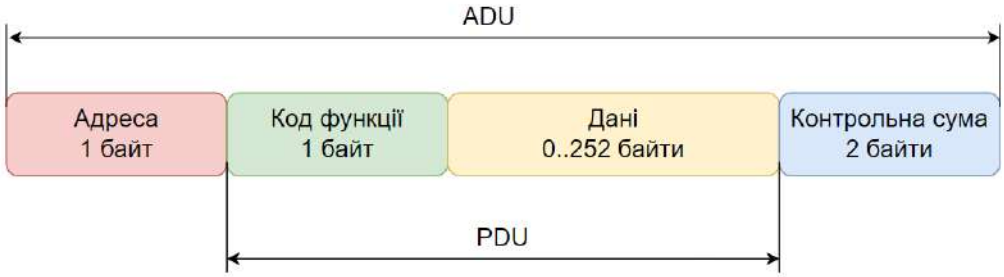
4.2 Організація обміну даними за протоколом Modbus

Протокол Modbus передбачає, що тільки один керуючий пристрій (контролер) і до 247 підлеглих (модулів вводу-виводу) можуть бути об'єднані в промислову мережу. Обмін даними завжди ініціюється керуючим. Підлеглі пристрої ніколи не починають передачу даних, доки не отримають запит від керуючого. Підлеглі пристрої не можуть обмінюватися даними один з одним. Тому будь-якої миті в мережі Modbus може відбуватися лише один акт обміну.

Адреси з 1 до 247 є адресами Modbus пристроїв у мережі, а з 248 до 255 зарезервовані. Керуючий пристрій не повинен мати адреси та в мережі не повинно бути двох пристроїв з однаковими адресами.

Керуючий пристрій може надсилати запити всім пристроям одночасно (широкомовний режим) або тільки одному. Для широкомовного режиму зарезервовано адресу «0» (при використанні команди цієї адреси вона приймається всіма пристроями мережі).

У протоколі Modbus RTU повідомлення починає сприйматися як нове після паузи (тиші) на шині тривалістю щонайменше 3,5 символів (14 біт), тобто величина паузи за секунди залежить від швидкості передачі. На рис. 4.3 подано формат кадру протоколу Modbus.



PDU – Protocol Data Unit (елемент даних протоколу);
ADU – Application Data Unit (елемент даних додатка)

Рисунок 4.3 – Формат кадру Modbus

Поле адреси завжди містить тільки адресу пристрою, навіть у відповідях на команду, надіслану майстром. Завдяки цьому керуючий пристрій знає, від якого модуля надійшла відповідь.

Поле «Адреса серверного пристрою» (Additional address) визначає, за якою адресою слід надіслати запит клієнта. Може приймати значення від 1 до 247. Адреса 0 використовується для широкомовної передачі даних від клієнта всім серверним пристроям (відповідь сервера при цьому не передбачена), а адреси 248…255 вважаються зарезервованими. У деяких реалізаціях протоколу поле ігнорується – наприклад, Modbus TCP, де найчастіше застосовується стандартна IP-адресація.

Поле «Код функції» говорить модулю про те, яку дію потрібно виконати. Це поле визначає, яку дію необхідно виконати на серверному пристрої. Значення кодів функцій лежать від 1 до 255, причому коди від 128 до 255 зарезервовані для повідомлень про помилки. Код 0 не використовується.

Для кодів з діапазонів 65-72 та 100-110 користувачі можуть реалізувати власні функції (User-Defined Function Codes). Деякі коди, наприклад 9, 10, 13 та інші, зарезервовані певними постачальниками для обладнання та закриті для загального використання (Reserved Function Codes). Коди, що не входять до цих двох підмножин, відносяться до публічних (Public Function Codes) – це задокументовані функції, що знаходяться у відкритому доступі.

Поле «Дані» може містити довільну кількість байт. У ньому може бути інформація про параметри, що використовуються в запитах контролера або відповіді модуля. Дані, необхідні для виконання вибраної функції серверного пристрою. Найчастіше це адреси регістрів для читання чи запису, їх кількість тощо. Довжина та формат поля залежать від коду функції. Деякі функції не потребують передачі даних.

Поле «Контрольна сума» містить контрольну суму CRC довжиною 2 байти. Дане поле містить розраховане за допомогою спеціального алгоритму число перевірки цілісності пакета. Як алгоритм для розрахунків використовується CRC-16 або LRC-8. У деяких реалізаціях протоколу поле відсутнє – наприклад, Modbus TCP, де контроль цілісності пакета забезпечується засобами протоколу TCP/IP.

4.3 Різновиди протоколу Modbus

Modbus – це протокол прикладного (сьомого) рівня моделі OSI. Він не залежить від нижчих рівнів і може використовуватися спільно з іншими протоколами, наприклад Ethernet TCP/IP або UDP/IP, а як фізичне середовище для передачі сигналів застосовувати послідовні інтерфейси RS-232, RS-422, RS-485, оптоволокно, радіоканали та інше (рис. 4.4).

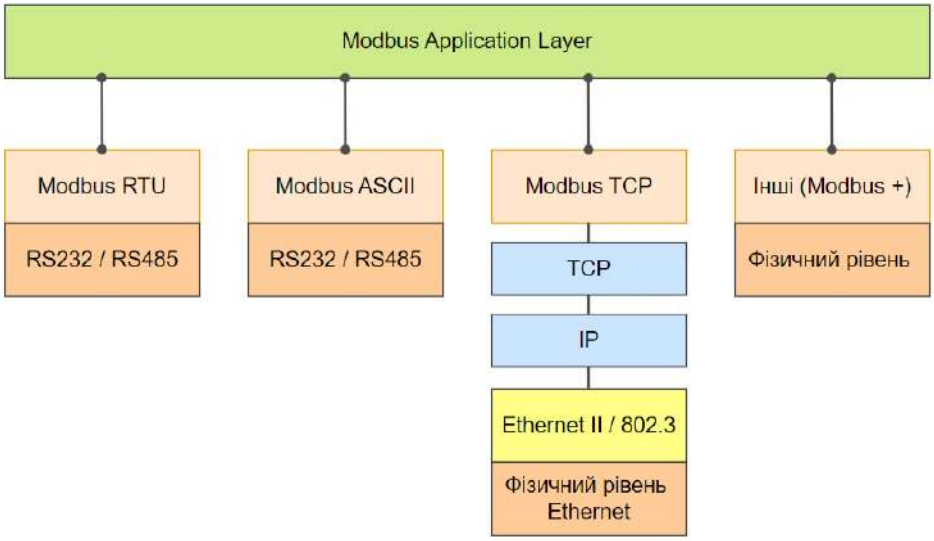


Рисунок 4.4 – Різновиди протоколу Modbus

4.3.1 Modbus RTU (Remote Terminal Unit)

Modbus RTU (Remote Terminal Unit) – це різновид протоколу, який як фізичний рівень мережі найчастіше використовує послідовний інтерфейс RS-485, рідше RS-232 і RS-422. По суті, всі ці інтерфейси визначають зв'язок за допомогою кручених пар, але відрізняються характеристиками виду максимальної довжини кабелю, кількості вузлів і таке інше.

Формат пакета Modbus RTU загалом збігається з узагальненою формою, описаною раніше: додаткові поля не використовуються. Контроль цілісності пакетів здійснюється за допомогою алгоритму CRC-16.

За замовчуванням в RTU режимі біт паритету встановлюють рівним «1», якщо кількість двійкових одиниць у байті непарне, і рівним «0», якщо воно парне. Такий паритет називають парним (even parity) та метод контролю називають контролем парності (рис. 4.5).

Рисунок 4.5 – Послідовність бітів у режимі RTU

За відсутності біта паритету на його місце записується другий стоп-біт. При парній кількості двійкових одиниць у байті, біт паритету може дорівнювати «1». У цьому випадку кажуть, що паритет є непарним (odd parity).

Контроль парності може бути відсутнім взагалі. У цьому випадку замість біту паритету має використовуватися другий стоповий біт. Для забезпечення максимальної сумісності з іншими продуктами рекомендується використовувати заміну біта паритету на другий стоповий біт.

Підлеглі пристрої можуть сприймати будь-який з варіантів: парний, непарний паритет або його відсутність.

Повідомлення Modbus RTU передаються у вигляді кадрів, кожному з яких відомо початок і кінець. Ознакою початку кадру є пауза (тиша) тривалістю щонайменше 3,5 шістнадцяткових символів (14 біт). Кадр повинен передаватися безперервно. Якщо в процесі передачі кадру виявляється пауза тривалістю більше 1,5 шістнадцяткових символів (6 біт), то вважається, що кадр містить помилку і повинен бути відхилений приймаючим модулем. Ці величини пауз повинні суворо дотримуватися при швидкостях нижче 19200 біт/с, проте для більш високих швидкостей рекомендується використовувати фіксовані паузи, 1,75 мс і 750 мкс відповідно. Приклад пакету даних в форматі Modbus RTU подано на рис. 4.6.

Рисунок 4.6 – Приклад пакету даних в форматі Modbus RTU

У режимі RTU є два рівні контролю помилок у повідомленні:

– контроль паритету для кожного байту (опційно);

– контроль кадру загалом з допомогою CRC методу.

Метод CRC використовується незалежно від перевірки паритету. Значення CRC встановлюється у провідному пристрої перед передачею. Коли повідомлення приймається, обчислюється CRC всього повідомлення і порівнюється з його значенням, зазначеним у полі CRC кадру. Якщо обидва значення збігаються, вважається, що повідомлення не містить помилки.

Стартові, стопові біти та біт паритету у обчисленні CRC не беруть участі.

4.3.2 Modbus ASCII

Modbus ASCII – це різновид протоколу Modbus, що також працює поверх інтерфейсів RS-232/RS-485, але для кодування повідомлень використовує ASCII-символи.

У порівнянні з Modbus RTU у форматі пакета додаються ще два поля – спеціальні символи для позначки початку та кінця повідомлення: двокрапка та символи повернення каретки / переведення рядка (рис. 4.7). Часові паузи між пакетами не потрібні. Для перевірки цілісності використовується алгоритм LRC-8.

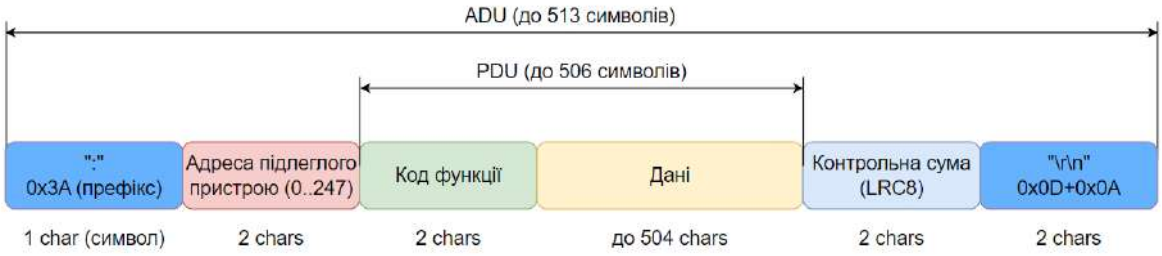


Рисунок 4.7 – Структура пакета даних в Modbus ASCII

Загалом цей варіант протоколу зараз використовується вкрай рідко – через складності кодування та великий розмір повідомлень. Однак він може стати гарною альтернативою Modbus RTU на лініях з мережними затримками та устаткуванні з менш точними таймерами.

4.3.3 Modbus TCP

Modbus TCP – це реалізація ModBus у мережах Ethernet, що працює поверх TCP/IP стека. На відміну від Modbus RTU і ASCII, Modbus TCP з’єднання

встановлюється з конкретним пристроєм засобами TCP/IP. Тому адреса в пакеті Modbus найчастіше ігнорується, а широкомовне розсилання повідомлень не використовується. Однак адреса може бути потрібна, якщо з'єднання встановлюється зі шлюзом, який, у свою чергу, виводить на мережу RS-485, щоб далі спілкуватися з пристроями вже мовою Modbus.

Контроль цілісності пакетів також забезпечується засобами протоколу TCP/IP, тому немає необхідності його Modbus-реалізації.

Поряд з адресою в заголовку пакета Modbus TCP є ряд додаткових полів (рис. 4.8):

– ID транзакції (або ID обміну). Це поле найчастіше заповнюється нулями. Поле необхідно для випадків, коли клієнтський пристрій надсилає кілька повідомлень, не чекаючи відповіді на попередні, щоб потім зв'язати відповіді із запитами;

– ID протоколу. Поле завжди заповнюється нулями, зарезервовано для майбутнього використання;

– довжина залишку пакета. Довжина частини пакета, що залишилася: адреси і PDU (коду функції і даних).

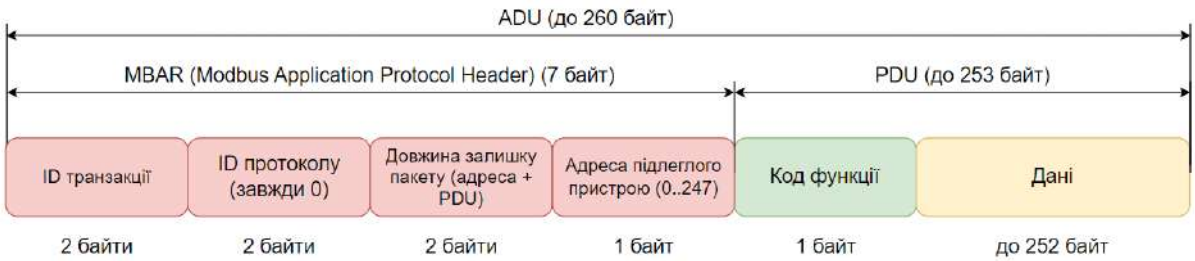


Рисунок 4.8 – Структура пакету протоколу Modbus TCP

4.4 Регістри та функції протоколу Modbus

4.4.1 Стандартна адресація регістрів Modbus

Оскільки Modbus призначений для роботи з промисловою автоматикою, обмін даними з Modbus-пристроями відбувається через регістри. Вони діляться на входи та виходи. Входи можна лише читати, а виходи – читати та писати. Бувають 1-бітні регістри Modbus для опису дискретних входів/виходів (Discrete Inputs та Coils) та 16-бітові регістри для аналогових входів/виходів (Input Registers та Holding Registers) [7, 8].

Доступ до регістрів здійснюється за допомогою 16-бітної адреси. Першому елементу в кожній групі регістрів відповідає адреса 0. Тобто адреса будь-якого регістру може приймати значення діапазону 0-65535 (0x0000-0xFFFF в HEX-форматі). При цьому специфікація протоколу не визначає, що фізично представляють собою адресні простори і за якими внутрішніми адресами пристрою повинні бути доступні регістри. У випадку значення регістрів з однаковою адресою, але різними типами відрізняються один від одного.

В таблиці 4.1 наведено стандартну адресацію регістрів Modbus.

Зазвичай, дискретні виходи починаються з адреси (осередка) 00001, а дискретні входи – з адреси 10001. Кожному з них потрібно один біт пам'яті. Вміст вхідних регістрів (в термінології контролерів вони іменувалися contacts) можна тільки читати, в той час як вміст вихідних регістрів (в термінології ПЛК вони іменувалися coils) можна і читати, і записувати.

Таблиця 4.1 – Таблиця розподілення регістрів Modbus

Адреса регістрів	Опис
00001 - 10000	1-бітні дискретні виходи (читання / запис) «Coils»
10001 - 20000	1-бітні дискретні входи (читання) «Discrete Inputs»
30001 - 40000	16-бітові аналогові входи (читання) «Input Registers»
40001 - 50000	16-бітові регістри зберігання (читання / запис) «Holding Registers»

Регістри аналогових входів і входів є 16-розрядними. Їх адреси починаються з 30001 – це адреса першого аналогового входу (тільки читання, наприклад, для введення сигналів від датчика температури).

З адреси 40001 починається діапазон універсальних регістрів (читання і запис), які можуть служити також і аналоговими виходами.

Залежно від фірми-виробника ПЛК ці регістри можуть бути внутрішніми регістрами, аналоговими входами, аналоговими виходами і навіть дискретними входами і виходами. Однак не всі функції працюють з адресами цих регістрів.

Перелік основних функцій протоколу ModBus наведено у таблиці 4.2.

Незважаючи на те що кодам функцій відведений діапазон від 1 до 127, в якості призначених для загального користування кодів визначені приблизно 20 кодів. В цей же діапазон входять коди, призначення яких визначається

кожним окремим користувачем. Слід мати на увазі, що багато Modbus-пристроїв підтримують тільки невеликі підмножини наявних кодів.

Таблиця 4.2 – Стандартні функції протоколу ModBus

Код	Розрядність	Опис	Діапазон адрес входів-виходів
1	1	Read coils Читання поточного стану (ON / OFF) дискретних виходів	00001 - 10000
2	1	Read contacts Читання поточного стану (ON / OFF) дискретних входів	10001 - 20000
5	1	Write a single coil Зміна стану дискретного виходу в ON або OFF	00001 - 10000
15	1	Write multiple coils Зміна стану (ON / OFF) декількох дискретних виходів	00001 - 10000
3	16	Read holding registers Читання регістрів зберігання	40001 - 50000
4	16	Read input registers Читання вхідних регістрів	30001 - 40000
6	16	Write single register Запис одного регістра	40001 - 50000
16	16	Write multiple registers Запис декількох регістрів	40001 - 50000
22	16	Mask write register Маскований запис регістра	40001 - 50000
23	16	Read/write multiple registers Читання / запис декількох регістрів	40001 - 50000
24	16	Read FIFO queue Читання вмісту черги FIFO	40001 - 50000

4.4.2 Опис функції читання вихідних контактів (дискретних виходів) (01)

Функція дозволяє користувачеві отримати статус вихідних контактів. Широкомовний режим не підтримується.

Крім полів адреси та функції, повідомлення вимагає, щоб інформаційне поле містило логічну адресу початкового біту і кількість бітів, статус яких необхідно отримати.

Адресація дозволяє отримати за один запит до 2000 логічних осередків. Однак, деякі прилади мають обмеження на максимальне число бітів, статус яких можна отримати за один запит. Біти нумеруються з нуля (біт 1 = 0, біт 2 = 1 і т.д.).

Припустимо, нам потрібно звернутися до підлеглого пристрою з адресою 11 та прочитати 19 його Coil-регістрів з номерами 20-38. Адресація дискретних виходів починається з 0, тому адреса першого потрібного нам виходу буде 0x13 (це 19 у HEX-системі). Необхідна для читання кількість виходів також дорівнюватиме 0x13 (для читання запитано 19 виходів). Адреса пристрою 0x11 (число 17 в HEX-форматі) та код функції 01. Контрольна сума формується за алгоритмом CRC-16 на основі інших полів пакета.

У таблиці 4.3 представлений запит на читання дискретних виходів 0020-0038 з приладу з адресою 17 (0x11). У таблиці 4.4 представлений приклад відповіді.

Таблиця 4.3 – Приклад запита для читання дискретних виходів 0020-0038

Байт	Опис полів запита
11	Адреса підлеглого пристрою
01	Код функції
00	Адреса початкового біта, Hi байт
13	Адреса початкового біта, Lo байт
00	Кількість дискретних виходів, Hi байт
13	Кількість дискретних виходів, Lo байт
8E	Контрольна сума CRC
92	Контрольна сума CRC

Таблиця 4.4 – Приклад повідомлення у відповідь

Байт	Опис полів відповіді
11	Адреса підлеглого пристрою
01	Код функції
03	Кількість байт в полі даних
A3	Статус вихідних контактів, перший байт
24	Статус вихідних контактів, другий байт
07	Статус вихідних контактів, третій байт
94	Контрольна сума CRC
3E	Контрольна сума CRC

На рис. 4.9 подано приклад обміну повідомленнями при використанні функції 01 (читання дискретних виходів).

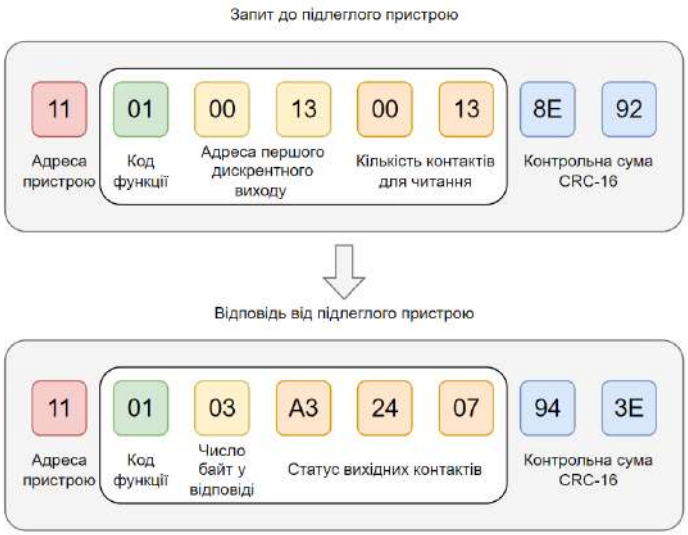


Рисунок 4.9 – Приклад обміну повідомленнями при використанні функції 01 (читання дискретних виходів)

Відповідь містить адресу, код функції, число байт в полі даних, дані і контрольну суму. Дані в полі даних у відповіді представлені у вигляді один біт на кожен осередок.

Молодший значущий біт першого байту поля даних містить перший осередок, за яким слідують інші. Якщо число осередків не ділиться на 8, то інші біти заповнюються нулями в порядку від старших бітів до молодших.

Статус осередків 20-27 дорівнює 0xA3 = 1010 0011. Читаючи зліва направо, бачимо, що осередки 27, 25, 21 і 20 встановлені. Інші дані розбираються так само. Так як було запитано число регістрів, що не ділиться на 8, старші п’ять бітів в останньому байті даних (0x07) заповнені нулями.

Принцип побудови послідовності відповідних байтів у зворотному повідомленні подано на рис. 4.10.

На даному прикладі показано випадок, коли робиться запит на читання десяти вхідних контактів, починаючи з третього входу (адреса входу DI2). Область, що підлягає зчитуванню виділена синім в модулі введення дискретних сигналів.

Нижче на рис. 4.10 подано принцип збирання бітів у байти відповіді. Молодші біти переходять до правої частини результуючого байту, а старші – до

лівої. Якщо запит прийшов на кількість бітів, що менша за кількість бітів в байті, біти, що залишаються, заповнюються нулями (другий байт відповіді 0x02).

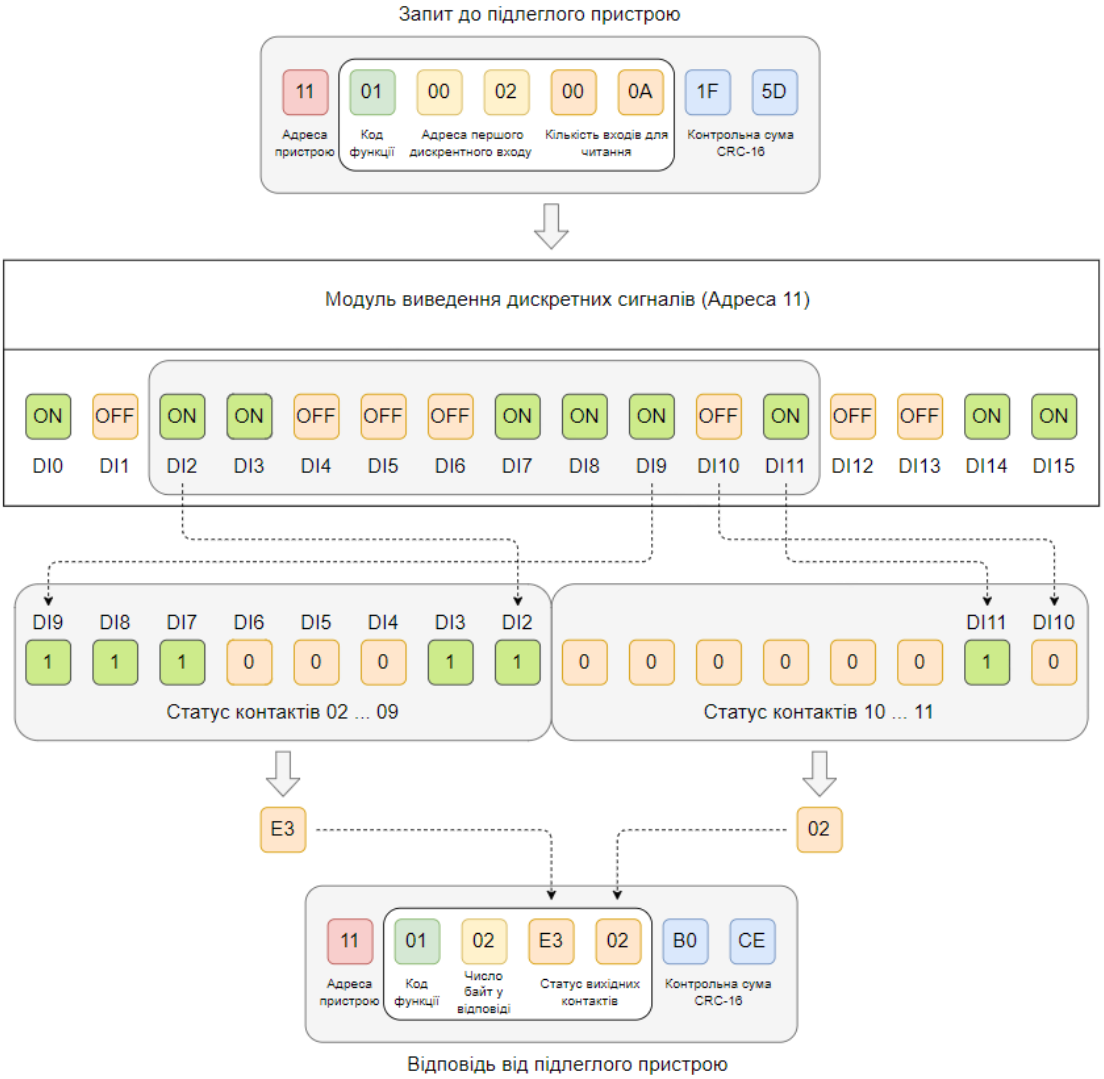


Рисунок 4.10 – Принцип побудови послідовності відповідних байтів у зворотному повідомленні

Запит обслуговується в кінці робочого циклу приладу, тому потрібно мати на увазі, що дані у відповідному повідомленні відображають стан регістрів на той момент.

4.4.3 Функція читання дискретних входів (02)

Ця функція дозволяє користувачеві отримати стан (ВМК / ВИМК) вхідних дискретних ліній пристрою, що адресується. Циклічний запит не підтримується. На додаток до адреси пристрою і номеру функції, запит вимагає, щоб інформаційне поле містило початкову адресу і кількість необхідних ліній.

Адресація дозволяє отримати за один запит до 2000 ліній. Однак, деякі пристрої мають обмеження на максимальну кількість ліній, одержуваних за один запит. Вхідні лінії нумеруються з нуля (10001 = 0, 10002 = 1 і т.д.).

У таблиці 4.5 представлений приклад запита на читання дискретних входів 10197-10218 з пристрою з адресою 17 (0x11).

Таблиця 4.5 – Приклад запита на читання дискретних входів 10197-10218

Байт	Опис полів запита
11	Адреса підлеглого пристрою
02	Код функції
00	Адреса початкового біта, Hi байт
C4	Адреса початкового біта, Lo байт
00	Кількість дискретних входів, Hi байт
16	Кількість дискретних входів, Lo байт
BA	Контрольна сума CRC
A9	Контрольна сума CRC

Приклад відповіді на цей запит представлений в таблиці 4.6. Відповідь включає адресу пристрою, код функції, кількість байт даних, дані і поле контрольної суми.

Таблиця 4.6 – Приклад повідомлення у відповідь на функцію 02

Байт	Опис полів відповіді
11	Адреса підлеглого пристрою
02	Код функції
03	Кількість байт в полі даних
AC	Статус вхідних контактів, перший байт
DB	Статус вхідних контактів, другий байт
35	Статус вхідних контактів, третій байт
20	Контрольна сума CRC
18	Контрольна сума CRC

Дані упаковані по біту на кожен вхід (1 = ON, 0 = OFF). Молодший біт першого байту містить значення першого входу, що адресується, за яким

слідують інші. Якщо кількість запитаних входів не кратне 8, то інші біти заповнюються нулями. Кількість байт даних завжди визначається як кількість RTU даних.

Так як пристрій обслуговує запит в кінці робочого циклу, дані у відповіді відображають стан входів на даний момент. Деякі пристрої мають обмеження на максимальну кількість входів, запитуваних за один запит.

Статус входів 10197-10204 = 0xAC = 1010 1100. Читаючи зліва направо, бачимо, що входи 10204, 10202, 10200 і 10199 в стані ON. Всі інші байти даних розпаковуються аналогічно.

Так як була запрошена інформація про стан 22 вхідних контактів, останній байт даних (35h = 0011 0101) містить інформацію тільки про 6 входів (10213-10218) замість 8-ми. Два останніх біта заповнюються нулями.

Приклад обміну повідомленнями при використанні функції 02 (читання дискретних входів) подано на рис. 4.11 та 4.12.

Рисунок 4.11 – Приклад запита для читання дискретних входів 10197-10204

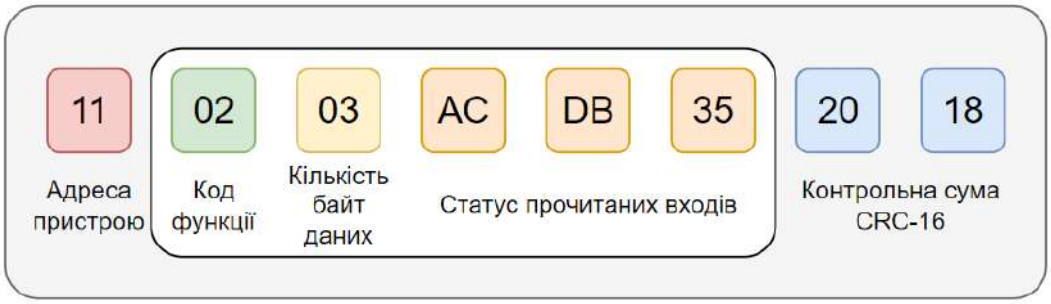


Рисунок 4.12 – Приклад відповіді на запит читання дискретних входів

4.4.4 Функція читання регістрів зберігання (03)

Ця функція дозволяє отримати бінарний вміст 16-ти розрядних регістрів підлеглого пристрою. Адресація дозволяє отримати за кожен запит до 125 регістрів. Однак, деякі пристрої мають обмеження на максимальну кількість

регістрів, які отримуються за один запит. Регістри нумеруються з нуля (40001 = 0, 40002 = 1 і т.д.). Широкомовний режим не допускається.

У таблиці 4.7 представлений приклад запита на читання регістрів 40108-40110 з пристрою з адресою 17 (0x11).

Таблиця 4.7 – Приклад повідомлення у відповідь на функцію 03

Байт	Опис полів запита
11	Адреса підлеглого пристрою
03	Код функції
00	Адреса першого байта, Hi байт
6B	Адреса першого байта, Lo байт
00	Число регістрів для читання, Hi байт
03	Число регістрів для читання, Lo байт
76	Контрольна сума CRC
87	Контрольна сума CRC

Пристрій, що адресується посилає у відповіді свою адресу, код виконаної функції і інформаційне поле.

Інформаційне поле містить 2 байти, які описують кількість байт даних, що повертаються. Довжина кожного регістру даних – 2 байти. Перший байт даних в посилці є старшим байтом регістру, другий – молодшим.

Так як пристрій зазвичай обслуговує запит в кінці свого робочого циклу, дані у відповіді відображають вміст регістрів в даний момент.

Деякі пристрої обмежують кількість регістрів, переданих за один запит. У цьому випадку для отримання, більшого числа регістрів, необхідно виконати кілька послідовних запитів.

У таблиці 4.8 представлений приклад відповіді на запит читання регістрів 40108-40110, які мають вміст, відповідно, 555, 0, 100, з пристрою з адресою 17 (0x11).

Таблиця 4.8 – Приклад відповіді на функцію 03

Байт	Опис полів відповіді
11	Адреса підлеглого пристрою
03	Код функції
06	Кількість байт в полі даних
02	Значення першого байта даних, Hi байт
2B	Значення першого байта даних, Lo байт
00	Значення другого байта даних, Hi байт
00	Значення другого байта даних, Lo байт
00	Значення третього байта даних, Hi байт
64	Значення третього байта даних, Lo байт
C8	Контрольна сума CRC
BA	Контрольна сума CRC

4.4.5 Запис одного регістра зберігання (06)

Ця функція використовується для запису одного регістра зберігання у віддаленому пристрої. PDU запита вказує адресу регістра, який потрібно записати. Адреси регістрів починаються з нуля, тому регістр під номером 1 адресується як 0. В таблиці 4.9 подано приклад запита для запису в регістр 40136 числа 926 (0x039E). Адреса пристрою 17 (0x11).

Таблиця 4.9 – Приклад запису в регістр 40136 числа 926

Байт	Опис полів запита
11	Адреса підлеглого пристрою
06	Код функції
00	Адреса першого байта, Hi байт
87	Адреса другого байта, Lo байт
03	Значення байта даних, Hi байт
9E	Значення байта даних, Lo байт
5A	Контрольна сума CRC
D3	Контрольна сума CRC

У випадку вдалого виконання запита у відповідь буде надіслане повідомлення, що ідентичне запита. Приклад відповіді подано в таблиці 4.10.

Таблиця 4.10 - Приклад відповіді на запит запису в регістр 40136 числа 926

Байт	Опис полів відповіді
11	Адреса підлеглого пристрою
06	Код функції
00	Адреса першого байта, Hi байт
87	Адреса другого байта, Lo байт
03	Значення байта даних, Hi байт
9E	Значення байта даних, Lo байт
5A	Контрольна сума CRC
D3	Контрольна сума CRC

Розглянемо приклад застосування даної функції протоколу Modbus на практиці. Наприклад, необхідно змінити режим роботи контролера, що керує роботою вентилятора. Для зміни режиму роботи необхідно змінити зміст регістра 40004 (рис. 4.13). Якщо ми хочемо перевести вентилятор в економний режим, нам потрібно змінити режим роботи на «1». В даному режимі двигун буде обертатися з мінімальною швидкістю 150 об/хв.

На рис. 4.13 подано приклад запита для переведення вентилятора в режим роботи №1.

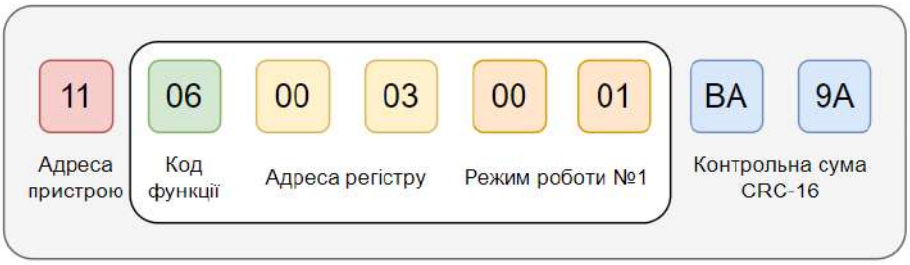


Рисунок 4.13 – Приклад запита для зміни режиму роботи двигуна вентилятора

Очікувана відповідь повторює запит (рис. 4.14).

4.4.6 Запис декількох регістрів зберігання (0x10)

Часто виникають задачі модифікації відразу декількох регістрів зберігання. Для цього в протоколі Modbus передбачена функція з номером

16 (0x10). В таблиці 4.11 подано приклад запита для запису в регістри 40136 та 40137 значень 0x00a0 та 0x0102. Адреса пристрою 17 (0x11).

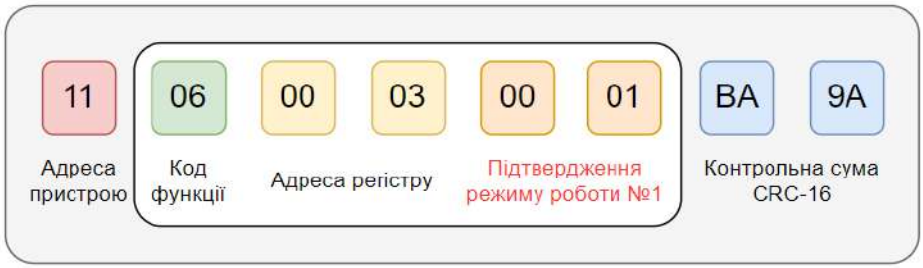


Рисунок 4.14 – Підтвердження запису в регістр 40003 режиму роботи двигуна вентилятора №1

Таблиця 4.11 – Приклад запита для запису значень в регістри 40136 та 40137

Байт	Опис полів запита
11	Адреса підлеглого пристрою
10	Код функції
00	Адреса першого байта, Hi байт
87	Адреса першого байта, Lo байт
00	Кількість регістрів для запису, Hi байт
02	Кількість регістрів для запису, Lo байт
04	Кількість байт в полі даних
00	Значення першого байта даних, Lo байт
0A	Значення першого байта даних, Lo байт
01	Значення другого байта даних, Hi байт
02	Значення другого байта даних, Lo байт
4E	Контрольна сума CRC
BA	Контрольна сума CRC

Повідомлення дозволяє записувати регістри з максимальною логічною адресою до 0xFFFF. Не використані старші біти адреси регістрів повинні заповнюватися нулями. Якщо використовується широкомовний запит, то вміст поля даних записується в усі пристрої, підключені до шини. У випадку вдалого виконання запита у відповідь буде надіслане повідомлення, що містить адресу першого байту, та кількість байт даних, що було записано. Приклад відповіді подано в таблиці 4.12.

Таблиця 4.12 – Відповідь на запит щодо запису даних в два регістри зберігання

Байт	Опис полів відповіді
11	Адреса підлеглого пристрою
10	Код функції
00	Адреса першого байта, Hi байт
87	Адреса другого байта, Lo байт
00	Кількість записаних регістрів, Hi байт
02	Кількість записаних регістрів, Lo байт
F3	Контрольна сума CRC
71	Контрольна сума CRC

Розглянемо приклад застосування даної функції протоколу Modbus на практиці. Наприклад, необхідно за один сеанс обміну інформацією змінити режим роботи контролеру, що керує роботою вентилятора, та швидкість передавання даних в мережі.

Режим роботи пристрою змінюється за допомогою регістра 40004, до якого записується число в діапазоні від 0 до 4. Швидкість передавання даних задається в залежності від значення, що міститься в регістрі 40005. На рис. 4.15 подано приклад запита, в якому міститься інформація про новий режим роботи (режим №2), та про нову швидкість передавання даних (19200 біт/с).

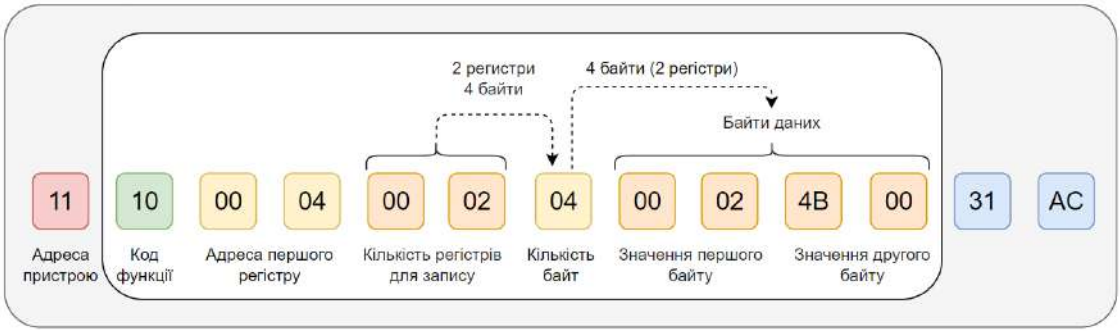


Рисунок 4.15 – Приклад запита для зміни режиму роботи та швидкості передавання даних

На рис. 4.16 подано отриману відповідь від контролера. Тут можна бачити, що в контролері МС-1 було змінено два регістри, починаючи з адреси 40005.

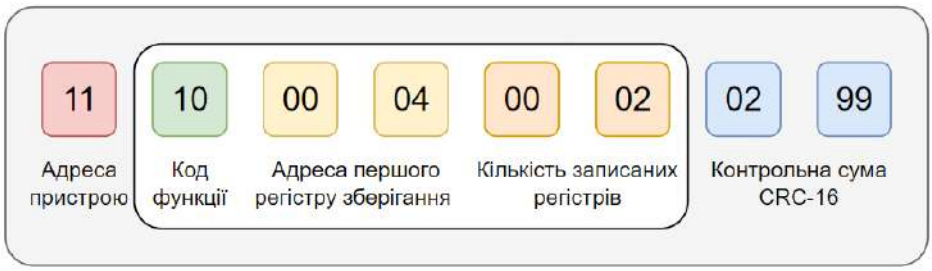


Рисунок 4.16 – Отримана відповідь від контролера

4.4.7 Зміна стану одного вихідного контакту (05)

Це повідомлення модифікує один логічний осередок, що являє собою один дискретний вихід контролеру. Осередки нумеруються з нуля (осередок 1 = 0, осередок 2 = 1 і т. д.).

Для увімкнення дискретного виходу необхідно передати число 0xFF00. Ця команда встановлює певний осередок в «1». Якщо необхідно вимкнути дискретний вихід, необхідно передати число 0x0000. Ця команда переведе певний осередок в стан логічного «0». Інші передані числа не впливають на вміст осередка, що адресується. Ця функція може використовуватися в широкомовному режимі. У таблиці 4.13 подано приклад установки в «1» дискретного виходу з адресою 0173 (0x00AD) в пристрій з адресою 17 (0x11). У відповідь повинне надійти повідомлення, яке повністю співпадає з запитом.

Таблиця 4.13 – Приклад встановлення в стан логічної «1» осередка 0173

Байт	Опис полів запиту
11	Адреса підлеглого пристрою
05	Код функції
00	Адреса дискретного виходу, Hi байт
AD	Адреса дискретного виходу, Lo байт
FF	Ознака встановлення (0xFF), або скидання (0x00)
00	Завжди містить значення 0x00
1F	Контрольна сума CRC
4B	Контрольна сума CRC

4.4.8 Зміна стану декількох вихідних контактів (0F)

Цей код функції використовується для примусового ввімкнення або вимкнення кожного вихідного контакту (котушки) в межах адресного простору

у віддаленому пристрої. PDU запита вказує посилання на певні контакти, які потрібно примусово встановити, або вимкнути. Контакти адресуються починаючи з нуля. Тому котушка під номером 1 адресується як 0.

Необхідні стани On/Off визначаються вмістом поля даних запита. Логічна «1» у бітовій позиції поля вимагає, щоб відповідний вихід був увімкнений. Логічний «0» вимагає, щоб його було вимкнено.

У таблиці 4.14 наведено приклад зміни стану 9 дискретних виходів з адресами починаючи з 00028 до 00036 для віддаленого пристрою з адресою 11 (0x0B).

Таблиця 4.14 – Приклад зміни стану 9 дискретних виходів

Байт	Опис полів запиту
0B	Адреса підлеглого пристрою
0F	Код функції
00	Початкова адреса дискретного виходу, Hi байт
1B	Початкова адреса дискретного виходу, Lo байт
00	Кількість виходів, стани яких потрібно змінити, Hi байт
09	Кількість виходів, стани яких потрібно змінити, Lo байт
02	Кількість байт даних
4D	Перший байт даних
01	Другий байт даних
6C	Контрольна сума CRC
A7	Контрольна сума CRC

Принцип побудови байт даних подано на рис. 4.17. Кількість байт даних визначається таким чином:

9 дискретних виходів = 1 байт + 1 біт + 7 порожніх біт = 2 байти.

Далі збираються байти даних. Дискретні виходи 35-28 мають наступну комбінацію: 0100 1101. Дискретний вихід 36 повинен бути увімкнений (логічна «1»). Після додавання семи порожніх біт, отримаємо комбінацію: 0000 0001. Таким чином, два байти даних будуть мати наступне значення: 4D 01.

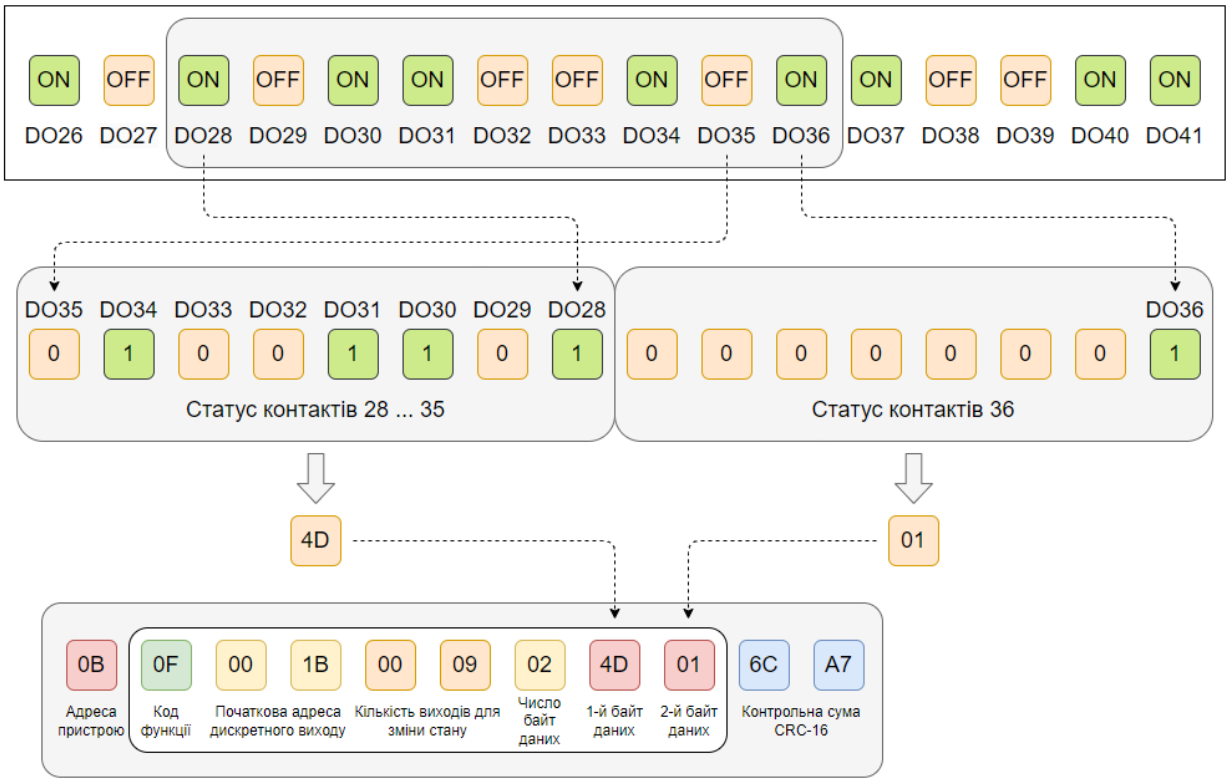


Рисунок 4.17 – Принцип побудови байт даних для формування повідомлення зміни декількох дискретних виходів

Як можна бачити з рис. 4.17, старші біти містять старші змінні вихідних контактів. З наведеного прикладу видно, що котушка 28 увімкнена (логічна «1»), а котушка 35 вимкнена (логічний «0»). Останнє поле даних містить статус лише одного дискретного виходу (логічна «1»). Невикористані біти в останньому байті даних заповнюються нулями – це тримачі простору.

Відповідь від підлеглого пристрою містить адресу розміщення першого дискретного виходу та підтвердження кількості записаних дискретних виходів (табл. 4.15).

4.4.9 Читання стану вхідних регістрів (04)

Вхідні регістри в промислових контролерах зазвичай використовуються для запису даних, що отримуються від аналого-цифрових перетворювачів. Це можуть біти значення виміряного струму, напруги або, наприклад, температури. Вхідні регістри 16-розрядні. Якщо значення параметру, що вимірюється, перевищують 16 біт (наприклад, АЦП розрядністю 24 біти), то в пам’яті ПЛК вони займатимуть два регістри.

Таблиця 4.15 – Відповідь на запит щодо зміни стану декількох вихідних контактів

Байт	Опис полів відповіді
0B	Адреса підлеглого пристрою
0F	Код функції
00	Адреса першого байта, Hi байт
1B	Адреса другого байта, Lo байт
00	Кількість записаних дискретних виходів, Hi байт
1B	Кількість записаних дискретних виходів, Lo байт
E5	Контрольна сума CRC
60	Контрольна сума CRC

В протоколі Modbus для читання вхідних регістрів використовується функція 0x04. Вона використовується для безперервного читання від 1 до 125 вхідних регістрів з віддаленого пристрою. PDU запита вказує початкову адресу регістра та кількість регістрів. В PDU адресація регістрів починається з нуля. Тому вхідний регістр під номером «1» адресується як «0».

У таблиці 4.16 поданий приклад запита на читання значення АЦП з вхідного регістра 30011 з пристрою з адресою 17 (0x11).

Таблиця 4.16 – Приклад запиту на читання значення АЦП

Байт	Опис полів запиту
11	Адреса підлеглого пристрою
04	Код функції
00	Адреса першого байта вхідного регістра, Hi байт
0A	Адреса першого байта вхідного регістра, Lo байт
00	Число регістрів для читання, Hi байт
01	Число регістрів для читання, Lo байт
13	Контрольна сума CRC
58	Контрольна сума CRC

Пристрій, що адресується посилає у відповіді свою адресу, код виконаної функції і інформаційне поле.

Для даного прикладу інформаційне поле містить 2 байти, які описують кількість байт даних, що повертаються. Довжина кожного регістра даних – 2 байти. Перший байт даних в посилці є старшим байтом регістру, другий – молодшим.

У таблиці 4.17 поданий приклад відповіді на запит читання вхідного регістру 30011, в якому міститься число 0x102F (4143), що у двійковому форматі має наступний вигляд – 0001000000101111.

Таблиця 4.17 – Приклад відповіді на функцію 04

Байт	Опис полів відповіді
11	Адреса підлеглого пристрою
04	Код функції
02	Кількість байт в полі даних
10	Значення першого байта даних, Hi байт
2F	Значення першого байта даних, Lo байт
34	Контрольна сума CRC
EF	Контрольна сума CRC

4.5 Огляд інструментів для аналізу мережевого трафіку на прикладі програмного засобу Wireshark

Wireshark – це широко поширений інструмент для захоплення та аналізу мережного трафіку, який активно використовується як для освітніх цілей, так і для усунення несправностей на комп’ютері або мережі. Wireshark працює практично з усіма протоколами моделі OSI, має зрозумілий для звичайного користувача інтерфейс і зручну систему фільтрації даних. Крім того, програма є кросплатформною і підтримує наступні операційні системи: Windows, Linux, Mac OS X, Solaris, FreeBSD, NetBSD, OpenBSD.

Wireshark дозволяє захоплювати і аналізувати трафік на всіх рівнях Інтернет-моделі TCP/IP (канальний, мережевий, транспортний і прикладний) для великої кількості протоколів (мультимедіа, мобільні мережі тощо). Це дозволяє локалізувати проблему: проблема з мережею, конфігурацією ПЗ чи помилка ПЗ.

Після запуску Wireshark в головному вікні програми буде список мережевих інтерфейсів (Ethernet, Wi-Fi, VPN мають теж окремий інтерфейс). Необхідно обрати потрібний інтерфейс, через який проходить трафік, що аналізується (рис. 4.18).

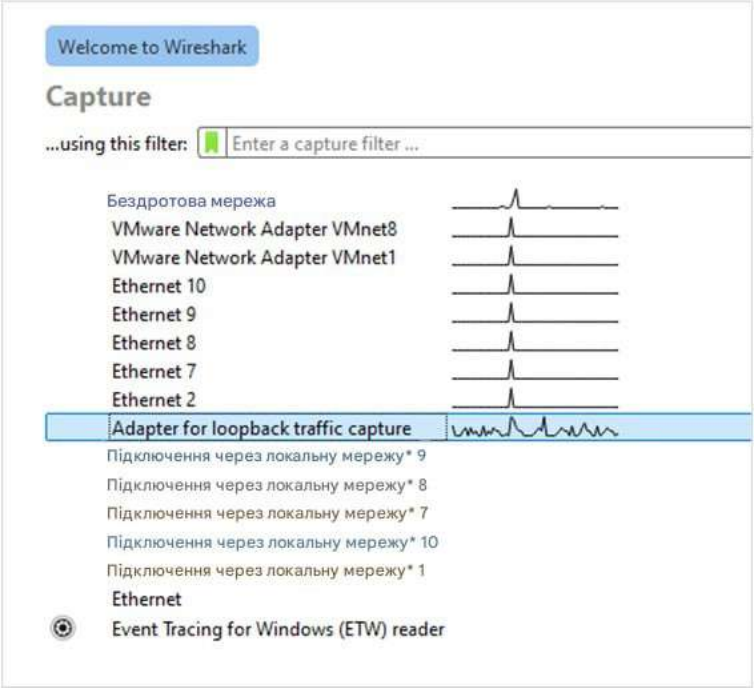


Рисунок 4.18 – Вибір мережевого інтерфейсу

Мережевий інтерфейс – це програмне забезпечення, яке взаємодіє з мережним драйвером та з рівнем IP [9]. Він забезпечує рівню IP доступ до всіх наявних мережних адаптерів, трафік яких ми будемо перехоплювати. Найчастіше у програмі Wireshark можна зустріти мережний інтерфейс бездротової (Wi-Fi) та кабельний (Ethernet). На рис. 4.19 подано приклад перехоплення загального мережевого трафіку.

На рис. 4.20 подано загальний інтерфейс програми Wireshark. Панель фільтрів (1) дозволяє знайти необхідну інформацію. Панель найменувань (2), що поділяє інформацію з отриманого трафіку на номер, час від початку захоплення трафіку, джерело та адресат, а також протокол, розмір пакета і невелику інформацію про мережевий пакет. Панель пакетів (3) оновлюється в реальному часі. Тут інформація про пакети розділена на стовпці, визначені на панелі найменувань. Панель рівнів (4), що описує рівні моделі OSI вибраного мережного пакета. Панель метаданих (5), що представляє дані у шістнадцятковому коді та символах.

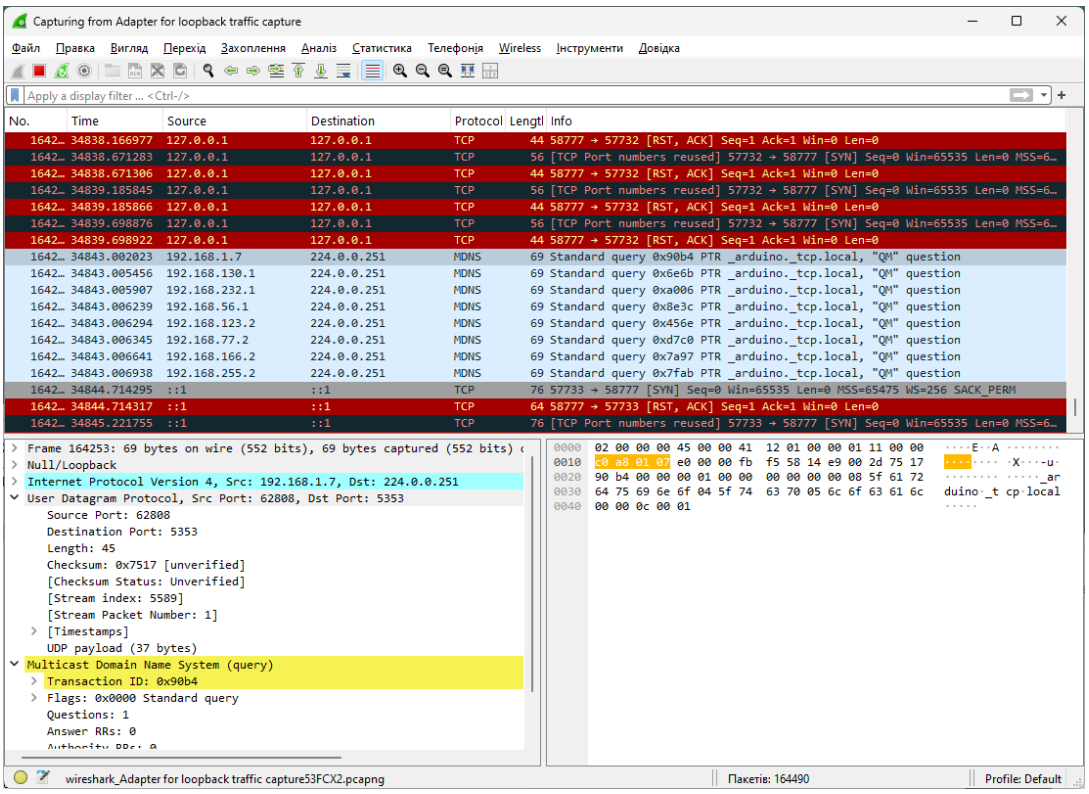


Рисунок 4.19 – Приклад перехоплення загального мережевого трафіку

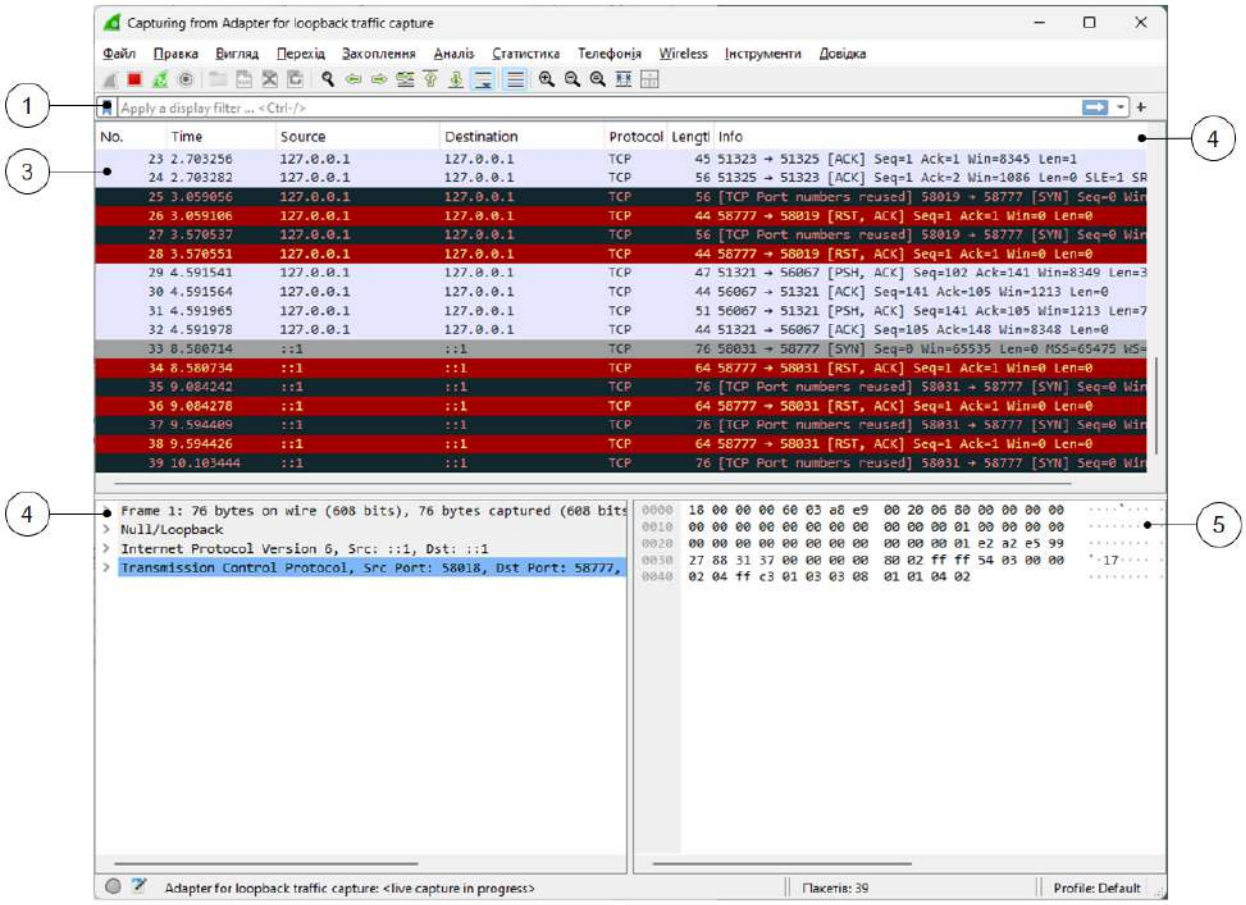


Рисунок 4.20 – Загальний інтерфейс програми

У спеціальному полі Filter можна ввести необхідні команди або скористатися підказками (рис. 4.21).

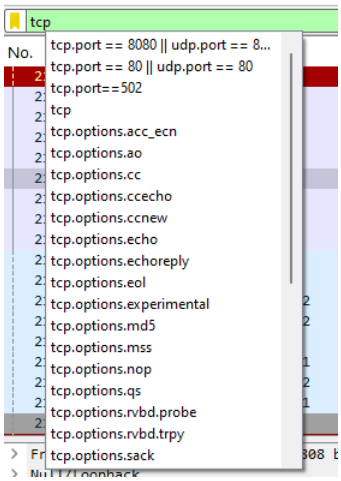


Рисунок 4.21 – Варіанти підказок при заповненні поля Filter

Найчастіше використовується фільтрація за IP-адресами, за номерами порту та протоколами (рис. 4.22).

Фільтрування за IP-адресою дозволяє нам переглядати всі пакети, що надходять від кого-небудь або ті, що йдуть будь-кому. Наприклад, відберемо всі пакети, що надходять від IP-адреси 127.0.0.1 за допомогою введення у фільтрі «ip.src == x.x.x.x».

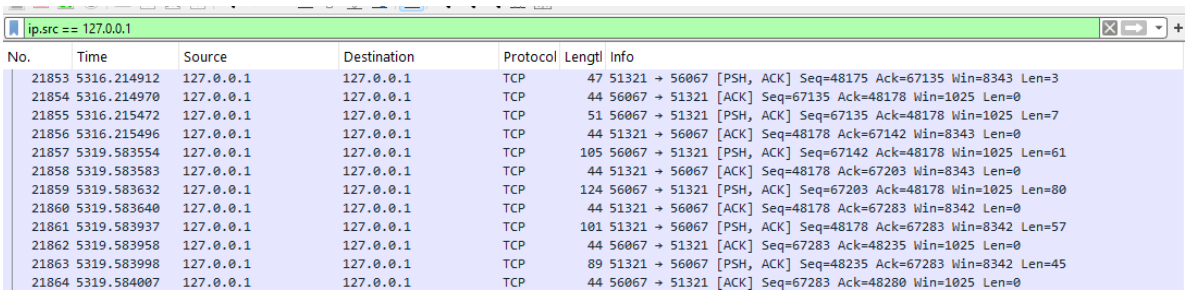


Рисунок 4.22 – Фільтрація пакетів, що надходять від IP-адреси 127.0.0.1

Також можна відфільтрувати трафік мережі IP-адресою одержувача пакетів за допомогою команди «ip.dst == x.x.x.x» (рис. 4.23).

ip.dst == 127.0.0.1

No.	Time	Source	Destination	Protocol	Lengtl	Info
1616…	34234.586444	127.0.0.1	127.0.0.1	WebSoc…	105	WebSocket Text [FIN] [MASKED]
1616…	34234.586465	127.0.0.1	127.0.0.1	TCP	44	51321 → 53650 [ACK] Seq=61084 Ack=85734 Win=2140928 Len=0
1616…	34234.586509	127.0.0.1	127.0.0.1	WebSoc…	124	WebSocket Binary [FIN] [MASKED]
1616…	34234.586517	127.0.0.1	127.0.0.1	TCP	44	51321 → 53650 [ACK] Seq=61084 Ack=85814 Win=2140928 Len=0
1616…	34234.586591	127.0.0.1	127.0.0.1	WebSoc…	51	WebSocket Text [FIN] [MASKED]
1616…	34234.586607	127.0.0.1	127.0.0.1	TCP	44	51321 → 53650 [ACK] Seq=61084 Ack=85821 Win=2140928 Len=0
1616…	34234.586767	127.0.0.1	127.0.0.1	WebSoc…	101	WebSocket Text [FIN]
1616…	34234.586784	127.0.0.1	127.0.0.1	TCP	44	53650 → 51321 [ACK] Seq=85821 Ack=61141 Win=266240 Len=0
1616…	34234.586817	127.0.0.1	127.0.0.1	WebSoc…	89	WebSocket Binary [FIN]
1616…	34234.586825	127.0.0.1	127.0.0.1	TCP	44	53650 → 51321 [ACK] Seq=85821 Ack=61186 Win=266240 Len=0
1616…	34234.808811	127.0.0.1	127.0.0.1	TCP	45	61471 → 61472 [PSH, ACK] Seq=1396 Ack=1 Win=2161152 Len=1
1616…	34234.808854	127.0.0.1	127.0.0.1	TCP	44	61472 → 61471 [ACK] Seq=1 Ack=1397 Win=325888 Len=0
1616…	34234.946021	127.0.0.1	127.0.0.1	TCP	45	[TCP Keep-Alive] 51323 → 51325 [ACK] Seq=69 Ack=69 Win=8345 Len=1
1616…	34234.946055	127.0.0.1	127.0.0.1	TCP	56	[TCP Keep-Alive ACK] 51325 → 51323 [ACK] Seq=69 Ack=70 Win=1086 Len=0 SLE…

Рисунок 4.23 – Фільтрація пакетів, що адресовані IP-адресі 127.0.0.1

Для фільтрації за номером порту використовується .port = x після назви протоколу. Наприклад, для перегляду TCP-порту 502, який використовується для обміну повідомлення за допомогою протоколу Modbus TCP, використовуємо команду «tcp.port==502» (рис. 4.24).

tcp.port==502

No.	Time	Source	Destination	Protocol	Lengtl	Info
3655	1009.606316	127.0.0.1	127.0.0.1	TCP	44	58445 → 502 [ACK] Seq=37 Ack=31 Win=327424 Len=0
3656	1009.606563	127.0.0.1	127.0.0.1	Modbus…	56	Query: Trans: 4; Unit: 1, Func: 1: Read Coils
3657	1009.606592	127.0.0.1	127.0.0.1	TCP	44	502 → 58445 [ACK] Seq=31 Ack=49 Win=2161152 Len=0
3658	1009.621077	127.0.0.1	127.0.0.1	Modbus…	54	Response: Trans: 4; Unit: 1, Func: 1: Read Coils
3659	1009.621126	127.0.0.1	127.0.0.1	TCP	44	58445 → 502 [ACK] Seq=49 Ack=41 Win=327424 Len=0
3660	1009.631899	127.0.0.1	127.0.0.1	TCP	44	58445 → 502 [FIN, ACK] Seq=49 Ack=41 Win=327424 Len=0
3661	1009.631923	127.0.0.1	127.0.0.1	TCP	44	502 → 58445 [ACK] Seq=41 Ack=50 Win=2161152 Len=0
3662	1009.635922	127.0.0.1	127.0.0.1	TCP	44	502 → 58445 [FIN, ACK] Seq=41 Ack=50 Win=2161152 Len=0
3663	1009.635948	127.0.0.1	127.0.0.1	TCP	44	58445 → 502 [ACK] Seq=50 Ack=42 Win=327424 Len=0
1624…	34414.399100	127.0.0.1	127.0.0.1	TCP	56	57524 → 502 [SYN] Seq=0 Win=65535 Len=0 MSS=65495 WS=256 SACK_PERM
1624…	34414.399144	127.0.0.1	127.0.0.1	TCP	56	502 → 57524 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=65495 WS=256 SACK_…
1624…	34414.399172	127.0.0.1	127.0.0.1	TCP	44	57524 → 502 [ACK] Seq=1 Ack=1 Win=2161152 Len=0
1624…	34414.424172	127.0.0.1	127.0.0.1	Modbus…	56	Query: Trans: 1; Unit: 1, Func: 1: Read Coils
1624…	34414.424190	127.0.0.1	127.0.0.1	TCP	44	502 → 57524 [ACK] Seq=1 Ack=13 Win=2161152 Len=0
1624…	34414.440190	127.0.0.1	127.0.0.1	Modbus…	54	Response: Trans: 1; Unit: 1, Func: 1: Read Coils
1624…	34414.440210	127.0.0.1	127.0.0.1	TCP	44	57524 → 502 [ACK] Seq=13 Ack=11 Win=2161152 Len=0
1624…	34414.451240	127.0.0.1	127.0.0.1	Modbus…	56	Query: Trans: 2; Unit: 1, Func: 1: Read Coils
1624…	34414.451258	127.0.0.1	127.0.0.1	TCP	44	502 → 57524 [ACK] Seq=11 Ack=25 Win=2161152 Len=0

Рисунок 4.24 – Фільтрація пакетів за номером порту 502

4.6 Аналіз протоколу Modbus

4.6.1 Підготовка до проведення експерименту

Виконаємо аналіз протоколу Modbus за допомогою програмного засобу Wireshark. Для проведення експерименту зберемо лабораторний стенд, що буде включати наступні компоненти (рис. 4.25):

– аналізатор мережевого трафіку Wireshark (1);

– симулятор підлеглого пристрою Modbus Simulator (2);

– ModbusMaster (3) – програмний інструмент для генерації запитів до підлеглого пристрою.

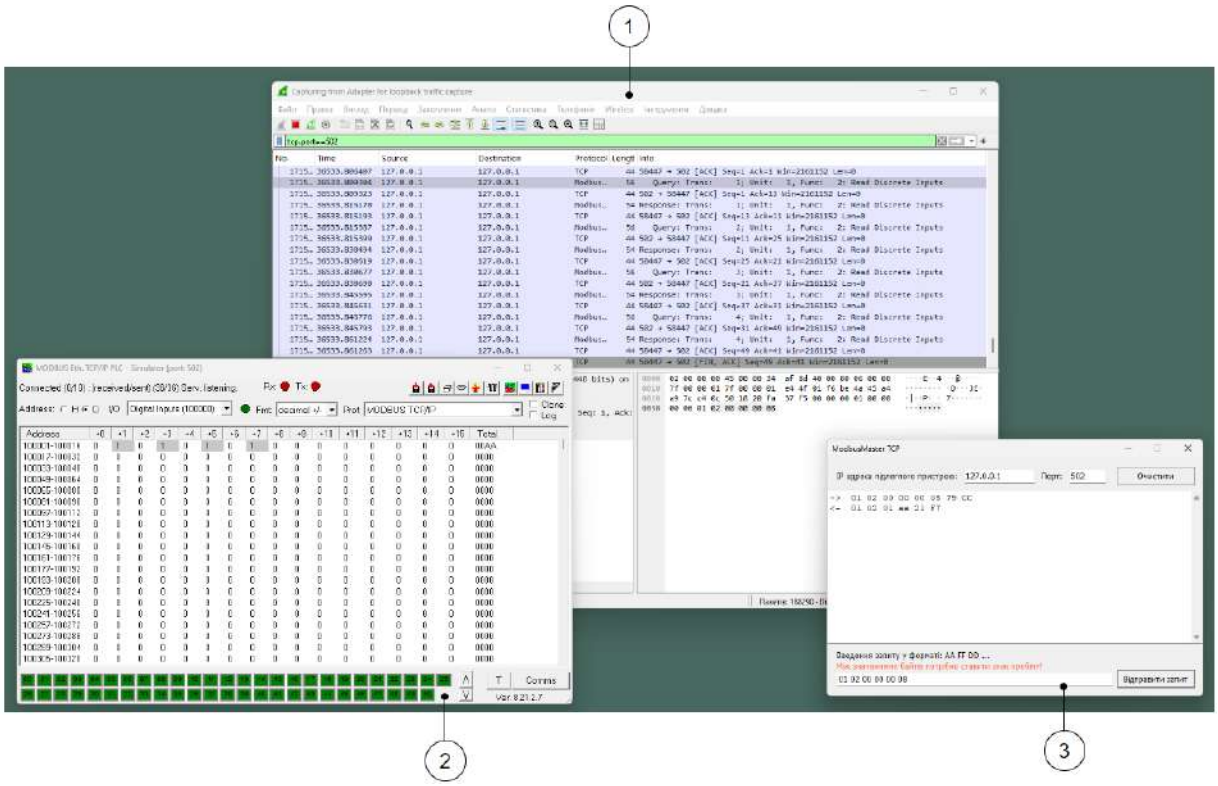


Рисунок 4.25 – Лабораторний стенд для дослідження методів аналізу мережевого трафіку

4.6.2 Програма ModRSsim2

Програма ModRSsim2 – це симулятор протоколу Modbus для RS-232 та TCP/IP. Програма підтримує наступні реалізації протоколу:

– симуляція підлеглого пристрою з доступом за протоколом Modbus TCP/IP;

– симуляція підлеглого пристрою з доступом за протоколом Modbus RTU.

Середовище надає можливості тестування з використанням різних сценаріїв автоматизації.

Після завантаження програми ModRSsim2 необхідно провести початкові налаштування для роботи з протоколом Modbus. Для цього потрібно обрати відповідний тип протоколу в правому верхньому меню (рис. 4.26).

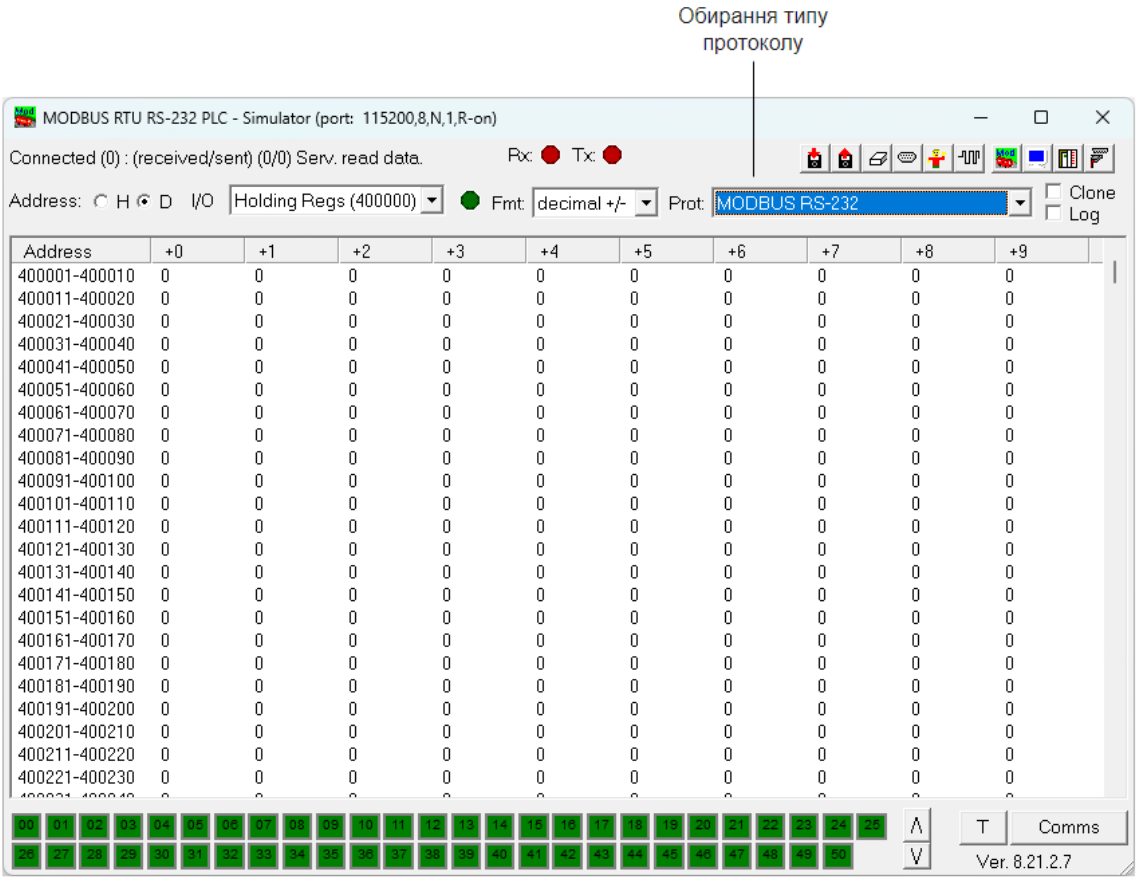


Рисунок 4.26 – Обирання типу протоколу

В меню, що випадає, можна обрати один з варіантів:

– Modbus RS232;

– Modbus TCP/IP;

– Allen Bradley DF1;

– Joy SCC DF1.

Приклад меню подано на рис. 4.27.

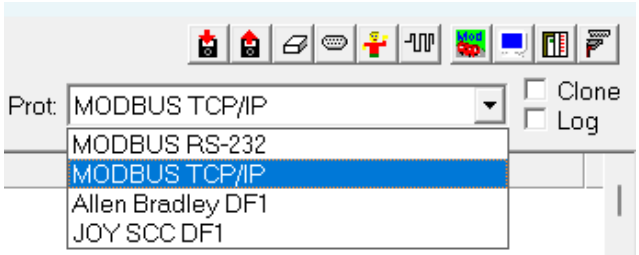


Рисунок 4.27 – Меню обирання типу протоколу

У випадку, якщо користувач вперше обирає протокол Modbus TCP/IP, система запропонує вікно налаштування мережевого захисника Windows (рис. 4.28).

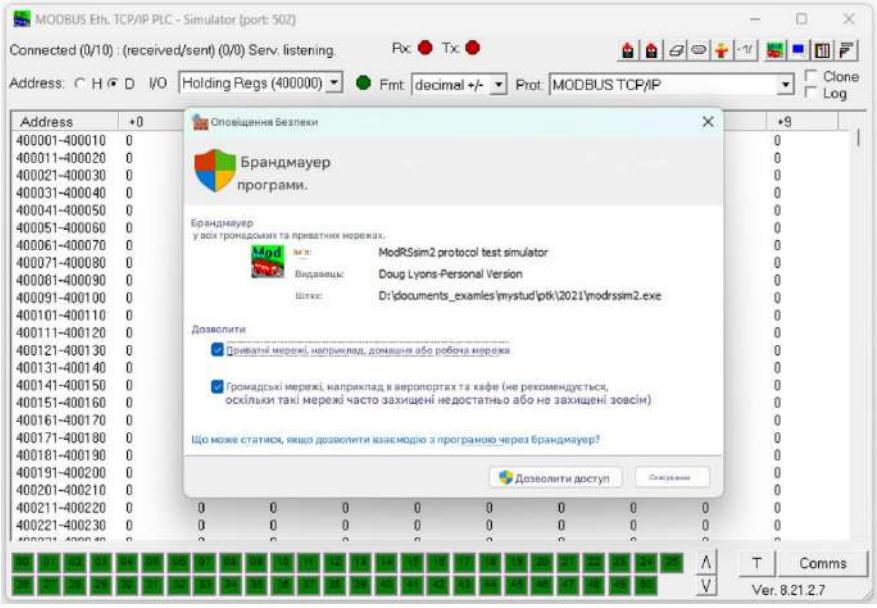


Рисунок 4.28 – Надання дозволу для роботи програми в робочій мережі

Після завершення налаштування мережевого захисника необхідно обрати параметри з’єднання. Для цього в програмі передбачена відповідна кнопка (рис. 4.29)

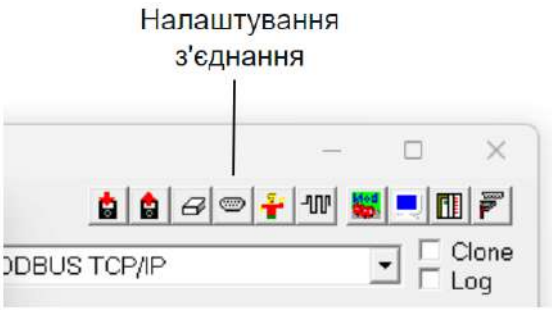


Рисунок 4.29 – Налаштування з’єднання

В параметрах налаштування залежать від типу обраного протоколу. Якщо обрано TCP/IP, то в параметрах налаштування з’єднання необхідно перевірити

правильність обирання порту, через який відбуватиметься передавання даних (рис. 4.30).

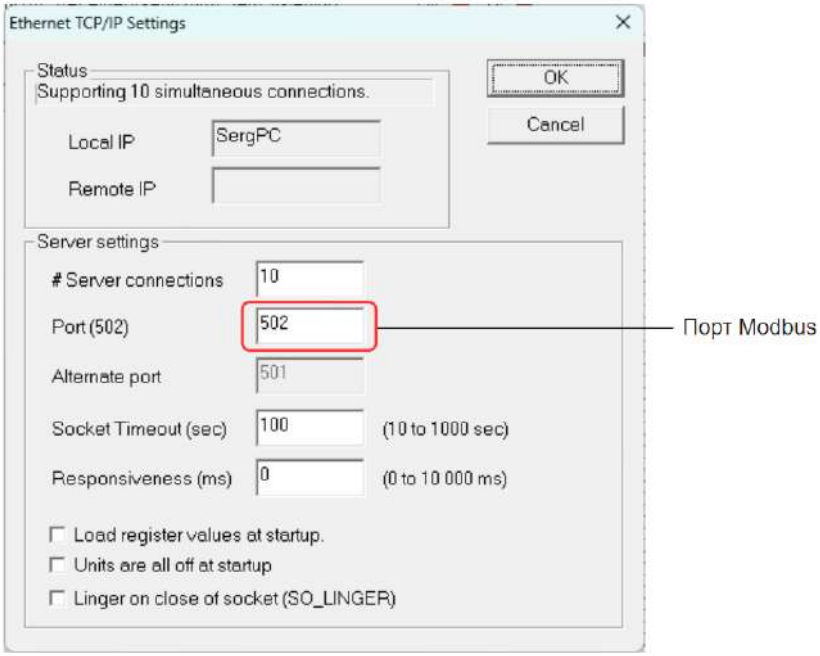


Рисунок 4.30 – Порт передавання даних для Modbus TCP/IP

Також потрібно перевірити IP-адресу яку прослуховує програма ModRSsim2. Зазвичай на локальному ПК обирається адреса 127.0.0.1 (localhost).

У разі, якщо обрано тип протоколу Modbus RS232 вікно налаштування буде мати наступний вигляд (рис. 4.31).

В даному вікні налаштовуються:

– COM-порт для доступу до мережі Modbus;

– швидкість передавання даних;

– тип контролю правильності передавання даних;

– кількість бід даних в повідомленні (7 або 8);

– кількість стопових біт (1, 1.5, 2);

– тип управління потоком даних.

Також можна задати час очікування відповіді від підлеглого пристрою, якщо використовується нестабільна мережа, або виникли проблеми зі зв’язком.

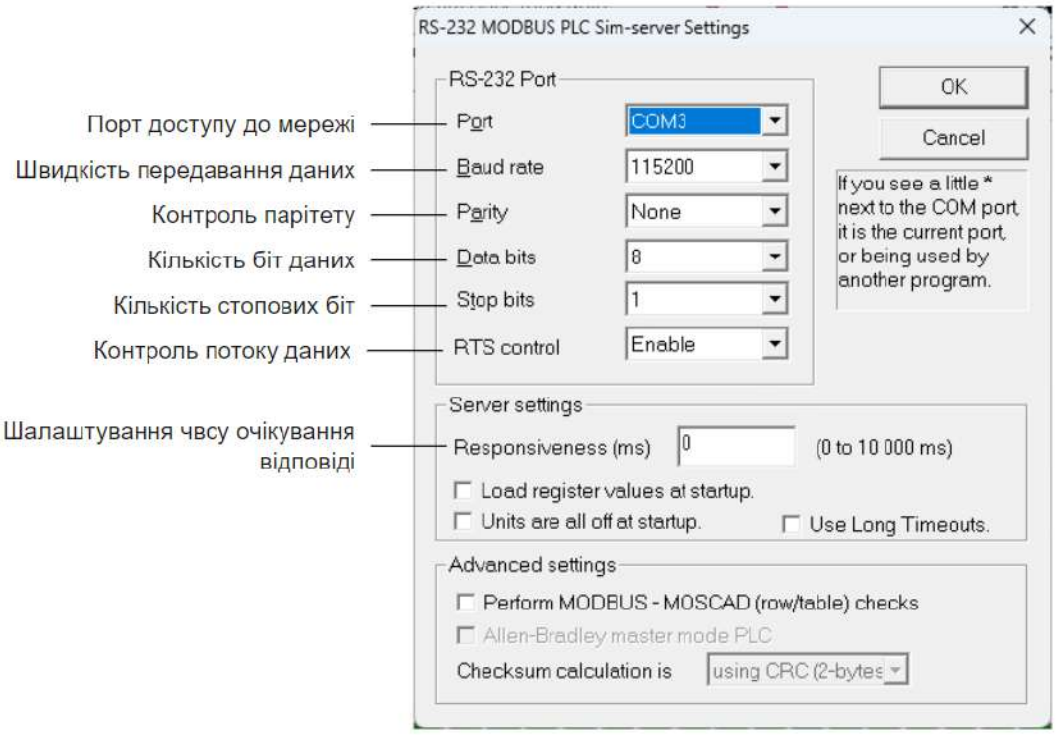


Рисунок 4.31 – Вікно налаштування параметрів з’єднання за протоколом Modbus RTU

Управління підключенням відбувається за допомогою відповідної кнопки (рис. 4.32).

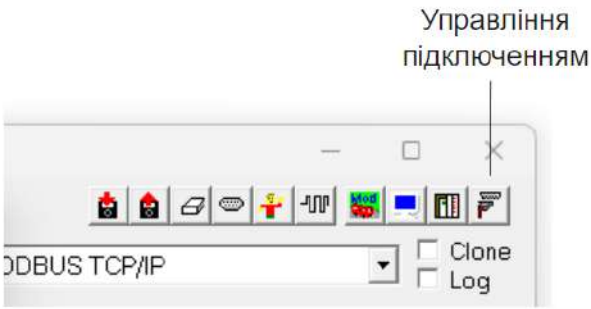
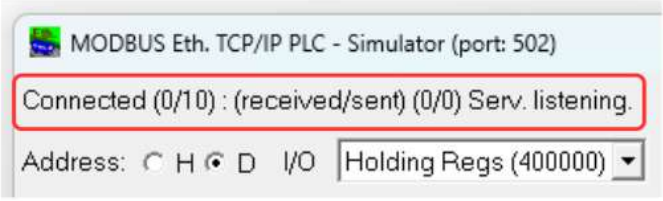
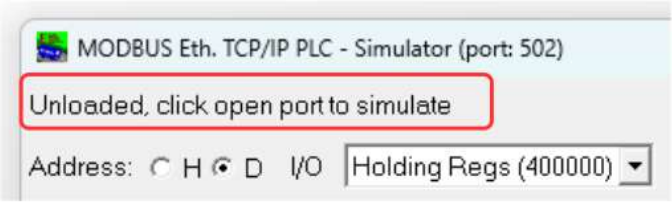


Рисунок 4.32 – Управління підключенням

Контролювати стан підключення можна за допомогою системного повідомлення, що виводиться на екран програми (рис. 4.33).



а)



б)

Рисунок 4.33 – Контроль стану підключення:
а) з’єднання встановлено; б) відсутність з’єднання

4.6.3 Програмний симулятор «Майстер мережі Modbus TCP/IP»

Для налагодження пристроїв, що працюють під управлінням протоколу Modbus TCP/IP, а також для виконання лабораторних робіт присвячених вивченню мережевих технологій, розроблено спеціальний програмний термінал [7, 8, 10]. Інтерфейс програми подано на рис. 4.34.

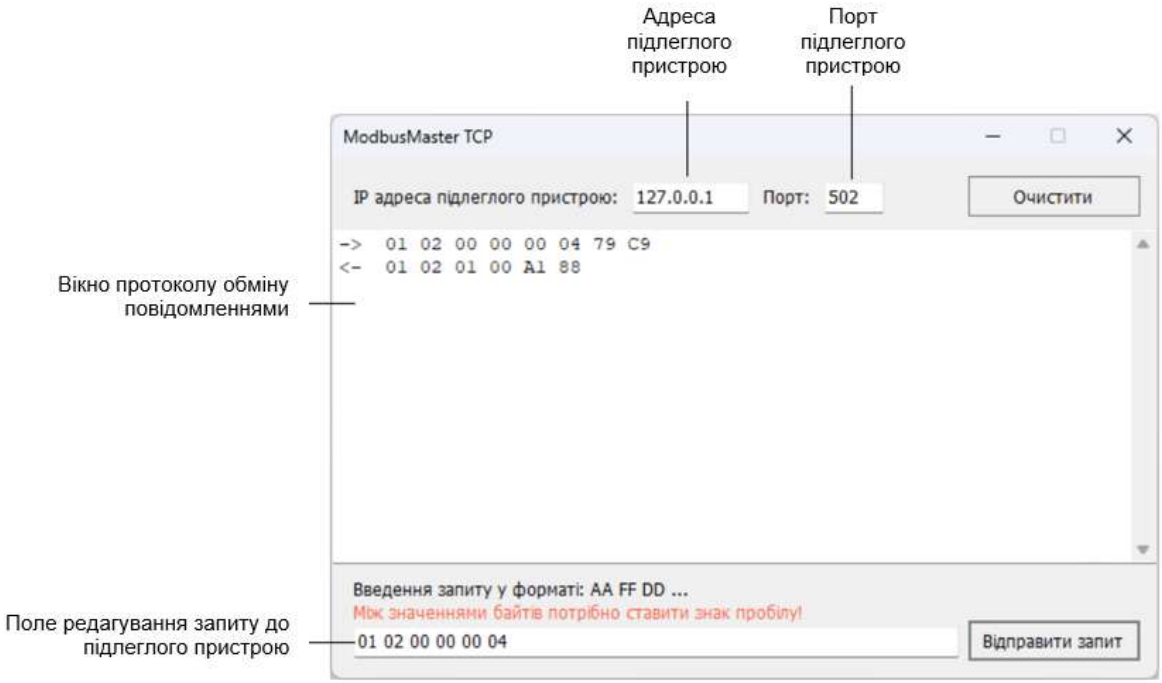


Рисунок 4.34 – Інтерфейс програми «Майстер мережі Modbus TCP/IP»

У верхній частині програми виконується налаштування підключення до мережі Ethernet. Тут можна ввести адресу підлеглого пристрою, до я кого будуть надсилатися запити, та вказати номер порту на підлеглому пристрою, через який відбувається прослуховування мережі.

За замовчення адреса підлеглого пристрою 127.0.0.1 (localhost), а порт доступу 502.

В нижній частині програми знаходиться поле введення запиту до підлеглого пристрою. Запит вводиться в HEX-форматі. Між байтами в запиті потрібно ставити пробіл. Наприклад, запит до підлеглого пристрою для читання дискретних входів буде мати такий вигляд:

01 02 00 00 00 04

де 01 – адреса пристрою, але для протоколу Modbus TCP/IP вона ігнорується;

02 – номер функції;

00 00 – адреса першого дискретного входу;

00 04 – кількість входів для читання.

Контрольну суму CRC вводити не потрібно. Програма сам підраховує ці два байти та автоматично додає до запиту.

В середній частині інтерфейсу програми розташовано вікно для відображення історії переданих та отриманих пакетів. Переданим пакетам передує префікс «→», а прийнятим – «←» (рис. 4.35). Кнопка «Очистити» призначена для очищення історії повідомлень.

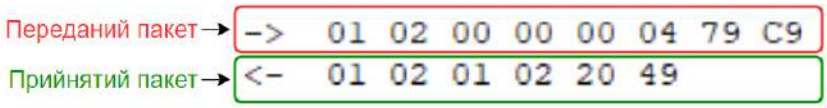


Рисунок 4.35 – Позначення переданих та прийнятих пакетів

4.6.4 Проведення експерименту з аналізу пакетів протоколу Modbus

Для аналізу пакетів протоколу Modbus в симуляторі ModRSsim2 виконаємо вибір адресного простору «Digital Input (100000)» (рис. 4.36).

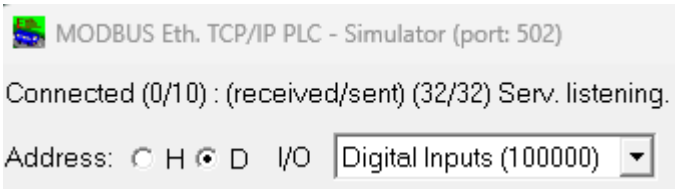


Рисунок 4.36 – Вибір адресного простору «Digital Input (100000)»

За допомогою вказівника миші виставимо наступну комбінацію на вході симулятора «01010101» (рис. 4.37).

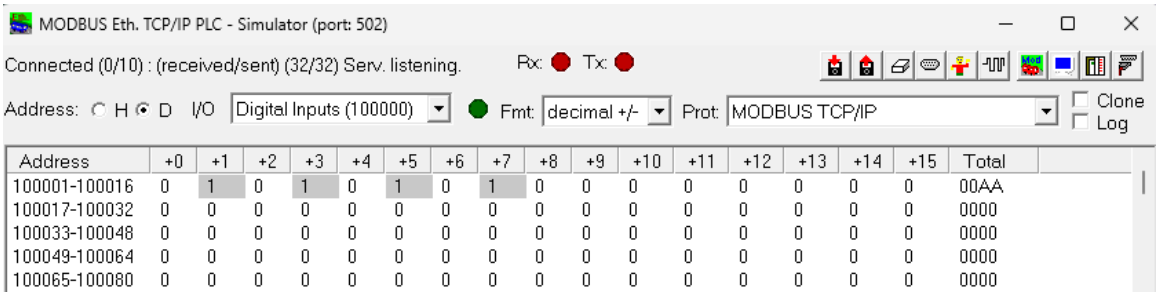


Рисунок 4.37 – Зміна стану вхідних контактів в симуляторі

Для читання стану вхідних контактів використовується функція «02». Щоб прочитати стан восьми вхідних контактів, починаючи з нульової адреси сформую наступний рядок байтів:

01 02 00 00 00 08

Значення обраних байтів описано в таблиці 4.18.

Таблиця 4.18 – Опис байтів тестового запиту читання стану вхідних контактів

Байт	Опис полів запита
01	Адреса підлеглого пристрою
02	Код функції
00	Адреса початкового біта, Hi байт
00	Адреса початкового біта, Lo байт
00	Кількість дискретних входів, Hi байт
08	Кількість дискретних входів, Lo байт

Перед виконанням запиту потрібно налаштувати програму Wireshark на перехоплення пакетів, що передаються через порт 502, який є стандартним для протоколу Modbus.

В програмі ModbusMaster TCP вводимо набір байтів та натискаємо кнопку «Відправити запит». Результат виконання запиту подано на рис. 4.38.

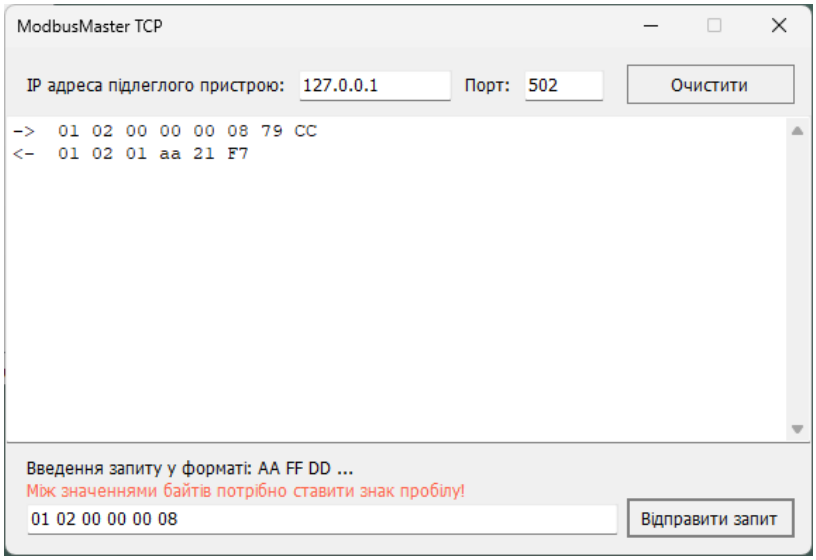
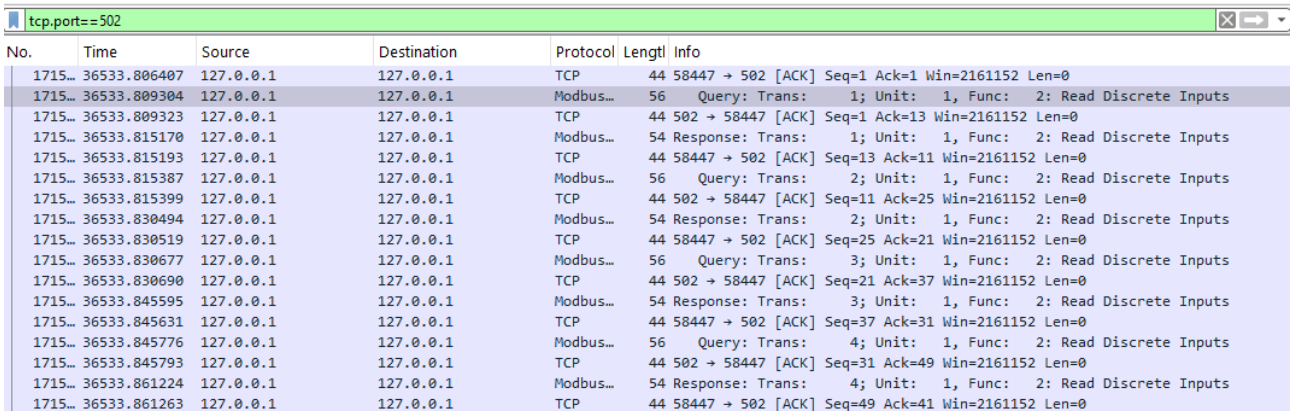


Рисунок 4.38 – Результат виконання запиту читання стану вхідних контактів

За допомогою програми Wireshark виконаємо аналіз трафіку через порт 502 (рис. 4.39).



tcp.port==502

No.	Time	Source	Destination	Protocol	Lengtl	Info
1715…	36533.806407	127.0.0.1	127.0.0.1	TCP	44	58447 → 502 [ACK] Seq=1 Ack=1 Win=2161152 Len=0
1715…	36533.809304	127.0.0.1	127.0.0.1	Modbus…	56	Query: Trans: 1; Unit: 1, Func: 2: Read Discrete Inputs
1715…	36533.809323	127.0.0.1	127.0.0.1	TCP	44	502 → 58447 [ACK] Seq=1 Ack=13 Win=2161152 Len=0
1715…	36533.815170	127.0.0.1	127.0.0.1	Modbus…	54	Response: Trans: 1; Unit: 1, Func: 2: Read Discrete Inputs
1715…	36533.815193	127.0.0.1	127.0.0.1	TCP	44	58447 → 502 [ACK] Seq=13 Ack=11 Win=2161152 Len=0
1715…	36533.815387	127.0.0.1	127.0.0.1	Modbus…	56	Query: Trans: 2; Unit: 1, Func: 2: Read Discrete Inputs
1715…	36533.815399	127.0.0.1	127.0.0.1	TCP	44	502 → 58447 [ACK] Seq=11 Ack=25 Win=2161152 Len=0
1715…	36533.830494	127.0.0.1	127.0.0.1	Modbus…	54	Response: Trans: 2; Unit: 1, Func: 2: Read Discrete Inputs
1715…	36533.830519	127.0.0.1	127.0.0.1	TCP	44	58447 → 502 [ACK] Seq=25 Ack=21 Win=2161152 Len=0
1715…	36533.830677	127.0.0.1	127.0.0.1	Modbus…	56	Query: Trans: 3; Unit: 1, Func: 2: Read Discrete Inputs
1715…	36533.830690	127.0.0.1	127.0.0.1	TCP	44	502 → 58447 [ACK] Seq=21 Ack=37 Win=2161152 Len=0
1715…	36533.845595	127.0.0.1	127.0.0.1	Modbus…	54	Response: Trans: 3; Unit: 1, Func: 2: Read Discrete Inputs
1715…	36533.845631	127.0.0.1	127.0.0.1	TCP	44	58447 → 502 [ACK] Seq=37 Ack=31 Win=2161152 Len=0
1715…	36533.845776	127.0.0.1	127.0.0.1	Modbus…	56	Query: Trans: 4; Unit: 1, Func: 2: Read Discrete Inputs
1715…	36533.845793	127.0.0.1	127.0.0.1	TCP	44	502 → 58447 [ACK] Seq=31 Ack=49 Win=2161152 Len=0
1715…	36533.861224	127.0.0.1	127.0.0.1	Modbus…	54	Response: Trans: 4; Unit: 1, Func: 2: Read Discrete Inputs
1715…	36533.861263	127.0.0.1	127.0.0.1	TCP	44	58447 → 502 [ACK] Seq=49 Ack=41 Win=2161152 Len=0

Рисунок 4.39 – Перехоплений трафік через порт 502

В перехопленому трафіку можна виділити пакети, що були направлені від майстра до Modbus симулятора (область 1 на рис. 4.39) та ті, що послані у відповідь від симулятора до ModbusMaster TCP (область 2 на рис. 4.40).

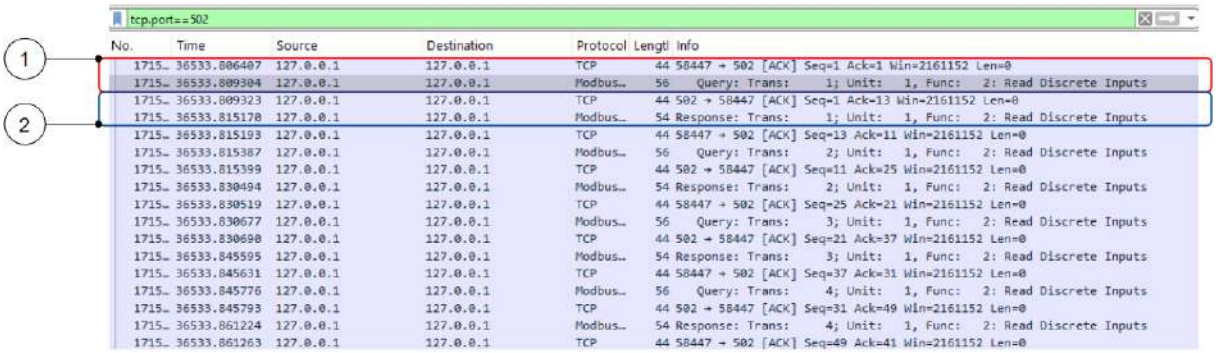


Рисунок 4.40 – Виділені пакети обміну повідомленнями між майстром та підлеглим пристроями

Для докладного аналізу пакетів спершу натиснемо на запит від майстра до підлеглого з позначкою «Modbus» в головному вікні та відкриємо панель рівнів моделі OSI (рис. 4.41).

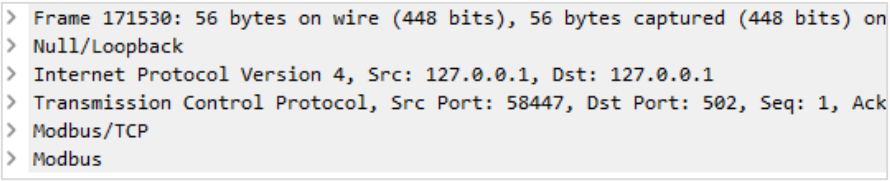


Рисунок 4.41 – Вміст панелі рівнів моделі OSI

У мережевому пакеті рівень Ethernet показує фізичні адреси джерела та призначення пакету. Фізичною адресою пристрою є його MAC-адреса (MediaAccess Control), яка записана на платі пристрою мережевого інтерфейсу (NIC – Network Interface Card). На основі цієї MAC-адреси комутатори приймають рішення, куди пересилати кадри Ethernet. MAC-адреса зазвичай не змінюється для пристрою і її не можна маршрутизувати, що означає, що вона не має сенсу за межами локальної мережі.

Наступний рівень, рівень Інтернет-протоколу, показує логічні адреси джерела та призначення пакета за допомогою IP-адрес. IP-адреса призначається пристрою вручну або отримується через DHCP чи BOOTP. У локальній мережі IP-адресу потрібно буде перетворити на MAC-адресу, перш ніж її можна буде пересилати комутатором. Це перетворення виконується за допомогою протоколу під назвою Address Resolution Protocol або ARP. Якщо пристрою необхідно

визначити фізичну адресу вузла за відомою IP-адресою, використовується протокол ARP. У цьому випадку пристрій надсилає широкомовний ARP-запит, який отримують усі вузли локальної мережі. У запиті міститься повідомлення з проханням визначити, який пристрій має вказану IP-адресу, та повідомити відповідну MAC-адресу. Пристрій, якому належить зазначена IP-адреса, надсилає відповідь із власною фізичною адресою відправнику запиту.

Відповідь на цей запит буде збережено в таблиці ARP пристрою, яка є засобом тимчасового запам'ятовування зв'язку між IP-адресою та MAC-адресою, тому запит не потрібно буде повторювати для кожного пакета.

Якщо IP-адреса призначення знаходиться поза межами локальної підмережі, пристрій передає пакет на шлюз за замовчуванням, яким зазвичай є маршрутизатор цієї мережі, за умови що він налаштований. Маршрутизатори здійснюють прийняття рішень щодо пересилання пакетів на основі IP-адрес, тому використання IP-адресації забезпечує можливість маршрутизації пакетів між різними підмережами та мережами. Завдяки цьому клієнт і сервер можуть знаходитися як у різних підмережах однієї мережі, так і в географічно віддалених мережах.

Технологія Ethernet разом із протоколом Інтернет (IP) забезпечує доставку пакета до потрібного вузла мережі, тоді як протокол керування передачею (TCP) відповідає за встановлення та підтримку з'єднання між клієнтською та серверною програмами.

У протоколі Modbus TCP зазвичай використовується порт призначення TCP 502, який є стандартним портом сервера Modbus. Номер порту джерела вибирається випадковим чином клієнтською стороною та використовується протоколом TCP разом з IP-адресами і номерами портів для ідентифікації та відстеження активних сеансів зв'язку.

На рис. 4.42 подано основні параметри протоколу, що аналізується.

В результаті аналізу можна виділити наступні характеристики:

– тип протоколу (1) – TCP;

– IP адреса пристрою з якого надіслано повідомлення (2) – 127.0.0.1;

– IP адреса пристрою до якого надійшло повідомлення (3) – 127.0.0.1;

– порт джерела сигналу (4) – 58447;

– порт призначення TCP (5) – 502.

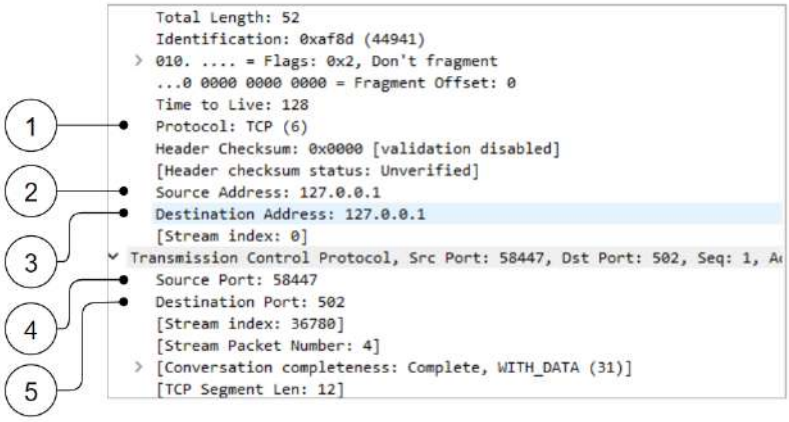


Рисунок 4.42 – Основні параметри протоколу, що аналізується

Далі пакет показує рівень Modbus/TCP. Це ADU, який Modbus використовує для зв’язку між пристроями. Wireshark добре виконує аналіз окремих полів протоколу Modbus. На рис. 4.43 можна побачити чотири поля заголовка MBAP в ADU, що заповнені відповідно рис. 4.8:

– ідентифікатор транзакції (1): 1;

– ідентифікатор протоколу (2): 0;

– довжина залишку пакету (3): 6;

– ідентифікатор складової одиниці пакету даних (4): 1.

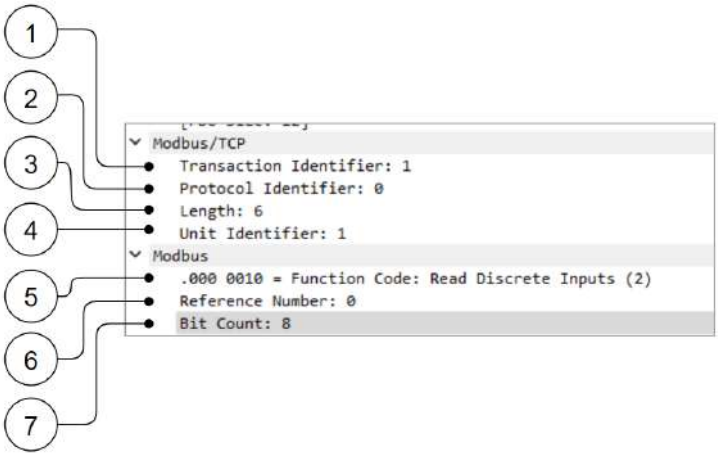


Рисунок 4.43 – Аналіз протоколу Modbus

Крім того також можна побачити поля для PDU кадру Modbus, що описують наступне:

– код функції: 2 (читання дискретних входів);

– номер зміщення відповідно початкової адреси: 0 (початок з дискретного входу з номером 0);

– кількість дискретних входів - 8 (читати 8 дискретних входів поспіль).

Натискаючи на відповідні поля протоколу в панелі рівнів OSI в панелі метаданих можна докладно подивитись на вміст пакету даних. Кольором буде підсвічено відповідні байти, що становлять обраний параметр протоколу Modbus (рис. 4.44).

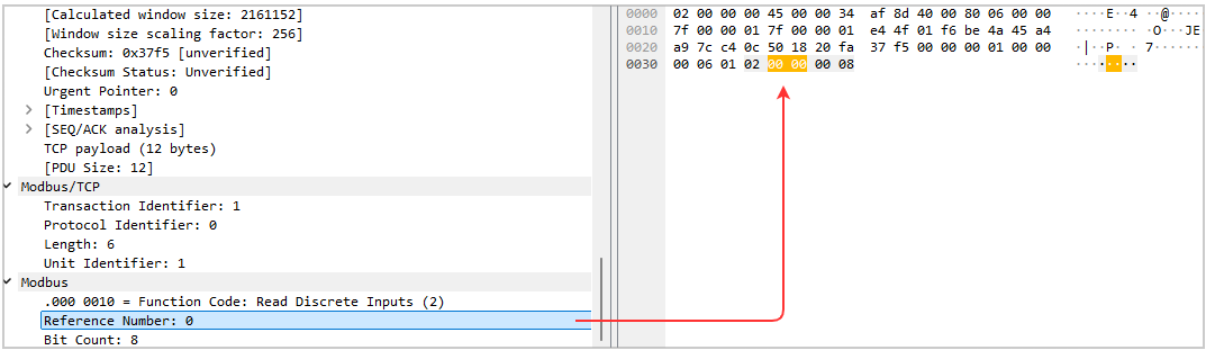


Рисунок 4.44 – Аналіз вмісту полів протоколу в режимі запиту

Реалізація частини даних PDU залежить від запитуваної функції. У цьому випадку за допомогою функції 2 (Read Descrete Inputs), частина даних PDU інтерпретується як окремі біти (рис. 4.45). Для функції 1, Read Coils, дані також будуть інтерпретуватися як біти, а для інших функцій частина даних PDU може містити додаткову інформацію разом, щоб підтримувати запит функції. Наприклад для функції 3, Read Holding Registers, частина даних PDU інтерпретується як слова (16-бітні цілі числа) (рис. 4.46).

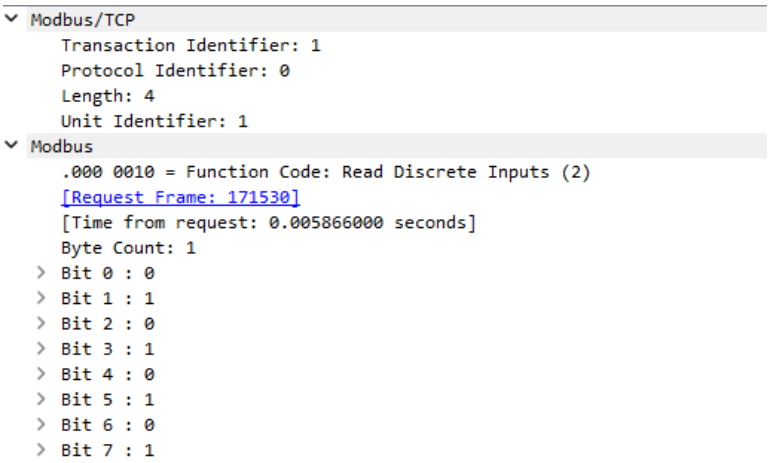


Рисунок 4.45 – Відображення частини даних PDU у відповіді на запит

У наступному фрагменті захопленого мережевого трафіку наведено відповідь сервера Modbus із запитуваними даними. Розкриваючи кожний біт можна побачити додаткову інформацію про місце знаходження певного біту в отриманому байті (рис. 4.46)

```
Modbus
    .000 0010 = Function Code: Read Discrete Inputs (2)
    [Request Frame: 171530]
    [Time from request: 0.005866000 seconds]
    Byte Count: 1
    Bit 0 : 0
        [Bit Number: 0]
        .... ...0 = Bit Value: False
    Bit 1 : 1
        [Bit Number: 1]
        .... ..1. = Bit Value: True
    Bit 2 : 0
        [Bit Number: 2]
        .... .0.. = Bit Value: False
    Bit 3 : 1
        [Bit Number: 3]
        .... 1... = Bit Value: True
    Bit 4 : 0
        [Bit Number: 4]
        ...0 .... = Bit Value: False
    Bit 5 : 1
        [Bit Number: 5]
        ..1. .... = Bit Value: True
    Bit 6 : 0
        [Bit Number: 6]
        .0.. .... = Bit Value: False
    Bit 7 : 1
        [Bit Number: 7]
        1... .... = Bit Value: True
```

Рисунок 4.46 – Докладна інформація про кожен біт отриманих даних

Виконаємо аналіз мережевого трафіку в процесі виконання задачі читання стану регістрів зберігання (Holding Register). Для цього підготуємо набір чисел «12345, 100, 200, 300, 159» та запишемо їх у відповідні регістри програми Modbus Simulator, починаючи з першого (рис. 4.47).

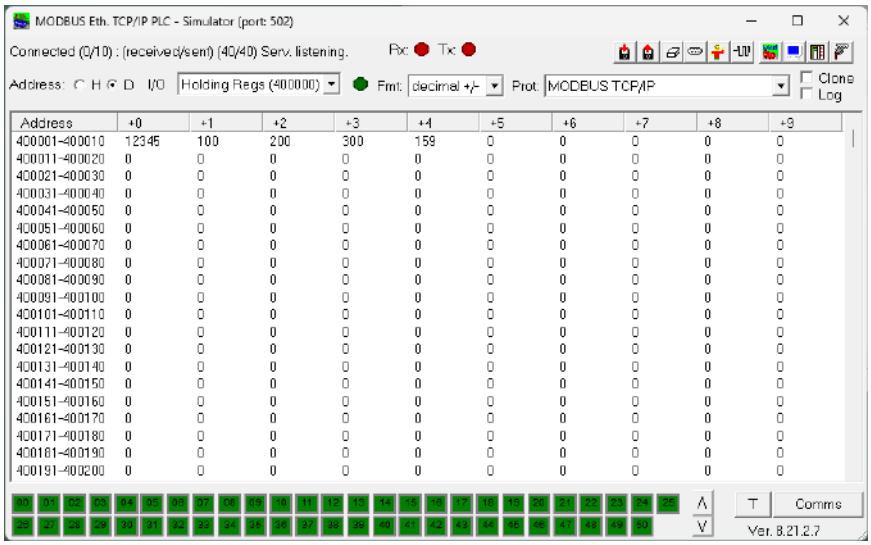


Рисунок 4.47 – Вміст регістрів зберігання

Для читання стану декількох регістрів зберігання використовується функція 03. У випадку виконання цієї функції запит та відповідь у програмі ModbusMaster TCP подана на рис. 4.48.

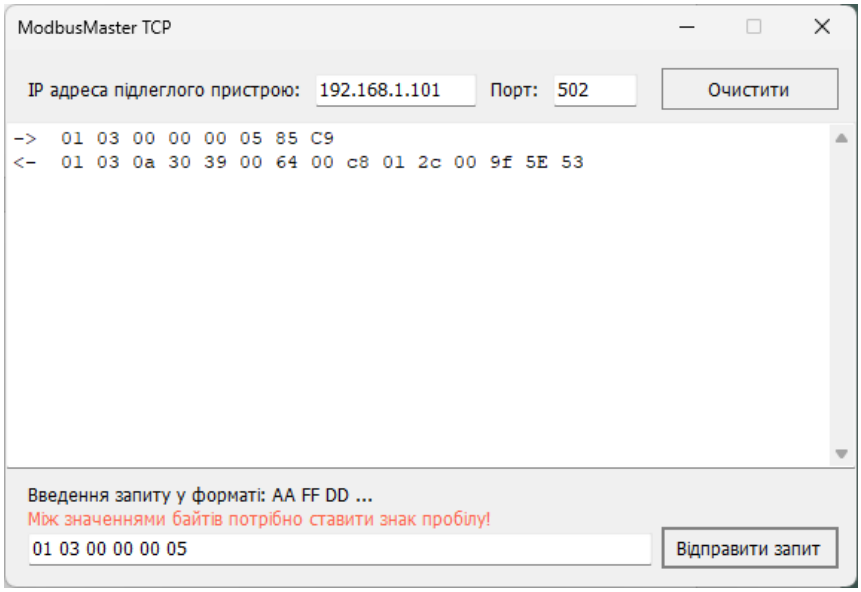


Рисунок 4.48 – Обмін інформацією в режимі читання п’яти регістрів зберігання

Результат аналізу мережевого трафіку та відповіді від програми Modbus Simulator, де можна бачити значення, що зберігаються в кожному регістрі окремо, подано на рис. 4.49.

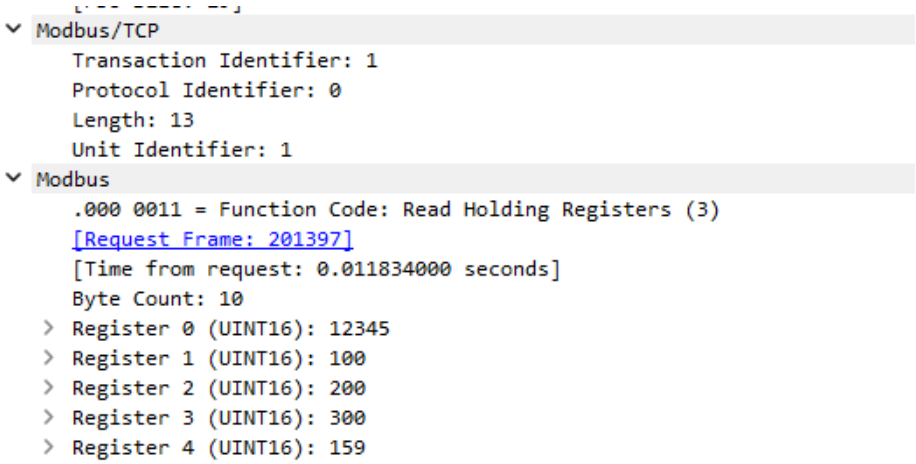


Рисунок 4.49 – Результат аналізу мережевого трафіку та відповіді від програми Modbus Simulator

4.7 Контрольні запитання та завдання

1. Що таке протокол Modbus і для яких задач він використовується у промислових мережах?

2. Які основні принципи організації обміну даними за протоколом Modbus?

3. Які ролі виконують пристрої у моделі взаємодії Modbus?

4. Які існують різновиди протоколу Modbus та чим вони відрізняються між собою?

5. У чому особливості протоколу Modbus RTU?

6. Які характеристики має протокол Modbus ASCII та де він застосовується?

7. У чому полягають переваги використання протоколу Modbus TCP?

8. Що таке регістри Modbus і які їх основні типи?

9. Як здійснюється адресація регістрів у протоколі Modbus?

10. Яке призначення функції читання вихідних контактів (01)?

11. У чому полягає відмінність між функціями читання дискретних входів (02) та регістрів зберігання (03)?

12. Які функції використовуються для запису даних у регістри Modbus?

13. Які можливості надає програмний засіб Wireshark для аналізу мережевого трафіку?

14. Які етапи підготовки необхідно виконати перед проведенням аналізу трафіку в промисловій мережі?

15. Як за допомогою аналізу мережевого трафіку можна виявити несанкціоноване або непередбачуване втручання в роботу системи Modbus?

5. МОНІТОРИНГ РАДІОЧАСТОТНОГО СЕРЕДОВИЩА ДЛЯ ВИЯВЛЕННЯ КІБЕРЗАГРОЗ В АСУТП

5.1 Особливість бездротових сегментів АСУТП з точки зору кібербезпеки

Задача аналізу радіочастотного середовища стає дедалі актуальнішою для кібербезпеки автоматизованих систем управління технологічними процесами (АСУТП), що зумовлено стрімким впровадженням бездротових технологій у промислову автоматизацію та критичну інфраструктуру. У цьому контексті радіочастотний аналіз слід розглядати як невід'ємну складову системи виявлення загроз, призначену для моніторингу відкритого ефіру, через який здійснюється обмін керуючими та вимірювальними даними між елементами системи.

Особливість бездротових сегментів АСУТП полягає в тому, що вони функціонують у відкритому електромагнітному просторі, доступному для спостереження та впливу з боку сторонніх осіб без необхідності фізичного проникнення на об'єкт. Це принципово відрізняє їх від класичних дротових мереж і зумовлює появу специфічного класу загроз, які не завжди можуть бути виявлені традиційними засобами мережевої або інформаційної безпеки. Радіочастотний моніторинг у цьому випадку дозволяє здійснювати постійне спостереження за реальним станом ефіру, виявляти нові або нетипові сигнали, аналізувати їх параметри та співвідносити отримані дані з проєктною документацією та допустимими режимами роботи системи.

З точки зору вирішення завдань кібербезпеки, аналіз радіочастотного середовища дає змогу ідентифікувати несанкціоновані бездротові пристрої, які можуть з'являтися в зоні дії промислової мережі, виявляти факти підміни або емуляції легітимних вузлів, а також фіксувати спроби навмисного впливу на канал зв'язку шляхом зашумлення або створення перешкод. Для АСУТП такі впливи є особливо небезпечними, оскільки навіть короткочасне порушення зв'язку між датчиками, контролерами та виконавчими механізмами може призвести до збоїв у технологічному процесі, аварійних зупинок або переходу системи в небезпечні режими. Окрім активних атак, радіочастотний аналіз дозволяє виявляти ознаки пасивної розвідки, коли зловмисник не передає

власних сигналів, але здійснює тривале спостереження за ефіром з метою збору інформації про структуру мережі, режими роботи обладнання та часові закономірності обміну даними.

Необхідною умовою застосування радіочастотного аналізу є оцінка реального рівня захищеності бездротових каналів зв'язку. Практика показує, що параметри потужності передавачів, розташування антен і особливості поширення радіохвиль часто призводять до того, що сигнали АСУТП виходять далеко за межі контрольованої території підприємства, створюючи додаткові можливості для несанкціонованого доступу. Систематичний моніторинг електромагнітної обстановки дозволяє виявляти такі ситуації та коригувати конфігурацію мережі з урахуванням вимог фізичної та інформаційної безпеки, що є важливою складовою відповідності сучасним стандартам захисту промислових систем.

Аналіз радіочастотного середовища в задачах кібербезпеки АСУТП слід розглядати як міждисциплінарний напрям, що поєднує методи радіотехніки, цифрової обробки сигналів і інформаційної безпеки. Його основне призначення полягає у своєчасному виявленні загроз, які не проявляються на логічному або мережевому рівні, але здатні безпосередньо впливати на надійність і безпечність технологічних процесів. Саме тому радіочастотний моніторинг дедалі частіше розглядається не як допоміжний інструмент, а як повноцінний елемент комплексної системи кіберзахисту критично важливих промислових об'єктів.

Швидке зростання кількості бездротових пристроїв для розвитку промислового Інтернету речей призводить до збільшення ризиків і вразливостей. Типи бездротових атак різноманітні, і не існує єдиної класифікації. Серед найвідоміших: підробка, прослуховування, побічний канал, заглушування, повторна передача та підробка. Всі вони базуються на використанні різних вразливостей, присутніх у радіопристроях або в їхніх протоколах зв'язку.

На рис. 5.1 подано класифікацію засобів бездротового зв'язку для промислового Інтернету речей, систематизовану за радіусом їхньої дії [11]. Залежно від дальності зв'язку, бездротові мережі класифікують за охопленням території: від персональних (WPAN) і локальних (WLAN) до регіональних (WMAN) та глобальних мереж (WWAN).

Сучасні бездротові технології зв'язку функціонують у широкому спектрі частотних діапазонів – від низькочастотних ISM-діапазонів (13,56 МГц) до високочастотних діапазонів понад 5 ГГц (рис. 5.2).

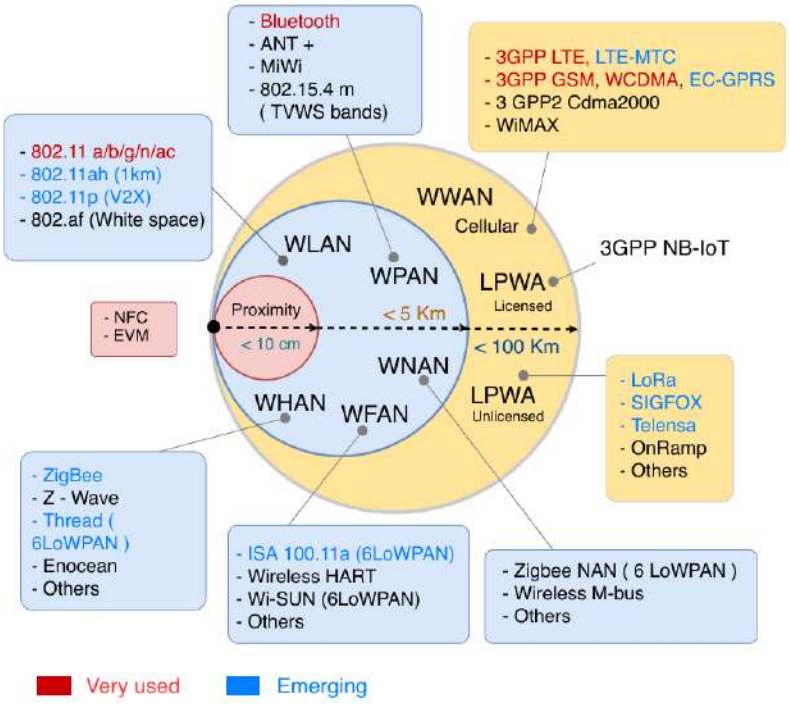


Рисунок 5.1 – Класифікація засобів бездротового зв’язку для промислового Інтернету речей, систематизована за радіусом їхньої дії

Вибір частотного діапазону безпосередньо впливає на характеристики зв’язку: дальність, швидкість передачі даних, проникність через перешкоди та енергоспоживання. Розглянемо застосування різних частот у технологіях різних класів мереж.

		54-698 MHz (TVWS)												MHZ
TECHNOLOGY	**STANDAR**	13.56	169	220	315	426	433	470	779	868	915	920	2400	5800	5900
NFC/EMV	ISO14443	●													
Wireless M-Bus	EN13757-4		●				●			●					
China WMRNET	WMRNET I,II, III, IV						●	●							
LoRa							●	●		●	●				
SIGFOX										●	●				
Telensa										●	●				
OnRamp (INGENU)	802.15 4 k												●		
Wi-SUN	802.15.4g/e/6LoWPAN											●			
ZigBee	802.15.4-2003, c d								●	●	●	●	●		
Thread	802.15.4-2003/6LoWPAN												●		
WirelessHART	802.15.4e												●		
ISA100.11a	802.15.4e/6LoWPAN												●		
	802.15.4m			●											
Z-Wave	ITU G9959									●	●	●			
EnOcean	ISO14543-3-10				●						●	●			
ANT+													●		
Bluetooth	802.15.1												●		
WiFi	802.11/a/b/g/n/ac												●	●	
WiFi HaLow	802.11ah (HaLow)								●	●	●	●			
V2X	802.11p														●
White Space	802.11af			●											
Positive Train Ctrl	802.15.4p			●											

Рисунок 5.2 – Спектр частотних діапазонів сучасної бездротові технології зв’язку

Низькочастотний діапазон (13,56-169 МГц).

NFC/EMV (13,56 МГц) функціонує на найнижчій частоті серед представлених технологій. Цей діапазон забезпечує роботу на надкоротких відстанях (до 10 см), що є критичним для безпечних платіжних операцій та ідентифікації. Низька частота дозволяє ефективно працювати з металевими поверхнями та забезпечує мінімальне енергоспоживання.

Wireless M-Bus (169 МГц) використовує дещо вищу частоту для автоматизованого зчитування показників лічильників. Цей діапазон обраний через відмінну проникність сигналу крізь стіни будівель та підвальні приміщення, де зазвичай розміщуються лічильники.

Субгігагерцові діапазони (315-920 МГц)

Цей частотний діапазон характеризується оптимальним балансом між дальністю зв'язку та швидкістю передачі даних, що робить його ідеальним для IoT-застосувань.

China WMRNET (433, 470 МГц) використовує нижню частину субгігагерцового діапазону, забезпечуючи дальність зв'язку кілька кілометрів з помірним енергоспоживанням.

LoRa (433, 470, 868, 920 МГц) демонструє гнучкість частотного планування, працюючи у декількох регіональних ISM-діапазонах. Низькі частоти забезпечують дальність зв'язку до 15 км у сільській місцевості, що робить технологію придатною для мереж LPWAN (Low Power Wide Area Network).

SIGFOX (868, 920 МГц) орієнтована на глобальні IoT-мережі з мінімальним енергоспоживанням. Вузькосмугова модуляція у поєднанні з субгігагерцовими частотами дозволяє досягти десятирічної автономності батарейного живлення.

Telensa (868, 920 МГц) спеціалізується на системах «розумного» освітлення, використовуючи ті ж діапазони для забезпечення надійного зв'язку в міському середовищі.

OnRamp INGENU (2400 МГц) є винятком, використовуючи діапазон 2,4 ГГц, що дозволяє досягти вищих швидкостей передачі даних з дещо меншою дальністю.

Діапазон 2,4 ГГц (глобальний ISM-діапазон)

Діапазон 2,4 ГГц є найбільш універсальним та перевантаженим частотним ресурсом, що використовується численними технологіями.

IEEE 802.15.4 та похідні протоколи:

– ZigBee забезпечує mesh-топологію для домашньої автоматизації;

– Thread орієнтований на IoT-пристрої у розумному домі;

– 802.15.4m надає базову специфікацію для низькошвидкісних WPAN.

Wi-Fi (802.11 b/g/n) домінує у сегменті локальних мереж, забезпечуючи швидкості до 600 Мбіт/с у стандарті 802.11n. Діапазон 2,4 ГГц обраний через компроміс між проникністю сигналу та пропускною здатністю.

Bluetooth (802.15.1) використовує частотні стрибки у діапазоні 2,4 ГГц для персональних мереж, забезпечуючи стійкість до завад.

WiFi HaLow (802.11ah) працює у субгігагерцових діапазонах (868, 920 МГц), але також має присутність у 2,4 ГГц для IoT-застосувань з підвищеною дальністю.

Високочастотні діапазони (5-6 ГГц)

Wi-Fi (802.11a/n/ac) активно використовує діапазони 5 ГГц (5150-5900 МГц) для досягнення високих швидкостей передачі даних. Менша завантаженість цього діапазону порівняно з 2,4 ГГц забезпечує стабільнішу роботу, хоча дальність зв’язку дещо менша через вищі втрати поширення.

V2X (802.11p) використовує виділений діапазон 5,9 ГГц для комунікації між транспортними засобами, де критична низька затримка та висока надійність зв’язку.

White Space (802.11af) інноваційно використовує вільні телевізійні канали у діапазоні 470 МГц, досягаючи великої дальності зв’язку при хорошій проникності.

Спеціалізовані технології

Z-Wave (868, 920 МГц) використовує субгігагерцові частоти для домашньої автоматизації, забезпечуючи відмінну проникність крізь стіни та низьке енергоспоживання.

EnOcean (315, 868, 920 МГц) застосовує технологію збору енергії (energy harvesting), працюючи у кількох субгігагерцових діапазонах залежно від регіону.

WirelessHART та ISA100.11a (2400 МГц) орієнтовані на промислову автоматизацію, використовуючи глобальний ISM-діапазон 2,4 ГГц для забезпечення детермінованого зв’язку у важких умовах.

Positive Train Control (169, 220 МГц) використовує низькі частоти для залізничного зв’язку, забезпечуючи дальність у десятки кілометрів та стабільність на високих швидкостях руху.

Широкий спектр частотних діапазонів, що використовуються у бездротових технологіях АСУТП – від 13,56 МГц (NFC) до 5,9 ГГц (V2X), створює розгалужену поверхню атаки з різними векторами загроз для кожного діапазону. Низькочастотні технології (169-920 МГц), такі як LoRa, SIGFOX та Wireless M-Bus, забезпечують дальність зв'язку до 15 км, що дозволяє зловмисникам здійснювати атаки з безпечної відстані поза фізичним периметром об'єкта, ускладнюючи виявлення та локалізацію джерела загрози. Перевантажений діапазон 2,4 ГГц, в якому працюють критичні промислові протоколи (WirelessHART, ISA100.11a, ZigBee), є особливо вразливим до атак типу «відмова в обслуговуванні» через глушіння, оскільки у цьому ж діапазоні функціонують побутові пристрої Wi-Fi та Bluetooth, що створює природне радіочастотне забруднення. Комплексний підхід до кібербезпеки АСУТП повинен включати постійний моніторинг радіочастотного спектру у всіх використовуваних діапазонах, криптографічний захист на всіх рівнях OSI-моделі, впровадження систем виявлення аномалій у бездротовому трафіку та фізичне екранування критичних зон об'єкта для обмеження дальності поширення сигналу.

5.2 Методи та технології моніторингу радіочастотного середовища для виявлення кіберзагроз в АСУТП

5.2.1 Класифікація радіочастотних атак

Кіберзагрози через бездротовий канал в АСУТП можна класифікувати за механізмом впливу:

– пасивні атаки;

– активні атаки типу «відмова в обслуговуванні»;

– атаки підміни та ін'єкції;

– атаки на рівні протоколу.

Пасивні атаки передбачають несанкціоноване перехоплення радіосигналів без активного втручання у функціонування мережі. Зловмисник може здійснювати моніторинг промислового трафіку для збору розвідувальної інформації про топологію мережі, типи обладнання, протоколи зв'язку та технологічні параметри процесу. Особливу небезпеку становить перехоплення незашифрованих команд керування або слабо захищених автентифікаційних даних.

Активні атаки типу «відмова в обслуговуванні» (DoS/DDoS) реалізуються шляхом радіочастотного глушіння у діапазонах роботи промислових протоколів. Атака на діапазон 2,4 ГГц, де функціонують WirelessHART та ISA100.11a, може призвести до втрати зв'язку з критичними датчиками тиску, температури або рівня, що у свою чергу може спричинити аварійну ситуацію на виробництві.

Атаки підміни та ін'єкції передбачають передачу несанкціонованих команд керування або підроблених даних телеметрії. Використовуючи слабкості протоколів автентифікації, зловмисник може видати себе за легітимний польовий пристрій або контролер, передаючи хибну інформацію або небезпечні керуючі команди.

Атаки на рівні протоколу експлуатують вразливості специфічних особливостей промислових протоколів, таких як відсутність взаємної автентифікації, слабкі криптографічні алгоритми або передбачувані послідовності пакетів.

На рис. 5.3 подана класифікація видів кібератак на бездротові мережі.

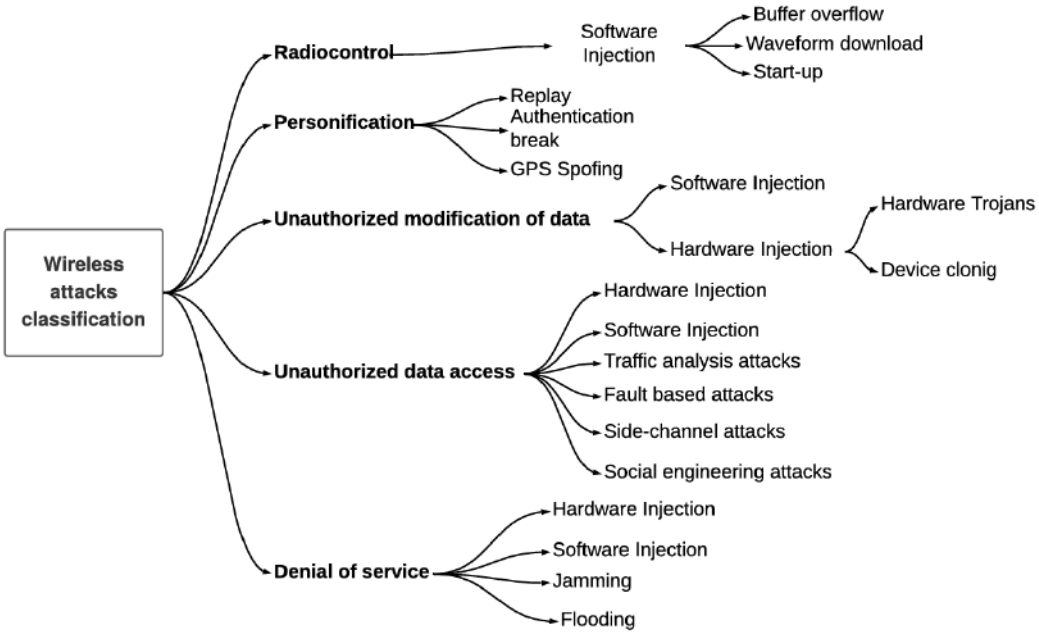


Рисунок 5.3 – Класифікація видів кібератак на бездротові мережі

5.2.2 Спеціалізовані системи виявлення вторгнень (WIDS)

Промислові системи виявлення бездротових вторгнень (WIDS, Wireless Intrusion Detection Systems) є першим рівнем захисту радіочастотного середовища АСУТП [12, 13]. Такі системи використовують мережу стаціонарних

сенсорів, розміщених по периметру та всередині об’єкта, що захищається, які постійно аналізують радіоефір у специфічних частотних діапазонах (рис. 5.4).

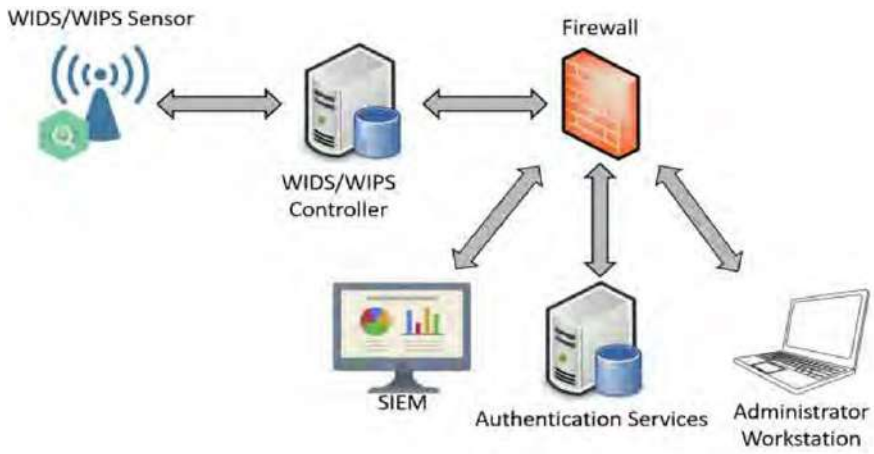


Рисунок 5.4 – Промислові системи виявлення бездротових вторгнень

Типова архітектура WIDS включає розподілені радіосенсори, центральний сервер аналізу та консоль оператора. Сенсори здійснюють безперервний моніторинг радіочастотного спектру, декодування пакетів легітимних протоколів та виявлення аномалій. Система формує базу даних відомих пристроїв (whitelist) та виявляє несанкціоновані джерела випромінювання.

Основні обмеження традиційних WIDS полягають у їхній вузькій спеціалізації: кожна система зазвичай підтримує обмежений набір протоколів (наприклад, лише Wi-Fi або лише ZigBee) та працює у фіксованому частотному діапазоні. Для комплексного моніторингу середовища, де одночасно функціонують WirelessHART (2,4 ГГц), Wireless M-Bus (169 МГц), LoRa (868/920 МГц) та інші технології, необхідне розгортання великої кількості спеціалізованих систем, що значно підвищує вартість та складність адміністрування.

5.2.3 Промислові системи радіомоніторингу

Деякі виробники промислового обладнання пропонують інтегровані рішення для моніторингу власних бездротових мереж. Наприклад, WirelessHART Network Manager включає функції діагностики якості радіоканалу, виявлення завад та оптимізації mesh-топології. ISA100.11a передбачає механізми виявлення аномалій у затримках доставки пакетів та рівні помилок.

Ці рішення є ефективними для моніторингу працездатності та оптимізації продуктивності бездротової мережі, однак мають обмежені можливості з точки зору кібербезпеки. Вони зазвичай не призначені для виявлення складних атак, таких як підміна легітимних пристроїв або тонке маніпулювання трафіком, і не забезпечують моніторинг загроз з інших частотних діапазонів, які можуть використовуватися для розвідки або координації атак.

5.2.4 Програмно-визначена радіосистема (SDR)

Програмно-визначена радіосистема (Software-Defined Radio, SDR) являє собою радикально інший підхід до проєктування радіообладнання, де більшість функцій традиційного апаратного тракту (модуляція, демодуляція, фільтрація, декодування) реалізуються програмно на універсальному обчислювальному пристрої замість спеціалізованих апаратних компонентів. Типова архітектура SDR включає мінімальний апаратний фронтенд (радіочастотний тракт, змішувачі, аналого-цифрові перетворювачі) та потужний програмний компонент, що виконується на процесорі загального призначення, сигнальному процесорі (DSP) або програмованій логічній матриці (FPGA) (рис. 5.5).

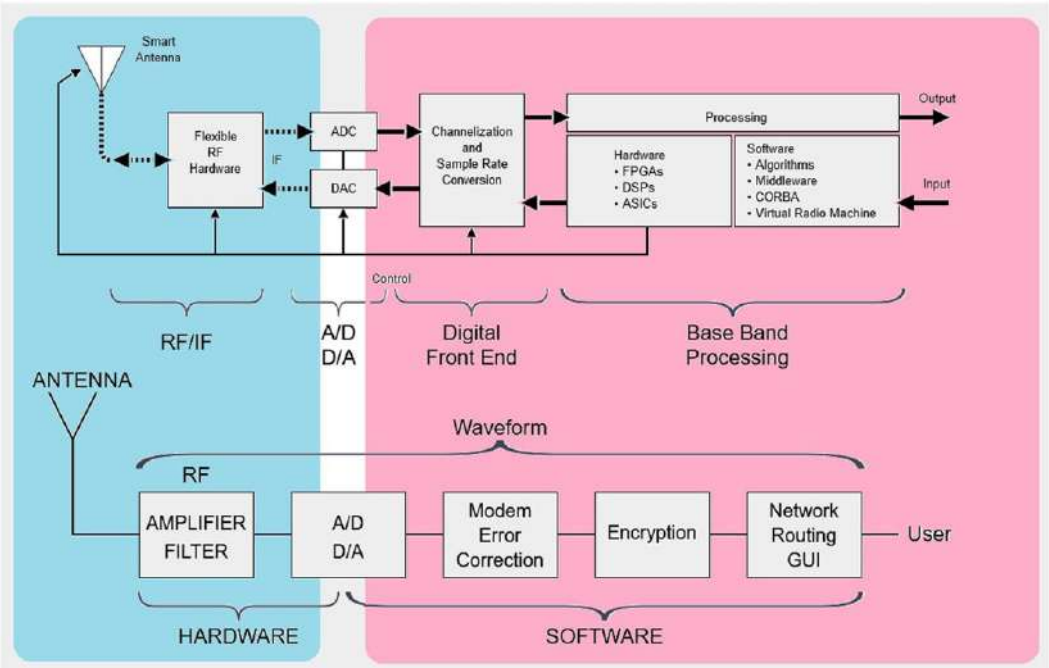


Рисунок 5.5 – Типова структура SDR

Антена приймає радіосигнал, який після підсилення та частотного перетворення оцифровується АЦП з високою частотою дискретизації. Отримані

цифрові відліки (IQ-samples) передаються до обчислювального пристрою, де програмне забезпечення здійснює всю подальшу обробку: фільтрацію, демодуляцію, декодування протоколів та аналіз.

Ключова перевага SDR полягає у його універсальності та гнучкості. Одне й те ж апаратне обладнання може бути програмно переконфігуроване для роботи у різних частотних діапазонах, з різними типами модуляції та протоколами зв'язку. Це досягається простим завантаженням відповідного програмного забезпечення без будь-яких апаратних змін.

SDR-пристрої класифікуються за діапазоном частот, смугою пропускання, розрядністю АЦП та функціональними можливостями. Бюджетні USB-приймачі (RTL-SDR, Airspy) базуються на мікросхемах цифрових TV-тюнерів та забезпечують прийом у діапазонах від 24 МГц до 1,7 ГГц зі смугою до 3,2 МГц. Відносно невисока вартість таких пристроїв робить їх доступними для масового розгортання розподілених систем моніторингу.

Напівпрофесійні трансивери (HackRF One, LimeSDR) підтримують як прийом, так і передачу у діапазонах від 1 МГц до 6 ГГц зі смугою до 20 МГц. Ці пристрої використовуються не лише для моніторингу, але й для тестування захищеності систем, дослідження протоколів та розробки нових рішень.

Професійні платформи (USRP серії B/N/X від Ettus Research) забезпечують високу якість сигналу, широку смугу пропускання (до 160 МГц), підтримку MIMO та можливість роботи у реальному часі. Використовуються для критичних застосувань, де необхідна максимальна точність та надійність.

Можна виділити характерні особливості технології SDR, що виділяє її на фоні інших інструментів аналізу кібератак в АСУТП. Сучасні SDR-пристрої покривають діапазони від декількох МГц до 6 ГГц, що дозволяє одному пристрою замінити безліч спеціалізованих приймачів для різних частотних діапазонів. Система на базі SDR може одночасно моніторити Wireless M-Bus на 169 МГц, LoRa на 868 МГц, WirelessHART на 2,4 ГГц та Wi-Fi на 5 ГГц, швидко перемикаючись між діапазонами або використовуючи декілька SDR-пристроїв, що працюють паралельно.

На відміну від апаратних рішень із закритою архітектурою, SDR-система може бути оновлена програмно для підтримки нових протоколів або методів виявлення атак. Коли з'являється нова промислова бездротова технологія або виявляється новий тип атаки, достатньо розробити та завантажити відповідний програмний модуль без заміни обладнання.

Низька вартість бюджетних SDR-пристроїв (порівняно зі спеціалізованим обладнанням) дозволяє створювати розподілені системи моніторингу з високою просторовою щільністю сенсорів. Це критично важливо для точної локалізації джерел загроз методом тріангуляції та для забезпечення повного покриття великих промислових об’єктів.

SDR надає доступ до сирих IQ-відліків сигналу, що дозволяє здійснювати аналіз на фізичному рівні (модуляція, потужність, частотні характеристики), канальному рівні (MAC-адреси, структура кадрів), мережевому рівні (маршрутизація, топологія) та рівні застосувань (зміст команд керування). Така багаторівнева аналітика неможлива при використанні більшості спеціалізованих рішень.

Більшість SDR-платформ підтримують відкриті стандарти (GNU Radio, SoapySDR) та мають добре документовані API, що полегшує інтеграцію з існуючими системами управління безпекою (SIEM), створення власних алгоритмів виявлення вторгнень та обмін даними між компонентами системи моніторингу.

Двонаправлені SDR-пристрої (HackRF, USRP) дозволяють не лише моніторити пасивно, але й проводити контрольоване тестування захищеності бездротових мереж АСУТП, генеруючи тестові атаки у контрольованому середовищі для перевірки ефективності засобів захисту.

У табл. 5.1 подано порівняння пристроїв SDR із різними технічними параметрами [14].

5.3 Спеціалізовані пристрої для аудиту безпеки бездротових мереж АСУТП та методи протидії

5.3.1 Платформа для атак на бездротові мережі WiFi Pineapple

WiFi Pineapple, що випускається компанією Hak5, є спеціалізованим апаратно-програмним комплексом для аудиту безпеки бездротових мереж стандарту IEEE 802.11 (Wi-Fi). Зовнішній вигляд пристрою подано на рис. 5.6.

Пристрій базується на системі-на-чипі (SoC) з архітектурою ARM, працює під керуванням модифікованої операційної системи OpenWrt (Linux) та оснащений двома або більше бездротовими адаптерами з підтримкою режиму точки доступу (AP) та моніторингу.

Таблиця 5.1 – Порівняльні характеристики пристроїв SDR

Платфома	**ADC/DAC, біт**	**ADC/DAC, Мвиб./с**	**Tx/Rx**	**F_{min}-F_{max}, МГц**
RTL [32]	8/-	3.2/-	Rx	25-1750
FUNCube [33]	16/-	96 kHz	Rx	64-1700
Airspy-mini [34]	12/-	20/-	Rx	24-1700
HackRF One [35]	8/10	20/20	Tx-Rx	1-6000
Pluto [36]	12/12	61.4/61.4	Tx-Rx	325-3800
BladeRF 2.0 Micro [37]	12/12	61.4/61.4	Tx-Rx	47-6000
USRP-1 [38]	12/14	64/128	Tx-Rx	DC-6000
Nuand PicoSDR [39]	12/12	80/80	Tx-Rx	56-6000
USRP-2 [38]	14/16	100/400	Tx-Rx	DC-6000
USRP-N210 [38]	14/16	100/400	Tx-Rx	DC-6000
WARP-V3 [40]	12/12	100/170	Tx-Rx	2400/5000
LimeRF LMS7002M [41]	12/12	160/640	Tx-Rx	0.1-3800
USRP X310 [38]	14/16	200/800	Tx-Rx	DC-6000
USRP N310 [38]	16/14	153.6/153.6	Tx-Rx	10-6000
AIR-T [42]	12/10	245.7/245.7	Tx-Rx	300-6000
CRIMSON [43]	16/16	370/160, 2500/16	Tx-Rx	0.1-6000

Сучасні моделі (наприклад, WiFi Pineapple Mark VII) підтримують діапазони 2,4 ГГц та 5 ГГц, мають вбудовану батарею для автономної роботи протягом 2-4 годин, Ethernet-інтерфейс для підключення до провідної мережі, USB-порти для розширення функціональності та веб-інтерфейс для управління з будь-якого пристрою через Wi-Fi або провідне з’єднання. Компактні розміри дозволяють легко приховати пристрій у загальнодоступних місцях промислового об’єкта.

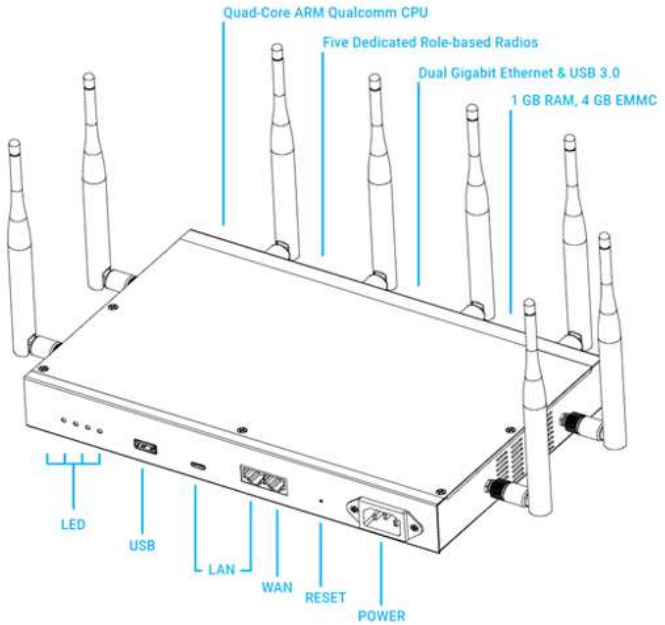


Рисунок 5.6 – Зовнішній вигляд WiFi Pineapple

WiFi Pineapple реалізує широкий спектр атак на бездротові мережі.

Атака «Evil Twin» (Злий близнюк). Пристрій створює фальшиву точку доступу з SSID (ім'ям мережі), ідентичним легітимній корпоративній або промисловій мережі. Коли користувачі або автоматизовані пристрої намагаються підключитися до знайомої мережі, вони фактично з'єднуються з WiFi Pineapple. Зловмисник отримує можливість перехоплювати весь трафік, включаючи незашифровані дані автентифікації, команди керування SCADA-системами або телеметрію з польових пристроїв.

Автоматизована атака через PineAP. Модуль PineAP є ключовою особливістю платформи, що автоматизує процес залучення клієнтів. Пристрій пасивно моніторить ефір, збираючи інформацію про SSID мереж, до яких раніше підключалися клієнтські пристрої поблизу (ця інформація передається у probe request кадрах). Потім WiFi Pineapple створює множину фальшивих точок доступу з цими SSID, максимізуючи ймовірність того, що якийсь пристрій автоматично підключиться. У промисловому середовищі це може призвести до компрометації планшетів операторів, ноутбуків інженерів або навіть бездротових HMI-панелей, налаштованих на автоматичне підключення до корпоративної мережі.

Атака типу "Man-in-the-Middle" (MITM). Після успішного підключення жертви до фальшивої точки доступу, WiFi Pineapple може виступати проксі між клієнтом та легітимною мережею (якщо має доступ до неї через Ethernet або іншу мережу). Це дозволяє не лише пасивно перехоплювати трафік, але й активно модифікувати його – підміняти команди керування, впроваджувати шкідливий код у веб-сторінки, що завантажуються операторами, або здійснювати SSL-stripping для розшифрування HTTPS-трафіку.

Збір облікових даних (Credential Harvesting). Пристрій може створювати фальшиві портали авторизації (captive portal), що імітують інтерфейс входу у корпоративну мережу. Користувачі, намагаючись підключитися до мережі, вводять свої облікові дані (логін та пароль від Active Directory, SCADA-системи тощо), які перехоплюються зловмисником.

DNS Spoofing та HTTP ін'єкції. WiFi Pineapple може підміняти DNS-відповіді, перенаправляючи запити до легітимних серверів (наприклад, веб-інтерфейсу ПЛК або HMI) на контрольовані зловмисником сервери. Крім того, пристрій може впроваджувати JavaScript-код у незашифровані HTTP-відповіді для компрометації браузерів користувачів.

В АСУТП WiFi Pineapple становить особливо серйозну загрозу з кількох причин.

По-перше, ноутбуки інженерів, що використовуються для програмування ПЛК, налаштування SCADA-систем або діагностики обладнання, часто налаштовані на автоматичне підключення до корпоративної Wi-Fi мережі. При підключенні до фальшивої точки доступу зловмисник може перехопити облікові дані, отримати доступ до проєктної документації або впровадити шкідливе ПЗ безпосередньо у середовище розробки.

По-друге, якщо бездротова мережа використовується для передачі даних промислових протоколів (Modbus/TCP, OPC UA, PROFINET тощо), атака MITM дозволяє перехоплювати телеметрію, команди керування та модифікувати їх у реальному часі, що може призвести до аварійних ситуацій.

По-третє, навіть якщо промислова мережа фізично ізольована, злочинець може використати скомпрометований через WiFi Pineapple ноутбук інженера як «місток» для проникнення у критичні сегменти мережі АСУТП.

По-п’яте, перехоплення трафіку дозволяє зловмиснику отримати детальну карту топології мережі, типів обладнання, версій програмного забезпечення та вразливостей, що можуть бути використані для підготовки більш складних цільових атак.

5.3.2 Пристрій для атак відмови в обслуговуванні WiFi Deauther

WiFi Deauther (деавтентифікатор) є компактним пристроєм, зазвичай побудованим на базі мікроконтролера ESP8266 або ESP32 з вбудованим Wi-Fi модулем (рис. 5.7). Розмір пристрою може бути дуже малим – від розміру USB-флешки до кредитної картки, що дозволяє легко приховати його на промисловому об’єкті. Пристрій живиться від вбудованої батареї або USB-порту та керується через веб-інтерфейс або фізичні кнопки.

Принцип роботи базується на експлуатації вразливості у стандарті IEEE 802.11, пов’язаної з незахищеністю керуючих кадрів (management frames). Згідно зі стандартом, точка доступу або клієнт можуть надіслати кадр деавтентифікації (deauthentication frame) для коректного розриву з’єднання. Критично важливо, що до впровадження стандарту 802.11w (Protected Management Frames) ці кадри передавалися незашифрованими та без автентифікації відправника.

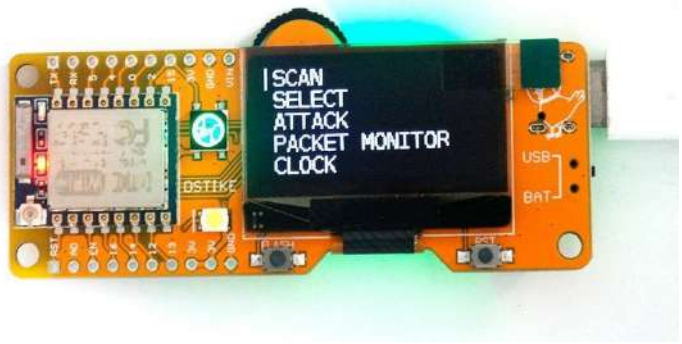


Рисунок 5.7 – Зовнішній вигляд пристрою WiFi Deauther

WiFi Deauther використовує цю вразливість, генеруючи підроблені кадри деавтентифікації від імені точки доступу або клієнта. Отримавши такий кадр, пристрій припиняє з’єднання, вважаючи запит легітимним. Якщо атака триває безперервно, пристрої не можуть відновити з’єднання, оскільки кожна спроба автентифікації негайно перебивається новим деавтентифікаційним кадром.

Даний пристрій може застосовуватись для декількох режимів атак.

Цільова деавтентифікація (Targeted Deauth*)* спрямована на конкретного клієнта у мережі. Зловмисник спочатку сканує ефір для виявлення MAC-адрес активних клієнтів та точок доступу, після чого обирає жертву та надсилає деавтентифікаційні кадри лише для цього з’єднання. Це дозволяє провести атаку менш помітною для систем моніторингу.

Масова деавтентифікація (Broadcast Deauth) спрямована на всі пристрої у радіусі дії. WiFi Deauther надсилає деавтентифікаційні кадри з широкомовною MAC-адресою (FF:FF:FF:FF:FF:FF), що змушує всі клієнти точки доступу одночасно розірвати з’єднання. Це створює повну відмову в обслуговуванні бездротової мережі у зоні покриття.

Деякі реалізації дозволяють одночасно атакувати десятки точок доступу, послідовно або паралельно надсилаючи деавтентифікаційні кадри для кожної виявленої мережі, створюючи радіочастотний хаос у всьому діапазоні 2,4 ГГц.

Окрім DoS-атак, WiFi Deauther часто використовується для підготовки до злому WPA/WPA2 мереж. Алгоритм атаки наступний:

– зловмисник налаштовує обладнання для перехоплення трафіку (наприклад, ESP8266 з відповідним програмним забезпеченням або ноутбук з адаптером у режимі моніторингу);

– WiFi Deauther надсилає кадр деавтентифікації легітимному клієнту мережі, примусово розриваючи його з’єднання;

– клієнт автоматично намагається повторно підключитися до точки доступу, ініціюючи процес 4-way handshake (обмін криптографічними ключами для WPA/WPA2);

– зловмисник перехоплює цей handshake, що містить хеш пароля мережі;

– перехоплений handshake зберігається та згодом піддається брутфорс-атаці offline (перебір паролів зі словника або повний перебір) на потужних обчислювальних системах або у хмарі;

– якщо підбір пароля виявляється успішним, зловмисник отримує доступ до мережі.

Застосування сценарію атак відмови в обслуговуванні становить ряд загроз для правильного функціонування АСУТП.

Якщо бездротова мережа використовується для зв’язку між оператором та системою керування (бездротові HMI-панелі, планшети операторів), атака деавтентифікації може призвести до втрати можливості моніторингу технологічного процесу та оперативного втручання у критичних ситуаціях.

Польові пристрої, що підключаються до мережі через Wi-Fi (температурні датчики, датчики тиску, рівня тощо), при деавтентифікації втрачають зв’язок з контролером. Це може призвести до помилкових спрацювань систем безпеки або, навпаки, до неспрацювання при реальній аварійній ситуації.

Використання деавтентифікації для перехоплення WPA2 handshake дозволяє зловмиснику offline зламати пароль від промислової мережі без ризику виявлення, оскільки активна частина атаки (деавтентифікація) триває лише кілька секунд та може бути розцінена як випадковий збій зв’язку.

Завдяки мініатюрним розмірам, WiFi Deauther може бути залишений на об’єкті (наприклад, за фальш-стелею, у електрощитовій, за обладнанням) та активований дистанційно або працювати за таймером, регулярно створюючи збої у роботі бездротової інфраструктури.

5.3.3 Методи виявлення атак із застосуванням WiFi Pineapple

Одним з перших є використання методу моніторингу аномалій у SSID. В даному випадку системи WIDS (Wireless Intrusion Detection Systems) повинні вести базу даних легітимних точок доступу з їхніми MAC-адресами, SSID та місцем розташування. Поява нової точки доступу з відомим SSID, але невідомою

MAC-адресою є індикатором атаки "Evil Twin". Сучасні WIDS можуть виявляти такі аномалії у реальному часі та оповіщати адміністраторів.

Іншим методом є аналіз Probe Response аномалій. Модуль PineAP у WiFi Pineapple відповідає на probe requests для множини SSID одночасно, що є ненормальною поведінкою. Легітимна точка доступу відповідає лише на запити для власного SSID. SDR-система або спеціалізований WIDS, що аналізує probe response кадри, може виявити пристрій, що видає себе за десятки різних мереж.

Наступним методом є виявлення рогових точок (Rogue AP) за потужністю сигналу. Якщо фальшива точка доступу розташована ближче до клієнтів, ніж легітимна, вона матиме сильніший сигнал. Клієнтські пристрої часто віддають перевагу підключенню до AP з кращим сигналом. Моніторинг рівнів сигналу від усіх точок доступу з однаковим SSID та виявлення нової AP з аномально сильним сигналом може вказувати на атаку.

При атаках MITM зловмисник часто не може надати валідний SSL-сертифікат для HTTPS-з’єднань та використовує самопідписані сертифікати або сертифікати з невірним Common Name. В даному випадку доцільним є аналіз SSL/TLS сертифікатів. Клієнтські пристрої або проксі-сервери безпеки повинні відслідковувати такі аномалії та блокувати підозрілі з’єднання.

Для ефективного виявлення втручання WiFi Pineapple є моніторинг мережевої активності. Передбачається, що WiFi Pineapple, виступаючи як проксі, створює специфічні патерни трафіку – подвійне NAT, аномальні затримки пакетів, незвичайні маршрути. Системи NDR (Network Detection and Response) можуть виявляти такі аномалії у поведінці мережі.

5.3.4 Методи виявлення WiFi Deauther

Нормальна робота мережі передбачає рідкісні події деавтентифікації (коли клієнт коректно відключається або втрачає зв’язок через віддалення). Різке зростання кількості деавтентифікаційних кадрів (особливо десятки або сотні за секунду) є явним індикатором атаки. WIDS повинен мати налаштовані пороги для виявлення таких аномалій. Даний метод має назву «Моніторинг частоти деавтентифікаційних кадрів»

Під час атаки деавтентифікації MAC-адреса відправника у кадрі підробляється (зазвичай копіюється адреса легітимної точки доступу). Однак, SDR-система з можливістю RF fingerprinting може виявити, що радіочастотні

характеристики сигналу (особливості модуляції, частотні зміщення, фазовий шум) не відповідають відомому обладнанню. Це дозволяє ідентифікувати підроблені кадри навіть при правильній MAC-адресі. Даний метод має назву «Аналіз джерела деавтентифікаційних кадрів».

Використовуючи розподілену мережу радіосенсорів (на базі SDR або спеціалізованих WIDS), можна визначити місцезнаходження WiFi Deauther методом тріангуляції за рівнями отриманого сигналу. Це дозволяє фізично локалізувати та вилучити пристрій.

Якщо системи моніторингу фіксують одночасне відключення множини клієнтів без очевидних причин (наприклад, відсутність проблем з точкою доступу, нормальні рівні сигналу до моменту відключення), це вказує на зовнішню атаку деавтентифікації.

Деякі реалізації WiFi Deauther мають характерні особливості – наприклад, надсилають кадри у специфічній послідовності, використовують певні поля заголовків або мають передбачувані інтервали між пакетами. Створення сигнатур для відомих варіантів WiFi Deauther дозволяє виявляти їх більш надійно.

5.3.5 Методи протидії та захисту

Для захисту від кібератак на бездротові засоби зв’язку рекомендується застосовувати наступні організаційно-технічні заходи:

– жорстка сегментація мереж;

– політика дозволених пристроїв (Whitelist);

– навчання персоналу;

– фізична безпека.

Критично важливі сегменти АСУТП не повинні мати бездротового доступу взагалі. Якщо бездротовий зв’язок необхідний, слід створювати ізольовані VLAN для бездротових клієнтів з жорсткими правилами міжмережевого екранування та IPS для інспекції трафіку між бездротовим та промисловим сегментами.

Впровадження суворої політики контролю доступу на основі MAC-адрес (802.1X з EAP-TLS та сертифікатами) гарантує, що до мережі зможуть підключитися лише попередньо авторизовані пристрої. Це ускладнює атаки, оскільки зловмисник повинен буде спочатку скомпрометувати легітимний пристрій.

Інженери та оператори повинні бути проінформовані про ризики підключення до невідомих або підозрілих Wi-Fi мереж. Політика безпеки повинна забороняти автоматичне підключення до мереж та вимагати перевірки сертифікатів при підключенні до корпоративних ресурсів.

Регулярні інспекції приміщень на предмет виявлення підозрілих пристроїв, контроль доступу до технічних зон та відеоспостереження знижують ризик залишення WiFi Pineapple або Deauther на об'єкті.

Також, слід впроваджувати і криптографічні заходи:

– впровадження WPA3;

– впровадження стандарту 802.11w;

– взаємна автентифікація (EAP-TLS);

– застосування VPN для критичного трафіку.

Стандарт WPA3 використовує більш стійкий протокол автентифікації SAE (Simultaneous Authentication of Equals), що робить атаку перехоплення handshake неефективною. Крім того, WPA3 забезпечує forward secrecy – навіть якщо пароль мережі буде зламаний у майбутньому, раніше перехоплений трафік залишиться незрозумілим.

Впровадження стандарту 802.11w забезпечує шифрування та автентифікацію керуючих кадрів, включаючи деавтентифікацію. Це робить атаки WiFi Deauther неможливими, оскільки підроблені незашифровані кадри будуть відкинуті приймачем. Важливо: 802.11w повинен бути обов'язковим (required), а не опціональним.

Використання сертифікатів для взаємної автентифікації клієнта та точки доступу (замість паролів) робить атаки «Evil Twin» значно складнішими. Клієнт перевірятиме сертифікат точки доступу перед підключенням, а WiFi Pineapple не зможе надати валідний сертифікат, підписаний довіреним центром сертифікації організації.

Навіть якщо зловмисник зможе змусити клієнта підключитися до фальшивої точки доступу, використання VPN-тунелю з сильним шифруванням (IPsec, WireGuard) для передачі критичних даних гарантує конфіденційність та цілісність інформації незалежно від безпеки бездротового каналу.

5.4 Застосування технології SDR для моніторингу радіочастотного середовища з метою виявлення кібератак

5.4.1 Реалізація програмно-апаратної системи радіомоніторингу на базі SDR

Комплексна система моніторингу радіочастотного середовища АСУТП на базі SDR включає три основні компоненти: розподілені радіосенсори, центральний сервер обробки та аналітики, консоль управління безпекою (рис. 5.8) [15].

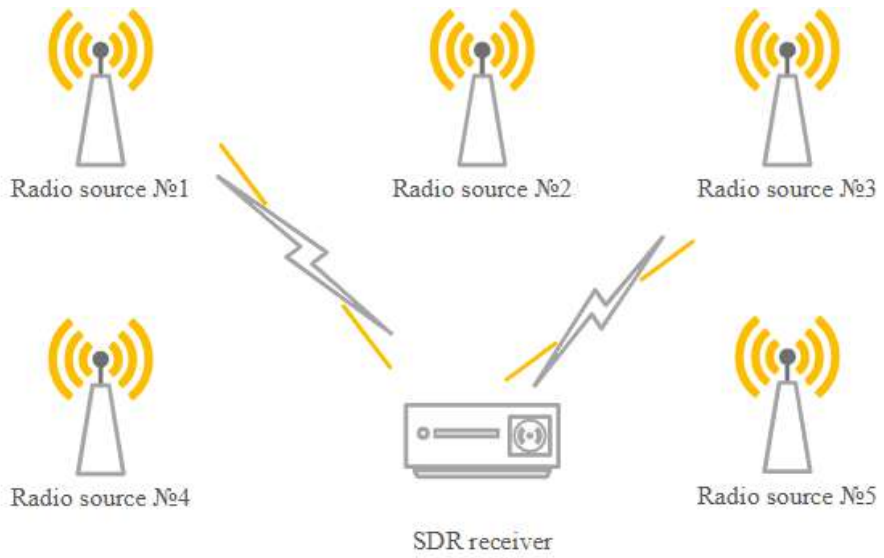


Рисунок 5.8 – Комплексна система моніторингу радіочастотного середовища з кількома частотними діапазонами

Радіосенсори є розподіленою мережею SDR-пристроїв, розміщених у стратегічно важливих точках промислового об’єкта. Кожен сенсор здійснює безперервний моніторинг призначених йому частотних діапазонів, оцифровує радіосигнали та виконує первинну обробку (фільтрацію, демодуляцію, виділення пакетів). Залежно від обчислювальних потужностей сенсора, він може здійснювати або базову обробку з передачею IQ-відліків на сервер, або повний аналіз локально з передачею лише метаданих та виявлених аномалій [16].

Сервер обробки агрегує дані з усіх радіосенсорів, здійснює декодування промислових протоколів (WirelessHART, ISA100.11a, Modbus RTU over Radio тощо), ведення бази даних легітимних пристроїв та їх характеристик, виявлення аномалій за допомогою сигнатурного аналізу та машинного навчання,

локалізацію джерел загроз методом тріангуляції на основі рівнів сигналу з різних сенсорів, та кореляцію подій з різних частотних діапазонів для виявлення складних координованих атак.

Консоль управління безпекою надає операторам візуалізацію стану радіочастотного середовища (спектрограми, карти покриття, топологія виявлених мереж), інформацію про виявлені загрози з класифікацією за рівнем критичності, інструменти для розслідування інцидентів (перегляд записаних IQ-відліків, декодованих пакетів, аналіз часових залежностей), та автоматизовані сценарії реагування (ізоляція скомпрометованого сегмента, оповіщення персоналу, запуск резервних каналів зв'язку) [16].

5.4.2 Апаратне забезпечення для проведення моніторингу радіочастото середовища

Одним із найбільш універсальних інструментів цього класу є апаратна платформа *HackRF One* виробництва Great Scott Gadgets (рис. 5.9).

Рисунок 5.9 – Зовнішній вигляд пристрою HackRF One

HackRF One – це напівдуплексна SDR-периферія з відкритим вихідним кодом, придатна для проєктування, тестування та верифікації сучасних радіотехнологій. Пристрій може функціонувати як зовнішній USB-модуль під керуванням хост-комп'ютера або як автономна система завдяки можливості програмування вбудованого мікроконтролера.

Архітектура HackRF One забезпечує гнучкість налаштувань у широкому діапазоні значень, що є критично важливим для виявлення аномалій у складних радіочастотних середовищах:

– діапазон робочих частот від 1 МГц до 6 ГГц, що охоплює більшість промислових стандартів зв'язку (від субгігагерцових датчиків до Wi-Fi 5/6 та 5G діапазонів);

– частота дискретизації до 20 млн вибірок на секунду (Msps);

– використання 8-бітних квадратурних вибірок (I та Q компоненти);

– підключення через високошвидкісний інтерфейс USB 2.0;

– наявність SMA-роз'ємів для входу/виходу тактового сигналу, що дозволяє синхронізувати кілька пристроїв для побудови фазованих антенних решіток;

– програмне керування коефіцієнтом підсилення RX/TX, фільтрами основної смуги, а також функцією живлення активної антени (до 50 мА за напруги 3,3 В).

Завдяки відкритій архітектурі, HackRF One має повну інтеграцію з провідним програмним забезпеченням для обробки сигналів:

– GNU Radio для розробки складних ланцюгів обробки та демодуляції;

– SDR# (SDRSharp) для візуального спектрального аналізу в реальному часі;

– спеціалізоване ПЗ для тестування на проникнення та аудиту бездротових мереж.

Ubertooth One – це спеціалізоване апаратне рішення з відкритим вихідним кодом, розроблене для дослідження та аудиту протоколів сімейства Bluetooth (рис. 5.10).

Ключовою технічною перевагою Ubertooth One, що вирізняє її серед стандартних адаптерів Bluetooth, є здатність працювати в режимі моніторингу (promiscuous mode). На відміну від типових комерційних контролерів, які ігнорують пакети, не адресовані безпосередньо їм, Ubertooth забезпечує:

– перехоплення трафіку в реальному часі. Реалізована можливість пасивного спостереження за обміном даними між сторонніми пристроями;

– повноцінна робота в спектрі 2,4 ГГц із підтримкою аналізу протоколів Bluetooth Classic та Bluetooth Low Energy (BLE);

– можливість не лише прийому, а й передавання сигналів, що дозволяє використовувати пристрій для тестування систем на вразливість.

Рисунок 5.10 – Спеціалізоване апаратне рішення Ubertooth One SDR

Впровадження Ubertooth One дозволить масштабувати методики аудиту безпеки, зокрема:

– проводити ідентифікацію прихованих пристроїв, які не транслюють свою присутність у звичайному режимі;

– виконувати аналіз стрибкоподібної перебудови робочої частоти (FHSS), що використовується в Bluetooth для захисту від завад та перехоплення;

– проводити розробку та верифікацію засобів захисту для бездротових вузлів у промислових мережах та системах «розумного дому».

KiwiSDR забезпечує прийом радіосигналів у смузі від 10 кГц до 30 МГц, охоплюючи наддовгі, довгі та короткі хвилі, а також AM-діапазони мовлення, сигнали комунальних служб і аматорські радіопередачі, що робить його придатним для глобального спостереження за радіочастотним середовищем. Апаратна реалізація KiwiSDR виконана у вигляді спеціалізованої друкованої плати, яка використовується спільно з одноплатними комп’ютерами BeagleBone Green або BeagleBone Black і доповнюється зовнішньою антеною, джерелом живлення та мережевим підключенням (рис. 5.11). Програмне забезпечення системи постачається у вигляді готового образу на картці microSD, що суттєво спрощує розгортання та експлуатацію приймача.

Функціонування KiwiSDR передбачає віддалений доступ користувачів через веббраузер із підтримкою HTML5, що дозволяє здійснювати прийом і аналіз сигналів з будь-якої точки світу за наявності інтернет-з’єднання.

Рисунок 5.11 – Пристрій KiwiSDR

Архітектура системи підтримує одночасну роботу до чотирьох користувачів, при цьому кожен із них отримує незалежний віртуальний канал приймача з можливістю налаштування частоти в межах усього доступного спектра. Візуалізація сигналів реалізована за допомогою спектрального водоспаду, який налаштовується незалежно від аудіоканалу та підтримує масштабування і панорамування, що підвищує інформативність аналізу радіочастотної обстановки.

З технічної точки зору KiwiSDR використовує багатоканальну паралельну цифрову обробку сигналів із реалізацією цифрового пониження частоти на основі CIC-фільтрів, оптимізованих за розрядністю, що забезпечує ефективну обробку широкого спектра сигналів за відносно невисоких апаратних витратах. Особливої уваги заслуговує висока чутливість і стабільність роботи приймача в діапазонах VLF і LF, що робить його ефективним засобом для моніторингу низькочастотних радіосигналів. Додатковою перевагою системи є автоматичне калібрування частоти з використанням сигналів точного часу GPS, завдяки чому досягається висока точність частотної прив’язки прийнятих сигналів. Архітектура KiwiSDR також передбачає інтерфейс розширення, який дозволяє інтегрувати додаткові програмні декодери та утиліти, розширюючи функціональні можливості приймача відповідно до конкретних дослідницьких або прикладних завдань.

RTL-SDR V4 є розвитком добре відомої лінійки приймачів на базі чипів Realtek RTL2832U та високочастотних тюнерів, які спочатку проєктувалися для цифрового телебачення, але згодом отримали широке застосування у сфері радіомоніторингу, досліджень і навчання завдяки можливості прямого доступу до сирих відліків сигналу. Основною особливістю версії V4 є суттєво вдосконалений радіочастотний тракт, що забезпечує розширення робочого діапазону та підвищення стабільності й чутливості прийому порівняно з попередніми поколіннями. На рис. 5.12 подано зовнішній вигляд пристрою.

Рисунок 5.12 – Пристрій RTL-SDR V4

RTL-SDR V4 підтримує прийом сигналів у діапазоні від наднизьких частот до приблизно 1,7 ГГц, що дозволяє використовувати його для моніторингу широкого спектра радіосистем, включно з радіомовленням, авіаційним і морським зв’язком, аматорськими діапазонами, телеметричними системами, бездротовими сенсорними мережами та низкою промислових протоколів. Для забезпечення прийому в області VLF і LF у даній версії реалізовано вдосконалене пряме підключення до аналого-цифрового перетворювача, що знижує рівень шумів і розширює функціональні можливості пристрою в задачах низькочастотного моніторингу. Покращений генератор тактової частоти та оптимізоване живлення сприяють зменшенню дрейфу частоти, що є критично важливим для спектрального аналізу та довготривалих спостережень.

З архітектурної точки зору RTL-SDR V4 поєднує мінімалістичну апаратну реалізацію з широкими можливостями програмної обробки. Пристрій виконує лише базові функції радіочастотного прийому та оцифрування сигналу, тоді як усі операції демодуляції, фільтрації, декодування та аналізу реалізуються

програмно на стороні персонального комп'ютера або вбудованої обчислювальної системи. Такий підхід повністю відповідає ідеології програмно визначеного радіо і дозволяє використовувати RTL-SDR V4 з різноманітним програмним забезпеченням, зокрема GNU Radio, SDR#, GQRX та іншими спеціалізованими інструментами, що значно розширює спектр можливих сценаріїв застосування.

З практичної точки зору RTL-SDR V4 широко застосовується як універсальний інструмент радіочастотного моніторингу, аналізу електромагнітної обстановки та експериментальних досліджень. Його низька вартість у поєднанні з достатньою для багатьох завдань чутливістю робить пристрій привабливим для навчальних лабораторій, наукових досліджень і прототипування систем виявлення радіозагроз. У контексті кібербезпеки та захисту АСУТП RTL-SDR V4 може використовуватися як сенсорний елемент розподіленої системи радіочастотного моніторингу для виявлення несанкціонованих передавачів, аналізу завад, контролю зайнятості спектра та фіксації аномальної радіоактивності в промисловому середовищі.

5.4.3 Методи виявлення кіберзагроз за допомогою програмно-апаратної системи SDR

Система на базі SDR реалізує багаторівневий підхід до виявлення кіберзагроз.

Спектральний аналіз здійснює моніторинг енергетичного розподілу сигналів у частотній області. Виявлення аномалій може включати несподівані піки потужності у частотних діапазонах промислових протоколів (індикатор атаки глушіння), сигнали у частотних каналах, що не використовуються легітимним обладнанням, аномальні характеристики модуляції, що відрізняються від очікуваних для відомих протоколів, та періодичні або імпульсні завади з характеристиками, що вказують на навмисне походження.

Аналіз протоколів включає декодування пакетів промислових протоколів та виявлення підозрілої активності:

– появу нових MAC-адрес пристроїв, що не зареєстровані у базі легітимного обладнання, команди керування від несанкціонованих джерел;

– аномалії у послідовності пакетів (пропуски, дублювання, порушення timing);

– спроби автентифікації з невірними криптографічними параметрами;

– передачу даних з аномальними значеннями (такими, що виходять за технологічно обґрунтовані межі).

Поведінковий аналіз використовує методи машинного навчання для виявлення відхилень від нормальної поведінки мережі:

– аномалії у профілях трафіку (раптові зміни інтенсивності, розмірів пакетів, міжпакетних інтервалів);

– порушення типових топологічних зв'язків між пристроями mesh-мережі, незвичайні часові патерни активності (передача даних у неробочий час);

– кореляція подій з різних сенсорів, що вказує на координовані дії.

Фізичний аналіз на основі характеристик радіосигналу включає ідентифікацію пристроїв за унікальними особливостями їхніх радіочастотних випромінювань (RF fingerprinting), виявлення клонованих пристроїв (однаковий MAC, різні RF fingerprints), локалізацію джерел загроз за рівнями сигналу з декількох точок спостереження, та аналіз багатопроменевого поширення для виявлення відбитих сигналів від об'єктів, що рухаються (можливе проникнення порушників).

З появою нового радіочастотного пристрою у заданому діапазоні, що передає дані з ідентифікаторами, схожими на легітимні датчики системи автоматизації, система декодує пакети, виявляє невідповідність формату даних очікуваному протоколу, ідентифікує спробу підміни даних телеметрії, блокує прийняття даних від підозрілого пристрою на рівні контролерів, та зберігає IQ-відліки сигналу для подальшого криміналістичного аналізу.

5.5 Практична реалізація методу виявлення незареєстрованого передавача LoRa WAN

5.5.1 Опис лабораторного макету, що використовується для проведення експериментального дослідження

Розглянемо приклад практичної реалізації методу виявлення незареєстрованого передавача LoRa WAN у лабораторних умовах, який базується на використанні програмно визначеного радіоприймача як сенсора радіочастотного середовища та програмних засобів спектрального і часово-частотного аналізу сигналів у діапазоні 868 МГц з метою фіксації, ідентифікації та подальшої класифікації радіоактивності, що не відповідає параметрам легітимних вузлів бездротової мережі.

Для забезпечення кібербезпеки АСУТП та промислових бездротових мереж задача RTL-SDR V4 полягає у демонстрації принципової можливості виявлення несанкціонованих або прихованих LoRa-передавачів без доступу до їхнього протокольного рівня. Оскільки багато загроз пов'язані не з порушенням криптографії, а з фактом наявності самого передавача в контрольованій зоні, RTL-SDR V4 дозволяє показати, що вже на фізичному рівні можна зафіксувати появу небажаної радіоактивності, оцінити її параметри та сформувати сигнал тривоги для системи безпеки.

Лабораторний макет складається з наступних пристроїв:

– модуль сканування раідочастотного середовища SDR RTL Blog V4;

– модуль імітації незареєстрованого передавача LoRa WAN TTGO V1.3;

– модуль імітації незареєстрованого приймача пакетів LoRa WAN TTGO V1.3;

– телескопічна антена для налаштування на заданий діапазон радіосигналів.

Зовнішній вигляд лабораторного макету подано на рис. 5.13.

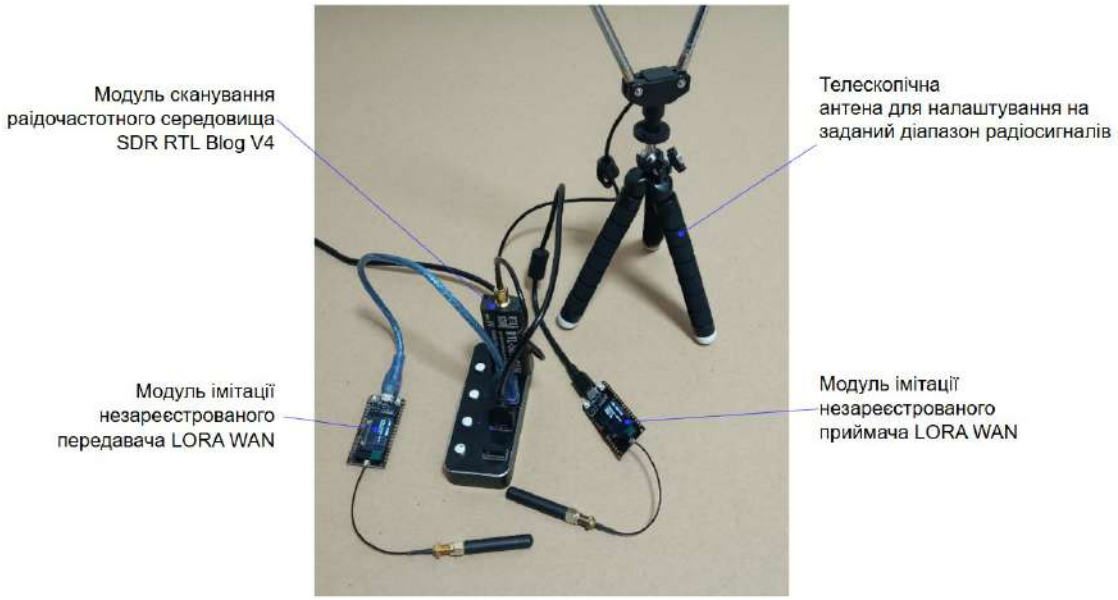


Рисунок 5.13 – Зовнішній вигляд лабораторного макету

Модуль SDR RTL Blog V4 є основою лабораторного макету виявлення несанкціонованого радіопередавача LoRa в діапазоні 868 МГц. Його роль полягає у виконанні функції універсального приймального сенсора радіочастотного середовища, який забезпечує оцифрування сигналів ефіру та

передавання їх на програмний рівень для подальшого аналізу, ідентифікації та прийняття рішень щодо наявності потенційної загрози.

У складі даного макету сам по собі RTL-SDR V4 не є повноцінною системою виявлення, а виступає ключовим апаратним елементом, що реалізує принцип програмно визначеного радіо і дозволяє перенести основну логіку виявлення несанкціонованих передавачів у програмне забезпечення.

RTL-SDR V4 призначений для безперервного або періодичного приймання сигналів у діапазоні 868 МГц, який активно використовується в промислових IoT-рішеннях, бездротових сенсорних мережах, а також у технології LoRa. На цьому етапі ключовою функцією модуля є фіксація широкосмугових IQ-даних із радіоефіру з належною часовою та частотною роздільною здатністю. Це забезпечує можливість подальшого програмного аналізу з метою виявлення характерних часових і спектральних особливостей сигналів LoRa, навіть за умов їх низької потужності або нерегулярної передачі. Таким чином, RTL-SDR V4 виконує функцію своєрідного «радіоока» системи, забезпечуючи надходження повної, первинно необробленої інформації про стан радіочастотного середовища.

Модуль забезпечує можливість експериментальної перевірки алгоритмів виявлення, таких як аналіз зайнятості спектра, виявлення сигналів із розширеним спектром, аналіз часових інтервалів передавання та порівняння параметрів реального сигналу з еталонними характеристиками дозволених LoRa-пристроїв.

Іншою складовою лабораторного макету виступають два модуля імітації незареєстрованого передавача та приймача LoRa WAN, що побудовані на основі пристрою LILYGO ESP32 з LoRa модемом SX1276 868 МГц TTGO V1.3 (рис. 5.14). Інтегрований модуль LILYGO TTGO LoRa V1.3 є обчислювальною платформою для розроблення систем промислового Інтернету речей, що поєднує в собі мікроконтролер ESP32 та можливості енергоефективного радіозв’язку великого радіуса дії за технологією LoRa. Апаратна частина пристрою базується на двоядерному процесорі з тактовою частотою до 240 МГц, оснащеному 4 Мб флешпам’яті та вбудованими стеками Wi-Fi і Bluetooth, що забезпечує гнучку інтеграцію в гетерогенні мережі.

Радіочастотна частина реалізована на базі трансивера Semtech SX1276, що працює в неліцензованому діапазоні 868 МГц. Завдяки високій чутливості до -148 дБм та вихідній потужності передавача до +20 дБм, модуль демонструє

виняткову стійкість до завад та здатність передавати дані на значні відстані в умовах щільної забудови або складного рельєфу.

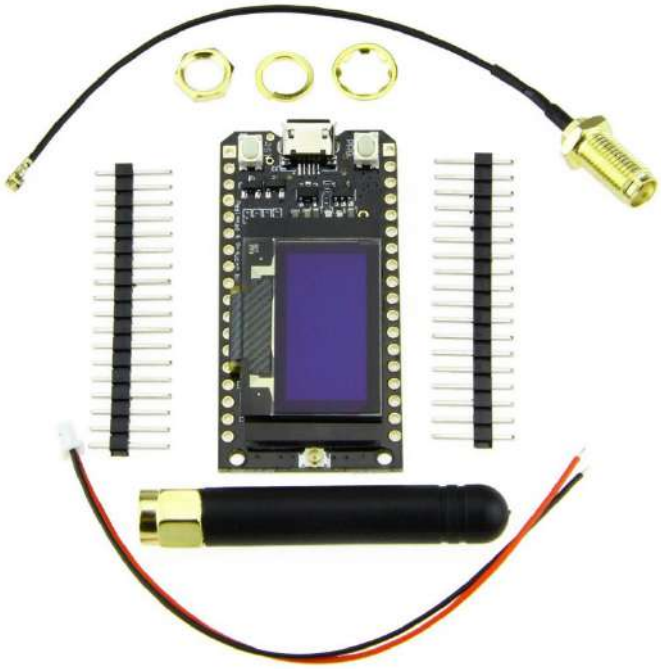


Рисунок 5.14 – Пристрій LILYGO ESP32 з LoRa модемом SX1276 868 МГц TTGO V1.3

Для візуалізації системних процесів та взаємодії з користувачем на платі інтегровано OLED-дисплей діагоналлю 0,96 дюйма. Енергосистема модуля оптимізована для автономної роботи: вона включає схему підключення та зарядки літієвого акумулятора, а також інтерфейс для резервного живлення, при цьому робочий діапазон напруги становить від 3,3 В до 7 В. Програмування та налагодження пристрою здійснюються через високошвидкісний інтерфейс USB-UART на базі конвертера 9102F, що забезпечує повну сумісність із середовищем розробки Arduino IDE та дозволяє швидко прототипувати додатки для моніторингу віддалених об’єктів або управління технологічними процесами.

Один з модулів виконує роль передавача, а інший приймача пакетів повідомлення. Розглянемо приклад коду програми для реалізації передавача:

```
#include <SPI.h>
#include <LoRa.h>
#include <Wire.h>
#include "SSD1306.h"
#include "images.h"
```

```
#define SCK     5    // GPIO5  -- SX1278's SCK
#define MISO    19   // GPIO19 -- SX1278's MISO
#define MOSI    27   // GPIO27 -- SX1278's MOSI
#define SS      18   // GPIO18 -- SX1278's CS
#define RST     14   // GPIO14 -- SX1278's RESET
#define DI0     26   // GPIO26 -- SX1278's IRQ(Interrupt Request)
#define BAND    868E6

unsigned int counter = 0;

SSD1306 display(0x3c, 4, 15);
String rssi = "RSSI --";
String packSize = "--";
String packet ;

void logo(){
  display.clear();
  display.drawXbm(0,5,logo_width,logo_height,logo_bits);
  display.display();
}

void setup() {
  pinMode(16,OUTPUT);
  pinMode(2,OUTPUT);

  digitalWrite(16, LOW);    // set GPIO16 low to reset OLED
  delay(50);
  digitalWrite(16, HIGH); // while OLED is running, must set GPIO16 in high

  Serial.begin(9600);
  while (!Serial);
  Serial.println();
  Serial.println("LoRa Sender Test");

  SPI.begin(SCK,MISO,MOSI,SS);
  LoRa.setPins(SS,RST,DI0);
  if (!LoRa.begin(868E6)) {
    Serial.println("Starting LoRa failed!");
    while (1);
  }
  //LoRa.onReceive(cbk);
//  LoRa.receive();
  Serial.println("init ok");
```

```
  display.init();
  display.flipScreenVertically();
  display.setFont(ArialMT_Plain_10);
  logo();
  delay(1500);
}

void loop() {
  display.clear();
  display.setTextAlignment(TEXT_ALIGN_LEFT);
  display.setFont(ArialMT_Plain_10);

  display.drawString(0, 0, "Sending packet: ");
  display.drawString(90, 0, String(counter));
  display.display();

  // send packet
  LoRa.beginPacket();
  LoRa.print("hello ");
  LoRa.print(counter);
  LoRa.endPacket();

  counter++;
  digitalWrite(2, HIGH);   // turn the LED on (HIGH is the voltage level)
  delay(1000);                       // wait for a second
  digitalWrite(2, LOW);    // turn the LED off by making the voltage LOW
  delay(1000);                       // wait for a second
}
```

Принцип роботи програми полягає в періодичному формуванні та передаванні тестових LoRa-пакетів у діапазоні 868 МГц із паралельною індикацією стану передавання на дисплеї та світлодіоді, що дозволяє наочно контролювати активність радіопередавача.

На етапі ініціалізації визначаються апаратні з’єднання між мікроконтролером і радіомодулем SX1278, зокрема лінії SPI, сигнал вибору пристрою, скидання та переривання, а також задається робоча частота 868 МГц, що відповідає європейському ISM-діапазону, характерному для мереж LoRa WAN. Додатково ініціалізується OLED-дисплей, який підключено через інтерфейс I²C, та задаються допоміжні змінні для зберігання лічильника переданих пакетів і текстових даних для відображення.

Функція setup призначена для початкового налаштування апаратної частини. Встановлюються режими роботи GPIO-виводів, один із яких використовується для апаратного скидання OLED-дисплея, а інший – для керування світлодіодом індикації. Після цього запускається послідовний інтерфейс для налагодження та виведення діагностичних повідомлень. Далі ініціалізується SPI-інтерфейс з явно заданими виводами, після чого налаштовуються керуючі сигнали радіомодуля LoRa і виконується його запуск на заданій частоті. В разі помилки ініціалізації, робота програми зупиняється, що дозволяє однозначно виявити проблему під час налагодження. Після успішного запуску радіомодуля, ініціалізується дисплей, встановлюється орієнтація зображення, обирається шрифт і короткочасно відображається графічний логотип, що слугує візуальним підтвердженням коректного старту системи.

Основна логіка роботи реалізована у функції loop, яка виконується циклічно. На кожній ітерації цикл починається з очищення дисплея та виведення текстової інформації про поточний номер переданого пакета, що дозволяє спостерігати за процесом передавання без використання зовнішніх засобів моніторингу. Після оновлення дисплея формується LoRa-пакет, до якого послідовно додається текстове повідомлення та значення лічильника. Завершення формування пакета автоматично ініціює його передавання в ефір. Таким чином, кожен переданий пакет є унікальним завдяки змінному числовому значенню, що спрощує його ідентифікацію на приймальній стороні або під час радіочастотного аналізу.

Після передавання пакета лічильник збільшується на одиницю, а світлодіод вмикається та вимикається з паузою в одну секунду, що виконує роль простої світлової індикації активності передавача. Затримки між передаваннями задають періодичний режим роботи, у якому LoRa-пакети випромінюються приблизно раз на дві секунди.

В результаті дана програма формує джерело LoRa-сигналу, яке використовується в макеті як тестовий передавач під час відпрацювання методів виявлення несанкціонованих LoRa-передавачів та аналізу радіочастотного середовища в діапазоні 868 МГц.

Приклад роботи LoRa-передавача подано на рис. 5.15.

Приймач також побудовано на аналогічному модулі LILYGO TTGO LoRa V1.3. Його роль – відображення прийнятих пакетів для контроля працездатності лабораторного макету.

Рисунок 5.15 – Приклад роботи LoRa-передавача

Поданий програмний код реалізує роботу приймального вузла LoRa на базі мікроконтролера з підключеним радіомодулем SX1278 та OLED-дисплеєм на контролері SSD1306 і призначений для прийому, обробки та візуалізації отриманих LoRa-пакетів у діапазоні 868 МГц:

```
#include <SPI.h>
#include <LoRa.h>
#include <Wire.h>
#include "SSD1306.h"
#include "images.h"

#define SCK     5    // GPIO5  -- SX1278's SCK
#define MISO    19   // GPIO19 -- SX1278's MISO
#define MOSI    27   // GPIO27 -- SX1278's MOSI
#define SS      18   // GPIO18 -- SX1278's CS
#define RST     14   // GPIO14 -- SX1278's RESET
#define DI0     26   // GPIO26 -- SX1278's IRQ(Interrupt Request)
#define BAND    868E6

SSD1306 display(0x3c, 4, 15);
String rssi = "RSSI --";
String packSize = "--";
String packet ;

void loraData(){
  display.clear();
  display.setTextAlignment(TEXT_ALIGN_LEFT);
  display.setFont(ArialMT_Plain_10);
  display.drawString(0 , 15 , "Received "+ packSize + " bytes");
  display.drawStringMaxWidth(0 , 26 , 128, packet);
  display.drawString(0, 0, rssi);
```

```
  display.display();
}

void cbk(int packetSize) {
  packet ="";
  packSize = String(packetSize,DEC);
  for (int i = 0; i < packetSize; i++) { packet += (char) LoRa.read(); }
  rssi = "RSSI " + String(LoRa.packetRssi(), DEC) ;
  loraData();
}

void setup() {
  pinMode(16,OUTPUT);
  digitalWrite(16, LOW);    // set GPIO16 low to reset OLED
  delay(50);
  digitalWrite(16, HIGH); // while OLED is running, must set GPIO16 in high
  Serial.begin(9600);
  while (!Serial);
  Serial.println();
  Serial.println("LoRa Receiver Callback");
  SPI.begin(SCK,MISO,MOSI,SS);
  LoRa.setPins(SS,RST,DI0);
  if (!LoRa.begin(868E6)) {
    Serial.println("Starting LoRa failed!");
    while (1);
  }
  //LoRa.onReceive(cbk);
  LoRa.receive();
  Serial.println("init ok");
  display.init();
  display.flipScreenVertically();
  display.setFont(ArialMT_Plain_10);

  delay(1500);
}

void loop() {
  int packetSize = LoRa.parsePacket();
  if (packetSize) { cbk(packetSize);  }
  delay(10);
}
```

Принцип роботи програми полягає в безперервному моніторингу радіоканалу, виявленні вхідних пакетів, зчитуванні їхнього вмісту та відображенні ключових параметрів прийнятого сигналу, що дозволяє використовувати даний код як основу приймального модуля у лабораторних дослідженнях і системах радіочастотного моніторингу.

На початковому етапі виконання програми визначаються апаратні з'єднання між мікроконтролером і радіомодулем SX1278, зокрема лінії інтерфейсу SPI, сигнали керування скиданням і перериванням, а також встановлюється робоча частота 868 МГц, яка відповідає європейському ISM-діапазону та широко використовується в системах LoRa WAN. Паралельно конфігурується OLED-дисплей, який підключено через інтерфейс I^2C, і оголошуються змінні для зберігання рівня прийнятого сигналу, розміру пакета та його корисного навантаження.

У функції setup здійснюється ініціалізація апаратної частини системи. Спочатку виконується апаратне скидання дисплея шляхом короткочасної подачі низького логічного рівня на відповідний вивід, після чого запускається послідовний інтерфейс для виведення діагностичних повідомлень. Далі ініціалізується SPI-інтерфейс із явно заданими виводами, налаштовуються керуючі сигнали радіомодуля і виконується його запуск у режимі прийому на заданій частоті. У разі помилки ініціалізації виконання програми зупиняється, що дозволяє оперативно виявити проблеми під час налагодження. Після успішного запуску радіомодуля система переводиться в режим постійного прийому LoRa-пакетів, а дисплей ініціалізується, налаштовується орієнтація зображення та шрифт для подальшого відображення інформації.

Основна логіка роботи реалізована у функції loop, де з коротким інтервалом опитується радіомодуль на наявність прийнятих пакетів. Функція аналізу вхідного потоку визначає розмір пакета, і у випадку його наявності викликається спеціальна обробна функція. У цій функції послідовно зчитується вміст прийнятого пакета з буфера радіомодуля, формується текстове представлення корисного навантаження, фіксується розмір пакета, а також визначається значення рівня прийнятого сигналу RSSI, яке є важливим параметром для оцінки потужності та віддаленості передавача. Отримані дані використовуються для оновлення інформації на OLED-дисплеї, де відображаються рівень сигналу, кількість прийнятих байтів і текст повідомлення.

Приклад роботи LoRa-приймача подано на рис. 5.16.

Рисунок 5.16 – Приклад роботи LoRa-приймача

5.5.2 Принцип організації обміну повідомленнями в мережі IoT за допомогою модулів LoRa

IoT-пристрої дозволяють в режимі реального часу спостерігати за роботою виробничих ліній, виявляти проблеми, отримувати інформацію про необхідні профілактичні заходи та обслуговуванні. Щоб об'єднані в мережу пристрою працювали ефективно і генерували потрібну для аналітики інформацію, підприємство повинно забезпечити зв'язаність своїх операцій і машин. Тобто операційні технології повинні функціонувати узгоджено з інформаційними, а обладнання повинно бути підключено до людино-машинного інтерфейсу, щоб фахівці могли працювати з інформацією.

В промисловості використовується різноманітне обладнання, яке потребує окремі канали зв’язку для обміну інформацією. В залежності від типу обладнання та сфери застосовування мережею передається різний обсяг даних. В залежності від типу мережі використовуються спеціальне обладнання. В мережах з високим навантаженням та великим обсягом даних використовуються шлюзи, роутери, маршрутизатори.

Шлюзи в мережах IoT забезпечують підключення пристроїв і аналітику даних, що надходять до пристроїв IoT, які, як правило, не мають цих можливостей. Будь-який шлюз може використовувати IoT модулі для виконання аналізу або попередньої обробки перед передачею повідомлень від підлеглих пристроїв в центр Інтернету речей.

В промисловості використовується різноманітне обладнання, яке потребує окремі канали зв’язку для обміну інформацією. В залежності від типу

обладнання та сфери застосовування, мережею передається різний обсяг даних [17]. В таблиці 5.2 подано приклади використання бездротових мереж різними програмними засобами.

Таблиця 5.2 – Класифікація бездротових мереж за типом повідомлень, що передаються

Тип повідомлення, що передається	**Обсяг інформації, що передається**	**Приклад застосування**
Потік даних, потокове відео	Великий (Мегабайти)	Управління технологічним обладнанням в реальному часі, моніторинг стану технологічного процесу, інтелектуальні системи відеоспостереження
JSON повідомлення, текстові дані	Середній (Кілобайти)	Охоронні системи, система розумного будинку
Фрагменти кадру повідомлення, стан обладнання	Малий (байти)	Системи обліку споживаних енергоресурсів, системи охоронної сигналізації, система розумного будинку

Модель захисту автоматизованої системи від комп'ютерних атак це абстрактний (формалізований або неформалізований) опис комплексу програмно-технічних засобів або організаційних заходів захисту (виявлення, протидія, усунення наслідків) від комп'ютерних атак. У загальному вигляді модель процесу захисту може бути представлена наступним чином (рис. 5.17).

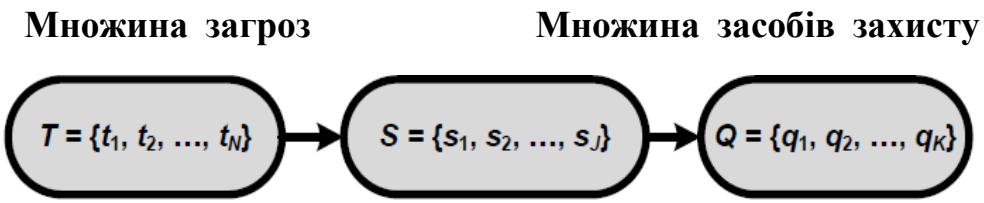


Рисунок 5.17 – Загальна модель процесу захисту від комп'ютерних атак

Шлюзи в мережах IoT забезпечують безпечне підключення пристроїв і аналітику даних, що надходять до пристроїв IoT, які, як правило, не мають цих можливостей.

Існує три шаблони використання пристрою IoT як шлюз: прозорий шлюз, шлюз перетворення протоколів та шлюз перетворення посвідчень.

Основна відмінність між шаблонами полягає в тому, що прозорий шлюз передає повідомлення між підлеглими пристроями і центром Інтернету речей, не вимагаючи додаткової обробки. Однак перетворення протоколу і перетворення фрагментів підтвердження вимагає обробки цих даних шлюзом для забезпечення взаємодії між пристроями.

Будь-який шлюз може використовувати IoT модулі для виконання аналізу або попередньої обробки перед передачею повідомлень від підлеглих пристроїв в центр Інтернету речей.

У шаблоні прозорого шлюзу пристрої, які теоретично можуть підключатися до центру Інтернету речей, можуть підключатися до пристрою шлюзу. Підлеглі пристрої мають власні сертифікати Центру Інтернету речей і використовують будь-який з протоколів MQTT, AMQP або HTTP. Шлюз просто забезпечує взаємодію між пристроями і Центром Інтернету речей.

У LoRa виділяють 3 класи пристроїв за енергоспоживанням:

А-клас – найекономніший. Пристрої цього класу живляться від батареї декілька років, що досягається за рахунок активності пристрою тільки під час передачі даних за програмованим розкладом.

С-клас, навпаки, постійно знаходиться у стані прийому. Саме тому в пристроях класу С не передбачається живлення від батарейок.

В-клас так само, як і А-клас, більшу частину часу перебуває в режимі енергозбереження, але при цьому має деякі можливості для взаємодії з сервером, що притаманні для С-класу.

LoRaWAN використовує неліцензовану частину радіочастотного спектру в діапазоні 868,0 – 868,6 МГц (в Україні). Стандарт передбачає наявність базових станцій і абонентських пристроїв, які, за умови автономного живлення, більшу частину часу перебувають в режимі збереження енергії. Пристрої «прокидаються» лише для обміну даними з сервером.

Одна з головних задач при проектуванні шлюзу промислової мережі – це оптимізація часу передавання пакетів від сенсорів до мережного сенсору та отримання зворотної відповіді.

Для моделювання роботи шлюзу необхідно визначитися с принципом передачі пакетів. Від зміни параметрів модуля, що відповідають за режими передачі та прийому залежить загальний час передачі та споживана потужність.

Змінюючи ці параметри можна обрати оптимальний режим роботи модуля LoRa для вирішення поставленої задачі та налаштувати модуль на мінімальний рівень споживаної енергії, що забезпечить максимальний час роботи пристрою від автономного живлення.

Для організації обміну повідомленнями на фізичному рівні забезпечується передача блоків даних між кінцевим пристроєм (End Node) і шлюзом LoRa (Gateway).

З боку передавального пристрою виконуються такі послідовні дії:

– прийом блоку даних від верхнього апаратного рівня (PHYPayload);

– формування фізичного заголовка пакета (PHDR + PHDR_CRC);

– кодування фізичного заголовка пакета (PHDR + PHDR_CRC) з фіксованою швидкістю 4/8;

– обчислення контрольної суми блоку корисних даних PHYPayload (CRC);

– кодування блоку корисних даних (PHYPayload + CRC) з попередньо встановленою швидкістю CR;

– передача радіоканалом преамбули;

– модуляція і передача радіоканалом фізичного блоку даних.

З боку приймального пристрою виконується:

– виявлення преамбули і визначення початку фізичного блоку даних;

– демодуляція сигналу;

– декодування фізичного заголовка пакета (PHDR + PHDR_CRC) і перевірка його контрольної суми;

– декодування блоку корисних даних (PHYPayload + CRC) і перевірка його контрольної суми;

– підтвердження прийнятих даних (для відповідних типів повідомлень);

– передача даних на верхній рівень модуля для подальшого використання кінцевими пристроями.

Загальний вигляд пакету LoRa подано на рис. 5.18 та складається з трьох елементів:

– преамбула;

– необов’язковий заголовок;

– корисне навантаження даних.

Преамбула використовується для синхронізації приймача з потоком вхідних даних. За замовчуванням пакет налаштований на послідовність 12 символів. Це програмована змінна, тому довжина преамбули може бути

збільшена, наприклад, з метою зменшення робочого циклу приймача в інтенсивних програмах прийому. Мінімальної довжини вистачає для будь-якого спілкування. Довжину преамбули, що передається, можна змінити, встановивши регістр PreambleLength від 6 до 65535, отримуючи загальну довжину преамбули від 6 + 4 до 65535 + 4 символів. Це дозволяє передавати майже довільно довгу послідовність преамбули.

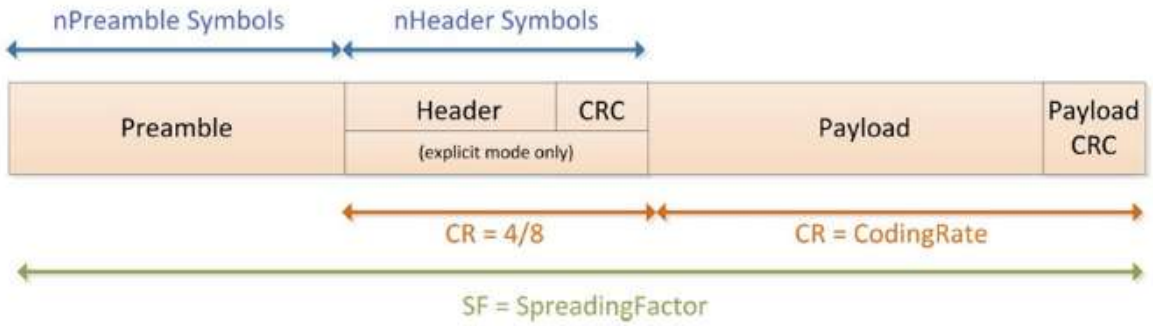


Рисунок 5.18 – Структура пакету даних пакета LoRa

Приймач здійснює процес виявлення преамбули, який періодично перезапускається. З цієї причини довжина преамбули повинна бути налаштована ідентично довжині преамбули передавача. Якщо довжина преамбули невідома або може змінюватися, максимальна довжина преамбули повинна бути запрограмована на стороні приймача.

Залежно від обраного режиму роботи модуля LoRa доступні два типи заголовків:

– explicit mode;

– implicit mode.

Тип заголовка обирається бітом ImplicitHeaderModeOn, який знаходиться у реєстрі RegModemConfig1. На рис. 5.19 подано приклад пакету даних в режимі explicit mode.

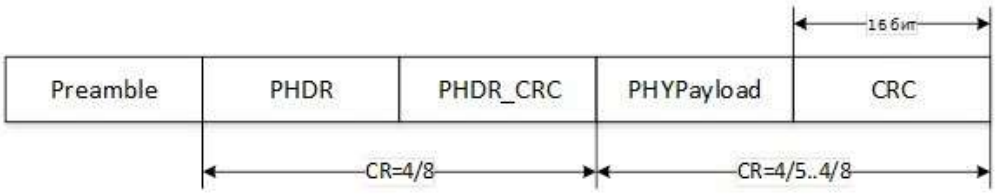


Рисунок 5.19 – Формат повідомлення explicit mode

На рис. 5.19 Preamble – це преамбула, яка використовується для синхронізації приймача з вхідним потоком і визначення початку фізичного блоку даних. Довжина преамбули для SX-1278 є програмованою величиною.

PHDR – фізичний заголовок пакета. Присутній тільки при використанні explicit mode і містить:

– довжина поля даних (Payload) в байтах,

– частота виправлення кодів помилок,

– наявність додаткового 16-бітового CRC для поля даних.

У певних сценаріях, коли поле даних, швидкість кодування та наявність CRC фіксовані або відомі заздалегідь, є сенс зменшити час передачі, викликаючи неявний режим заголовка. В даному режимі заголовок видаляється з пакета. У цьому випадку довжина поля даних, швидкість кодування помилок і наявність CRC корисного навантаження повинні бути налаштовані вручну з обох сторін радіолінії.

PHDR_CRC – контрольна сума поля PHDR.

PHYPayload – корисне навантаження.

CRC – контрольна сума поля PHYPayload (опціональне поле).

При цьому заголовок PHDR кодується надлишковим кодом з фіксованою швидкістю 4/8, а корисне навантаження – з програмованою швидкістю. На рис. 5.20 подано приклад пакету даних в implicit mode.

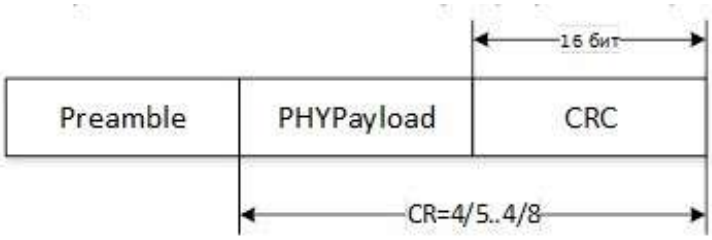


Рисунок 5.20 – Формат повідомлення implicit mode

Під час використання implicit mode фізичний заголовок пакета не передається і пристрої працюють з попередньо встановленими параметрами.

На рис. 5.21 подано випадок, коли поле контрольна сума (CRC) відсутнє.

Технологія LoRa використовує асинхронний режим прийому-передачі, в якому передавач може почати генерацію радіосигналу в будь-який момент часу [17]. В даному випадку існує механізм, що забезпечує синхронізацію приймача за сигналом від передавача. В якості такого механізму

використовується преамбула, що передує кожному сеансу зв'язку. Преамбула включає в себе послідовність символів, які дозволяють приймачу виявити активність передавача, визначити використовуваний передавачем коефіцієнт розширення спектра (SF) і виконати символьну синхронізацію. Тривалість преамбули можна змінювати, але вона не повинна бути менше, ніж

$$\tau_p = T_1 + 2T_2, \tag{1}$$

де T_1 – визначає максимальний час знаходження приймача в стані Sleep,

T_2 – визначає час пошуку приймачем преамбули.

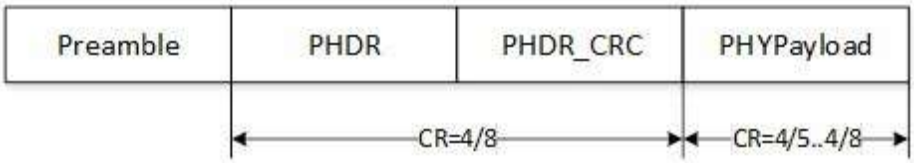


Рисунок 5.21 – Формат повідомлення з відсутнім полем CRC

Принцип передачі символів інформації блоку даних фізичного рівня за допомогою широкосмугового радіосигналу LoRa полягає в частотному зсуві

$$e^{j\cdot\Delta\omega\cdot k\cdot t}$$

відносно опорного сигналу

$$e^{j\cdot(\omega_n\cdot t+\mu\cdot t^2)},$$

де k = 0,1,2, ..., 2SF – інформаційний символ, розмірністю SF біт.

Таким чином, функція x(t) запишеться наступним чином:

$$x(t) = \begin{cases} A_0 \cdot cos\left(\omega_н \cdot t + \Delta\omega \cdot k \cdot t + \frac{\mu}{2} \cdot t^2\right), & 0 \le t < T_0 \\ A_0 \cdot cos\left(\omega_н \cdot t + \Delta\omega \cdot k \cdot t - BW \cdot t + \frac{\mu}{2} \cdot t^2\right), & T_0 \le t < T_{sym} \end{cases}$$

де BW – ширина спектра радіосигналу;

k = 0,1,2, ..., 2SF – інформаційний символ, розмірністю SF біт;

T_{sym} = 2SF/BW – тривалість радіосигналу;

μ=BW/T_{sym} – швидкість зміни частоти радіосигналу;

t – час передавання одиниці даних;

$\omega_н$ – частота радіосигналу.

Приклад залежності частоти радіосигналу від часу для кадру даних показаний показано на рис. 5.22.

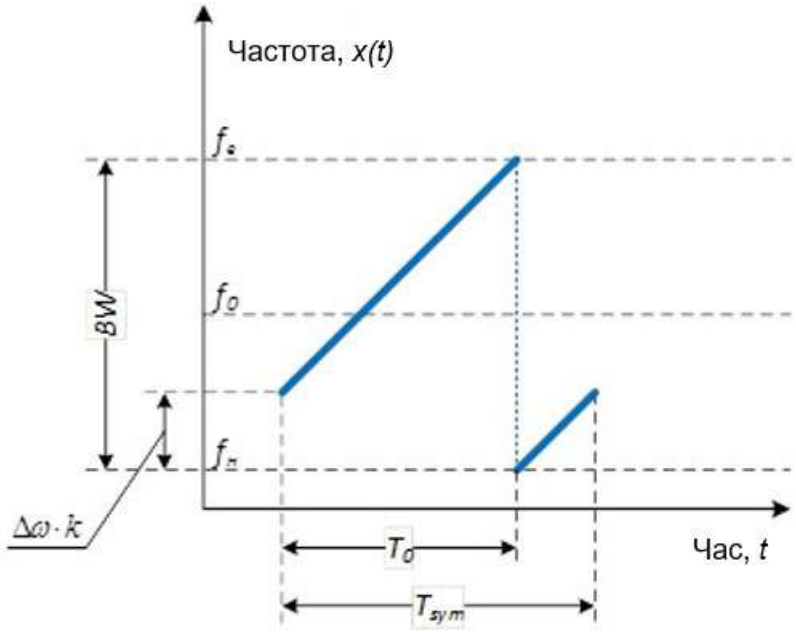


Рисунок 5.22 – Приклад залежності частоти радіосигналу від часу для кадру даних

5.5.3 Комп'ютерне моделювання та вибір оптимальних параметрів шлюзу

Моделювання роботи шлюзу проводилося за допомогою програмного засобу LoRa Modem Calculator від фірми Semtech. Вхідними умовами для моделювання є наступні параметри:

– довжина поля даних;

– довжина поля преамбули;

– ширина спектра радіосигналу BW;

– коефіцієнт розповсюдження (фактор розширення спектра);

– частотний діапазон;

– потужність передатчика;

– формат пакету даних.

В результаті моделювання необхідно визначити наступні умови експлуатації:

– час для передачі пакету даних;

– швидкість передачі даних;

– чутливість приймача;

– потужність, що необхідна для передачі даних;

– термін роботи від автономного джерела живлення.

Початкові умови, що є незмінними:

– інтервал передачі одна секунда;

– ємність батареї 1000 мА·год.;

– напруга живлення 3,3 В;

– робоча частота 433 мГц;

– розмір Payload може змінюватись в межах 7-3 байти.

Таким чином, проведено моделювання роботи пристрою для цих параметрів. Результати моделювання подано у таблиці 5.3.

Таблиця 5.3 – Результати моделювання роботи модуля прийому /передачі даних при різних розмірах поля Payload

Payload, байт	7	6	5	4	3
SF = 12, BW = 125 кГц, CR = 4/5, Header Enabled, Preamble = 10,25 симв., Payload = 7 байт, TR Power = 14 дБ					
Час в ефірі, мс	925,7	761,86	761,86	761,86	761,86
Час передачі симв, мс	32,77	32,77	32,77	32,77	32,77
Споживаний струм в процесі передавання, мА	44	44	44	44	44
Чутливість приймача, дБ	-138	-138	-138	-138	-138
Орієнтований час роботи від батареї, діб	80,54 1,02	80,54 1,24	80,54 1,24	80,54 1,24	80,54 1,24
CAD, мс	61,1	61,1	61,1	61,1	61,1

На рис. 5.23 подано результат моделювання в програмі LoRa Modem Calculator для режиму роботи Periodic Receiver.

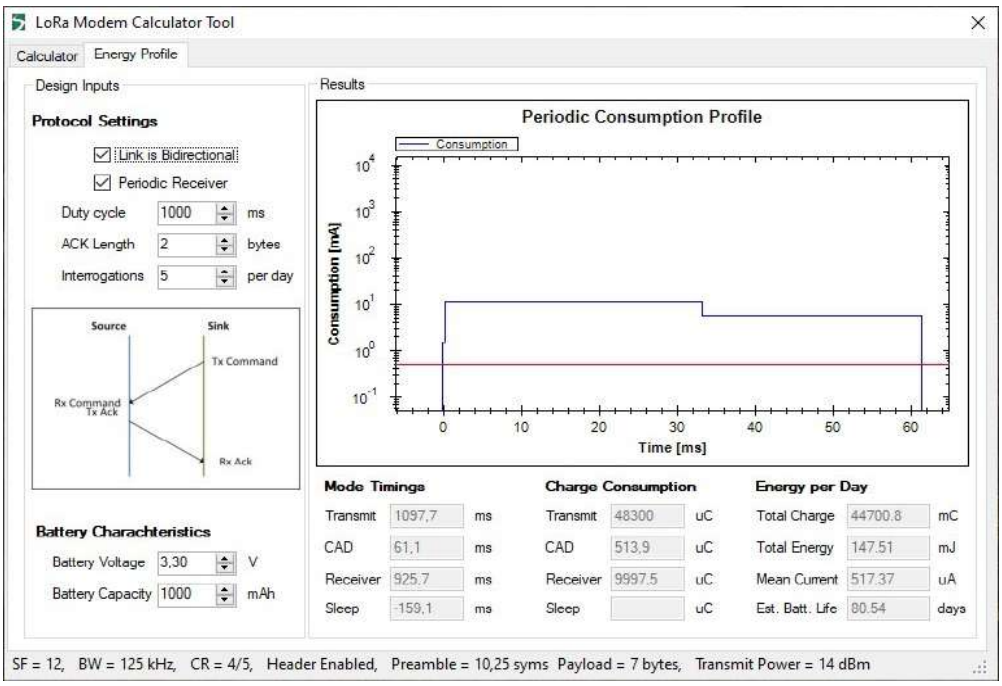


Рисунок 5.23 – Результат моделювання в програмі LoRa Modem Calculator для режиму роботи Periodic Receiver

На рис. 5.24 подано результат моделювання в програмі LoRa Modem Calculator для режиму роботи Periodic Transmitter

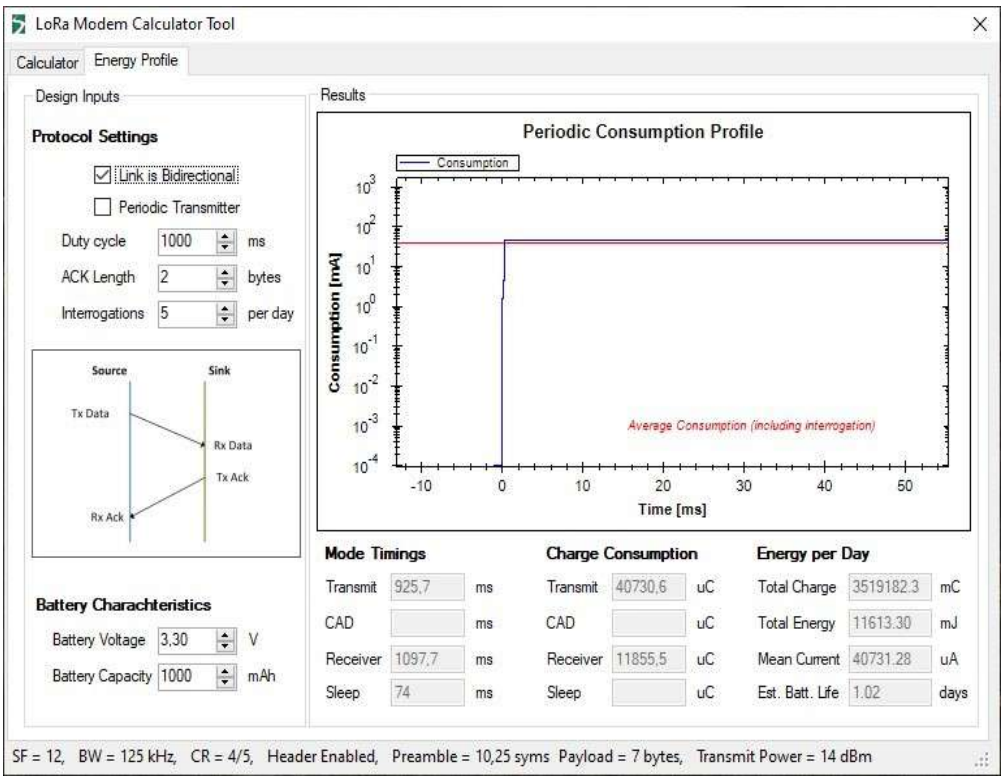


Рисунок 5.24 – Результат моделювання в програмі LoRa Modem Calculator для режиму роботи Periodic Transmitter

Виходячи з аналізу протоколу обміну даними можна зробити висновок, що для зниження споживаної потужності та зменшення часу в ефірі можна зробити змінну довжину кадру. Наприклад, для контролю наявності пристрою потрібно лише 4 байти (прибрати з протоколу порожні поля даних та адресу комірки). Як можна бачити з таблиці 1 зі зменшенням довжини поля даних зменшується час знаходження в ефірі з 925,7 до 761,86 мс. При цьому споживаний струм не зменшується. Таким чином, потрібно знайти інші параметри, що дадуть нам можливість подовжити час автономної роботи.

Іншим параметром, яким можна змінювати споживані властивості модуля є наявність або відсутність полів заголовку та контрольної суми. Визначено вплив цих параметрів для двох значень розміру поля даних – 7 та 4 байти.

Результати моделювання роботи модуля прийому/передачі даних при різних розмірах кадру представлені в таблиці 5.4.

Таблиця 5.4 – Результати моделювання роботи модуля прийому/передачі даних для різних розмірів кадру

Payload, байт	7	7	7	4	4	4
Header Enabled	Так	Ні	Ні	Так	Ні	Ні
CRC Enabled	Так	Так	Ні	Так	Так	Ні
SF = 12, BW = 125 кГц, CR = 4/5, Header Enabled, Preamble = 10,25 симв., Payload = 7 байт, TR Power = 14 дБ						
Час в ефірі, мс	925, 7	761,86	761,86	761,86	761,86	598,02
Час передачі симв, мс	32,77	32,77	32,77	32,77	32,77	32,77
Споживаний струм в процесі передавання, мА	44	44	44	44	44	44
Чутливість приймача, дБ	-138	-138	-138	-138	-138	-138
Орієнтований час роботи від батареї, діб	80,54 1,02	80,6 1,24	80,6 1,24	80,54 1,24	80,6 1,24	80,6 1,58
CAD, мс	61,1	61,1	61,1	61,1	61,1	61,1

Аналізуючи отримані результати можна бачити, що при відключенні одного з полів Header або CRC вдалось трохи збільшити автономний час роботи від батареї з 80,54 до 80,6 діб.

Для передавача при відключенні зазначених полів кадру вдалось збільшити час автономної роботи з 1,02 до 1,24 доби для 7 байт даних, та з 1,24

до 1,58 для 4 байт даних. Таким чином, приріст часу роботи передавача від батареї становить більше 20%.

Відключивши відразу два поля ми також зменшили загальний час знаходження в ефірі до 598,02 мс.

Щоб визначити присутній сигнал чи ні, замість використання індикатора потужності прийнятого сигналу (RSSI) в системі LoRa для ідентифікації присутності сигналу використовується комбінована адаптивна система виявлення активності каналу (Channel Activity Detection, CAD).

Вона може розрізняти шум і корисний сигнал LoRa. Процес функціонування цієї системи вимагає двох символів. Якщо система виявила сигнал, то переривання по CAD_Detected дасть підтвердження, і в цьому випадку, щоб отримати корисні дані, пристрій залишиться в режимі прийому.

На рис. 5.25 і рис. 5.26 подано результат моделювання роботи Channel Activity Detection. З рисунку можна бачити, що для режиму передачі цей параметр має нескінченно велике значення для заданих умов.

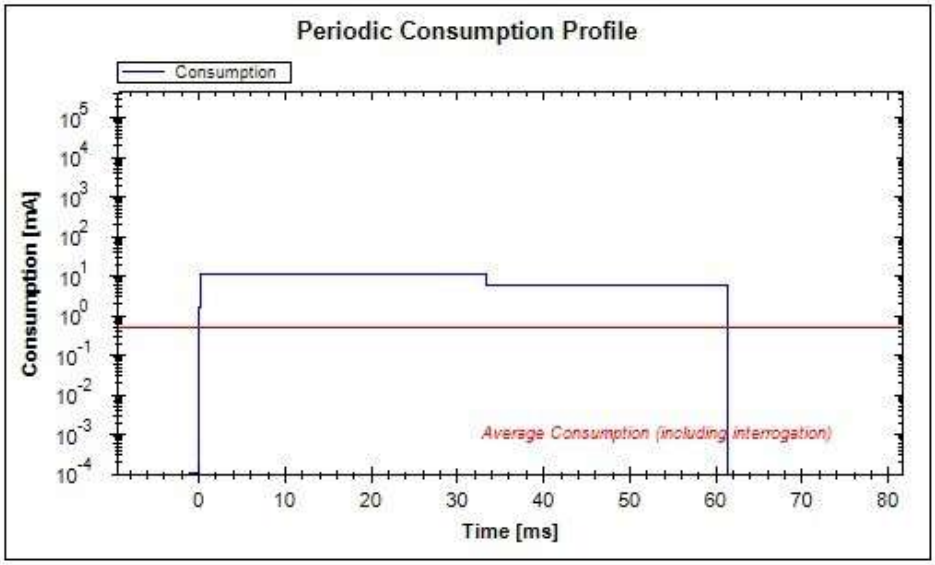


Рисунок 5.25 – Результат моделювання роботи Channel Activity Detection для приймача

Наступним параметром, який можна корегувати, є «Тип кодування (CR)». Програмісту при налаштуванні роботи пристрою доступні наступні значення цього параметру: CR = 4/5, 4/6, 4/7, 4/8. Результати моделювання подано в таблиці 5.5.

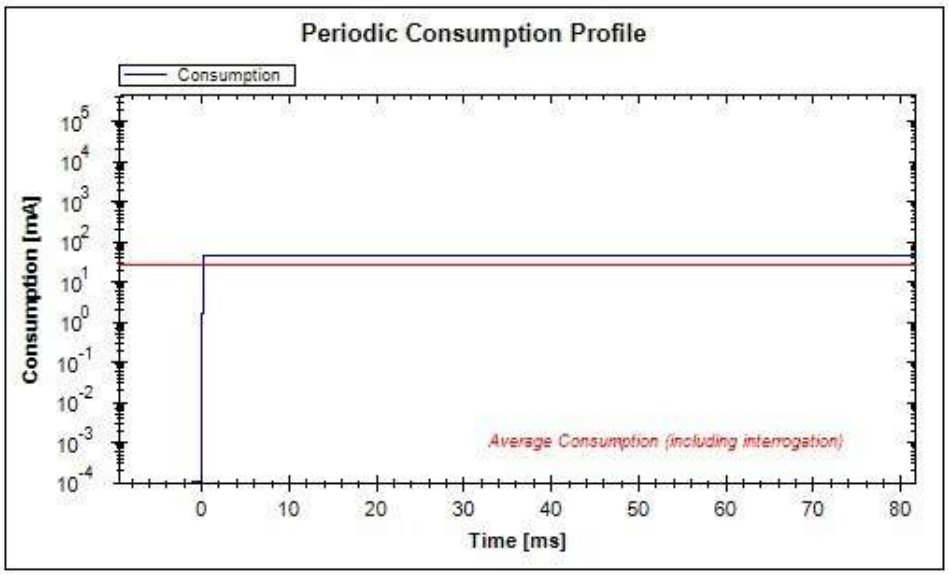


Рисунок 5.26 – Результат моделювання роботи Channel Activity Detection для передавача

Таблиця 5.5 – Результати моделювання роботи модуля прийому/передачі даних для різних значень параметру CR

Payload, байт	7	7	7	7	4	4	4	4
CR	4/5	4/6	4/7	4/8	4/5	4/6	4/7	4/8
SF = 12, BW = 125 кГц, CR = 4/5, Header Enabled, Preamble = 10,25 симв., Payload = 7 байт, TR Power = 14 дБ								
Час в ефірі, мс	925, 7	991,2	1057	1122	761,9	794,7	827,4	860,16
Час передачі симв, мс	32,7	32,7	32,7	32,7	32,7	32,77	32,77	32,77
Споживаний струм в процесі передавання, мА	44	44	44	44	44	44	44	44
Чутливість приймача, дБ	-138	-138	-138	-138	-138	-138	-138	-138
Орієнтований час роботи від батареї, діб	80,54 1,02	80,52 0,96	80,6 1,24	90.2 0,95	80,54 1,24	80,52 1,19	80,51 1,14	80,4 1,10
CAD, мс	61,1	61,1	61,1	61,1	61,1	61,1	61,1	61,1

За результатами експерименту можна бачити, що зміна параметру CR призводить до збільшення потрібного часу на передачу інформації і, відповідно, до зменшення часу автономної роботи модуля.

Ще одним висновком є те, що для значення параметру CR = 4/7, та CR = 4/8 ми виходимо за межі заданого значення інтервалу передачі – 1000 мс.

За вказаних значень параметру CR час знаходження в ефірі повинен становити 1057 та 1122 мс. Це суперечить початковим умовам, а значить не можна використовувати наведені параметри. Таким чином, для практичного використання залишаються два значення параметру CR = 4/5 і CR = 4/6.

Розглянуті параметри властиві для організації максимально можливої дальності передачі сигналу. Запатентована технологія організації зв'язку використовує свій спосіб модуляції, який має назву «чірп» – нелінійна модуляція за якою частота сигналу лінійно зростає від початкової частоти f0 до кінцевої f1. За стандартом LoRa кодування здійснюється шляхом циклічного зсуву чірпа щодо кадру часу.

Параметрами сигналу в LoRa є SF (Spreading factor) і Bandwidth (BW) – ширина смуги передачі. Параметр SF задається зумовленими значеннями SF7-SF12, де 7 найшвидший, а 12 – найповільніший режим. На рис. 5.27 подано приклад різних швидкостей передачі даних при різних значення SF.

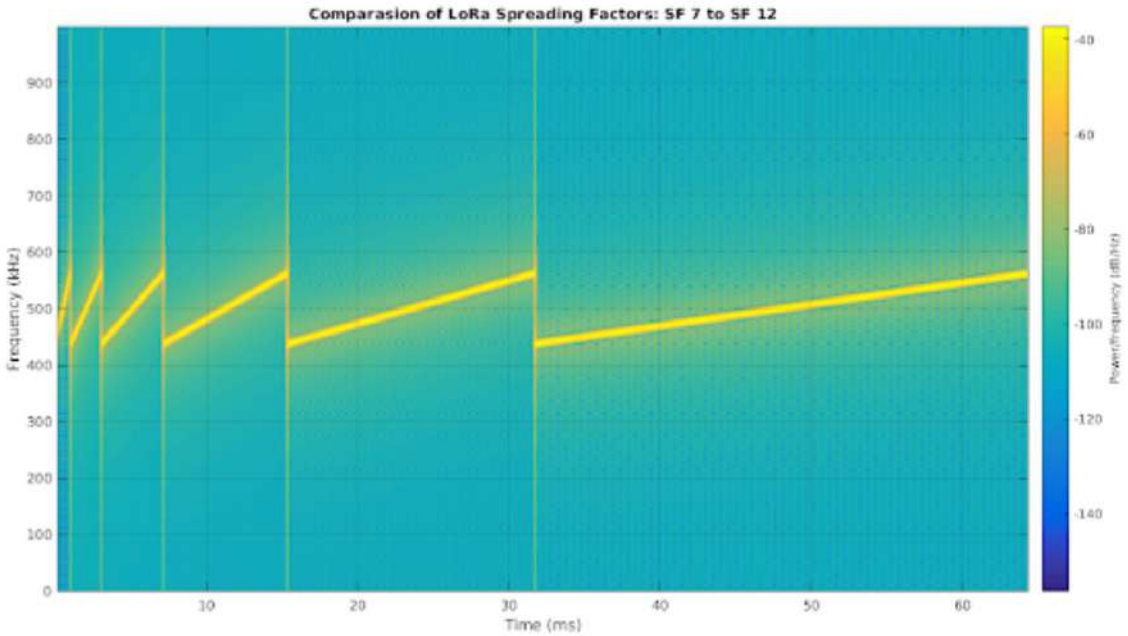


Рисунок 5.27 – Приклад залежності швидкості передачі даних від значень SF

Найповільніший сигнал відповідає найбільшій дальності передачі. Якщо дальність не є тим критерієм, якого треба досягти при проектуванні пристрою, то за документацією до модулю SX-1278 можна суттєво знизити споживану потужність змінюючи параметри SF та BW.

Виконаємо моделювання таких ситуацій за допомогою інструменту LoRa Modem Calculator. Параметр BW буде змінюватись наступним чином: 125 кГц, 250 кГц та 500 кГц. Результати моделювання занесені в таблицю 5.6.

Таблиця 5.6 – Результати моделювання роботи модуля прийому /передачі даних при різних значеннях параметру BW

BW, кГц	125	250	500
SF = 12, CR = 4/5, Header Enabled, Preamble = 10,25 симв., Payload = 7 байт, TR Power = 14 дБ			
Час в ефірі, мс	925, 7	462,85	231,42
Час передачі симв, мс	32,7	16,38	8,19
Споживаний струм в процесі передавання, мА	44	44	44
Чутливість приймача, дБ	-138	-135	-132
Орієнтований час роботи від батареї, діб	80,54 1,02	110,76 2,05	124,96 4,09
CAD, мс	61,1	44,6	36,3

Як можна бачити, розширення полоси пропускання значно впливає на скорочення часу передачі даних та приводить к зростанню загального часу автономної роботи. При максимальному значенні параметра BW = 500 кГц час автономної роботи збільшився в чотири рази для режиму передачі даних, та на 70% в режимі прийому.

Наступне моделювання проведемо для різних значень Spreading factor. Змінюватиме всі можливі значення цього параметру (SF = {6, 7, 8, 9, 10, 11, 12}). Результати моделювання роботи подано в таблиці 5.7.

Визначивши значення параметрів, за яких досягається максимальна енергоефективність, виконаємо моделювання такої ситуації за допомогою LoRa Modem Calculator. Моделювання проведено для двох значень розміру поля Payload (7 та 4 байти). Для першого випадку передачі пакету довжиною 7 байт використано такі параметри:

– Spreading factor SF = 6;

– розмір поля даних Payload = 7;

– поле заголовка відсутнє;

– поле контрольної суми відсутнє;

– тип кодування (CR) = 4/5.

Таблиця 5.7 – Результати моделювання роботи пристрою SX-1278 для різних значень параметру SF

SF	12	11	10	9	8	7	6
BW = 125 кГц, CR = 4/5, Header Enabled, Preamble = 10,25 симв., Payload = 7 байт, TR Power = 14 дБ							
Час в ефірі, мс	925, 7	462,8	231,42	115,7	68,1	34,05	17,02
Час передачі симв, мс	32,7	16,38	8,19	4,10	2,05	1,02	0,51
Споживаний струм в процесі передавання, мА	44	44	44	44	44	44	44
Чутливість приймача, дБ	-138	-135,5	-133	-130	-127	-124	-119
Орієнтований час роботи від батареї, діб	80,54 1,02	164,12 2,05	332,21 4,09	663,20 8,18	1288 13,90	2376 27,80	4041 55,57
CAD, мс	61,1	29,5	14,3	7	3,5	1,8	1

Результати моделювання покажемо для трьох значень ширини полоси пропускання 125, 250 та 500 кГц. Результати моделювання подано в таблиці 5.8.

Таблиця 5.8 – Результати моделювання для оптимальних значень параметрів модуля LoRa при Payload = 7 байт

BW, кГц	125	250	500
SF = 6, CR = 4/5, Header Disable, CRC Disable, Preamble = 10,25 симв., Payload = 7 байт, TR Power = 14 дБ			
Час в ефірі, мс	14,46	7,23	3,62
Час передачі символів, мс	0,51	0,26	0,13
Споживаний струм в процесі передавання, мА	44	44	44
Чутливість приймача, дБ	-119	-116	-113
Орієнтований час роботи від батареї, діб	4041,37 65,39	6269 130,64	8321,94 260,66
CAD, мс	1	0,6	0,4

Для моделювання передачі пакету довжиною 4 байти використані такі параметри:

– Spreading factor SF = 6;

– розмір поля даних Payload = 4;

– поле заголовка відсутнє;

– поле контрольної суми відсутнє;

– тип кодування (CR) = 4/5.

Результати моделювання для трьох значень ширини полоси пропускання 125, 250 та 500 кГц подано в таблиці 5.9.

Таблиця 5.9 – Результати моделювання для оптимальних значень параметрів модуля LoRa при Payload = 4 байт

BW, кГц	125	250	500
SF = 6, CR = 4/5, Header Disable, CRC Disable, Preamble = 10,25 симв., Payload = 4 байт, TR Power = 14 дБ			
Час в ефірі, мс	11, 9	5,95	2,98
Час передачі симв, мс	0,51	0,26	0,13
Споживаний струм в процесі передавання, мА	44	44	44
Чутливість приймача, дБ	-119	-116	-113
Орієнтований час роботи від батареї, діб	4041,37 79,44	6269 158,65	8321,94 316,40
CAD, мс	1	0,6	0,4

Для встановленого нами обмеження в цикл передачі даних на рівні 1000 мс час передавання повинен бути 10 мс.

5.5.4 Опис методу виявлення і візуальної ідентифікації LoRa-сигналу

Для виявлення і ідентифікації LoRa-сигналу в даному експерименті використовується програмне середовище SDR#. Інтерфейс програми подано на рис. 5.28.

Інтерфейс програми має наступні компоненти:

– бокове ліве меню з інтерфейсом доданих плагінів (A);

– кнопка виклику головного меню (1);

– кнопку запуску та зупинки роботи програми, яка відповідає за початок або завершення сеансу приймання сигналів (2);

– елемент відкриття нового сеансу або зрізу (slice), що дозволяє працювати з декількома незалежними приймальними каналами, починаючи з версії v.1741 та новіших оновлень (3);

– модуль конфігурації пристрою, призначений для налаштування параметрів апаратної частини та режимів роботи приймача (4);

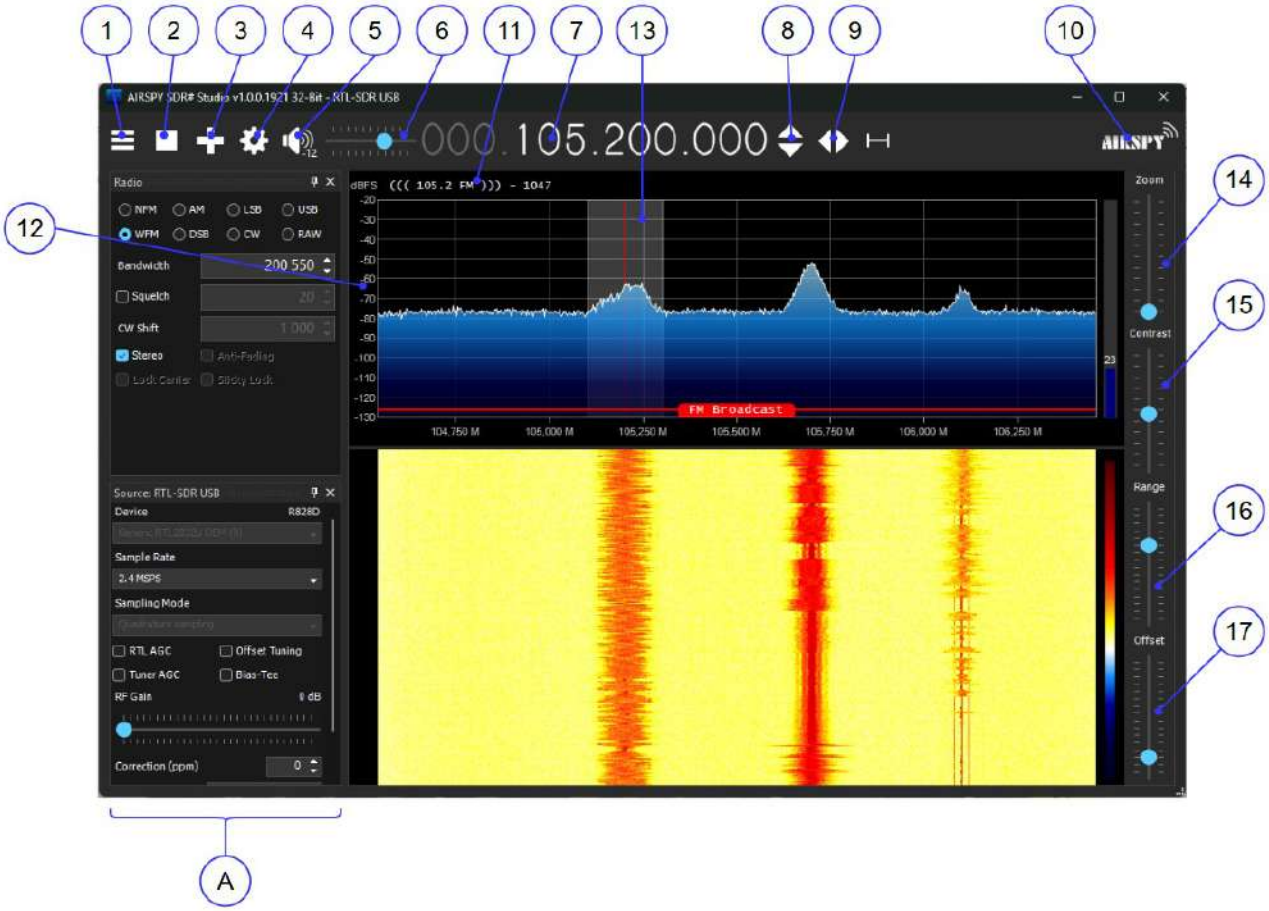


Рисунок 5.28 – Інтерфейс програми SDR#

– кнопку вмикання та вимикання звуку, що забезпечує миттєве приглушення аудіосигналу (mute) (5);

– повзунок регулювання гучності для плавного налаштування рівня вихідного аудіосигналу (6);

– поле введення та індикації частоти VFO, яке використовується для точного налаштування на потрібну частоту приймання (7);

– елемент вибору типу налаштування, що визначає режим роботи та спосіб керування частотою (8);

– модуль вибору кроку перебудови частоти, доступний починаючи з версії v.1782, який дозволяє задавати дискретність зміни частоти (9);

– логотип Airspy, при натисканні на який здійснюється перехід безпосередньо на офіційну домашню сторінку розробника (10);

– блок декодування RDS, що відображає інформацію PS, PI та RT для мовних станцій у діапазоні WFM (88…108 МГц) (11);

– шкалу рівня сигналу, подану в одиницях dBFS (децибели повної шкали), яка характеризує відносну потужність прийнятого сигналу (12);

– вертикальний індикатор спектра, що включає центральну червону лінію, а також інформацію про смугу пропускання і рівень сигналу (13);

– повзунок масштабування спектра радіочастот та водоспадного відображення, який дозволяє змінювати детальність огляду сигналів (14);

– повзунок регулювання контрасту для налаштування візуального сприйняття спектра та водоспаду (15);

– повзунок діапазону, що визначає динамічний діапазон сигналів, що відображається (16);

– повзунок зміщення, призначений для корекції положення спектра відносно шкали відображення (17).

Виявлення сигналу LoRa відбувається в певних діапазонах частот: 433 МГц, 470 МГц, 868 МГц та 920 МГц. Це стандартні частоти, але можуть бути і відхилення.

В даному експерименті скануватимо радіочастотний простір навколо до 868 МГц. Дану частоту беремо з вихідного коду імітатора передавача. Задача дослідника – знайти ознаки LoRa сигналу (рис. 5.29) [18].

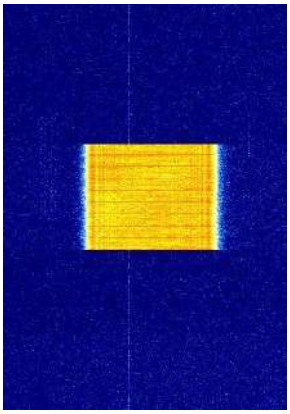


Рисунок 5.29 – Візуальні ознаки LoRa-сигналу

Візуальне зображення може відрізнятись від того, що показано на рис. 5.29. На відображення сигналу впливають такі параметри:

– Spreading Factor (SF);

– Bandwidth (BW);

– Coding Rate (CR);

– центральна частота.

Spreading Factor визначає тривалість кожного chirp-символу та, відповідно, швидкість передачі даних. Чим вищий SF, тим довший символ, тим більша

дальність зв'язку, але нижча швидкість. Візуально більш високий SF означає, що діагональні лінії chirps будуть ширшими (триваліші у часі), а при меншому SF – вужчими.

Bandwidth – ширина смуги частот, у якій відбувається частотне зміщення chirp. Типові значення: 125 кГц, 250 кГц, 500 кГц. Більша ширина смуги означає більший вертикальний розмір діагональних ліній на спектрограмі.

Coding Rate – коефіцієнт надлишкового кодування для виправлення помилок (4/5, 4/6, 4/7, 4/8). Впливає на загальну тривалість пакету, але не на візуальний вигляд окремих chirps.

Центральна частота – у Європі LoRa зазвичай працює у діапазоні 863-870 МГц (найчастіше 868 МГц), у США – 902-928 МГц (915 МГц), в Азії – 915-928 МГц або 433 МГц.

В залежності від кількості даних, що передається, висота блоку може як розширюватись, так і стискатися майже до горизонтальної лінії.

Для пошуку сигналу потрібно налаштувати програму SDR#. Центральну частоту виставляємо на 866 МГц, тип сигналу обираємо «NFM» (Narrowband FM) або «WFM» (Wideband FM), хоча для візуалізації тип модуляції не критичний – основна інформація буде у спектрограмі.

Параметри спектрограми (Waterfall) рекомендується обирати наступними:

– FFT Resolution 16384 або 32768 точок – вища роздільна здатність дозволяє краще бачити деталі chirp-структури;

– FFT Window Blackman-Harris для кращого співвідношення сигнал/шум;

– Waterfall speed середня або повільна, щоб мати час розгледіти структуру пакетів;

– Contrast/Range налаштовується так, щоб слабкі сигнали були помітні, але сильні не засвічували всю спектрограму;

– Gradient палітра обирається з високим контрастом (наприклад, «Hot» або «Rainbow») для кращого розрізнення сигналів різної інтенсивності.

Смуга перегляду (Filter bandwidth) встановлюється на 200-500 кГц, щоб захопити повну ширину LoRa-сигналу (125-250 кГц) з деяким запасом для компенсації частотних зміщень. RF Gain рекомендується встановити на відмітку 20-40 дБ залежно від умов – занадто високе підсилення призведе до перевантаження та інтермодуляції.

Дослідження радіоефіру дало змогу зафіксувати роботу імітатора передавача на частоті 866 МГц. На рис. 5.30 подано приклад виявленого радіосигнала від модуля імітації незареєстрованого передавача.

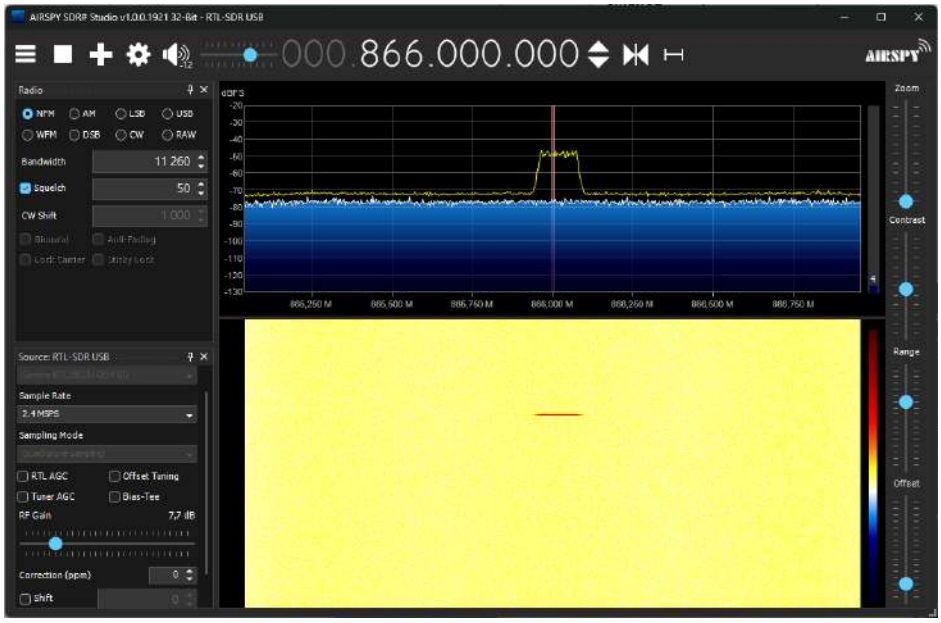


Рисунок 5.30 – Приклад виявленого радіосигнала від модуля імітації незареєстрованого передавача

На зображенні зі SDR# зафіксовано радіосигнал у смузі 866 МГц, що належить діапазону ISM EU 868 і типовий для мереж LoRa. Спектральна форма сигналу симетрична відносно центральної частоти та має характерний плоский спектральний профіль із чітко вираженими краями, що відповідає модуляції Chirp Spread Spectrum і дозволяє однозначно ідентифікувати сигнал як LoRa, а не FSK чи OOK.

За шкалою частот видно, що ефективна ширина смуги сигналу становить приблизно 140 кГц, при цьому реальна корисна смуга з урахуванням особливостей FFT та фільтрації приймача відповідає стандартному значенню LoRa з шириною каналу 125 кГц.

Рівень шумової підлоги знаходиться в межах –85…–90 dBFS, тоді як рівень корисного сигналу досягає приблизно –50…–55 dBFS, що забезпечує відношення сигнал/шум на рівні близько 30…35 дБ. Такий рівень SNR є достатнім для стабільного приймання і декодування LoRa-пакетів навіть за великих значень spreading factor.

Аналіз водоспадної спектрограми показує короткочасний широкосмуговий імпульс без чітко виражених діагональних структур, характерних для окремих чірпів. Це вказує на те, що зафіксовано короткий LoRa-пакет, а відсутність візуально помітної преамбули обумовлена або малою тривалістю сигналу, або параметрами відображення SDR# (FFT averaging, масштаб часу), які згладжують часову структуру чірпів. Така візуалізація є типовою для SDR# і не свідчить про дефект сигналу.

Точне визначення Spreading Factor на основі статичного спектрального зображення неможливе, однак з урахуванням короткої тривалості пакета та високої спектральної щільності енергії можна припустити використання малих або середніх значень SF, імовірно SF7 або SF8. Вищі значення SF, як правило, формують значно довші у часі пакети з більш помітною структурою на водоспаді.

Центральна частота сигналу чітко збігається з налаштованою частотою приймача, відсутні ознаки частотного зсуву, дрейфу або спектрального розмазування, що свідчить про хорошу частотну стабільність передавача та коректну роботу гетеродина приймача. Таким чином, параметри частотної стабільності не є обмежувальним фактором для декодування.

Для більш детального аналізу сигналу зробимо його запис, з подальшим відтворенням та масштабуванням.

Запис сигналу зробимо за допомогою інструменту «Baseband: Simple Recorder», що йде в складі пакету SDR# (рис. 5.31).

Simple Recorder у розділі Baseband в SDR# є інструментом для запису цифрових IQ-відліків (baseband samples) безпосередньо з виходу SDR-приймача. Це один з найважливіших інструментів для серйозного аналізу радіосигналів, особливо у контексті кібербезпеки та дослідження невідомих протоколів.

Baseband – це цифрове представлення радіосигналу після його перетворення з високої радіочастоти у низьку проміжну частоту або безпосередньо у цифрову форму. У SDR-системах це IQ-відліки (In-phase and Quadrature samples), які є комплексними числами, де:

– I (In-phase) – синфазна компонента сигналу;

– Q (Quadrature) – квадратурна компонента (зміщена на 90°).

Разом вони повністю описують амплітуду та фазу сигналу в кожний момент часу.

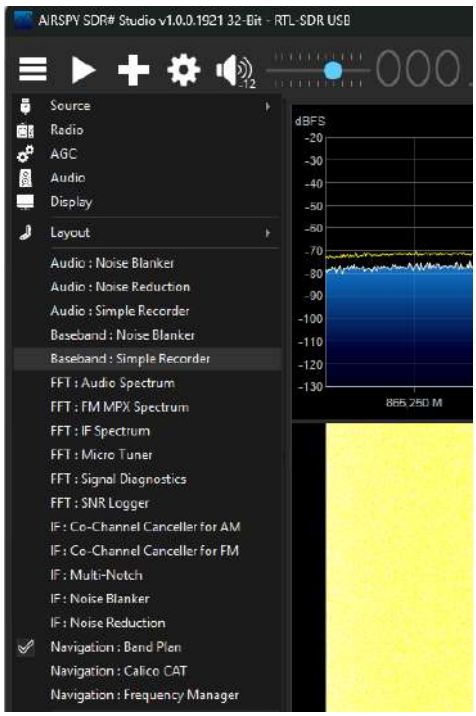


Рисунок 5.31 – Інструмент «Baseband: Simple Recorder» для запису сигналу

Сирі дані – це «необроблений» сигнал до демодуляції, декодування або будь-якої іншої обробки. Вони містять всю інформацію про сигнал, включаючи тип модуляції, частотні характеристики, фазові зміни тощо.

На рис. 5.32 подано зовнішній вигляд вікна інструменту «Baseband: Simple Recorder».

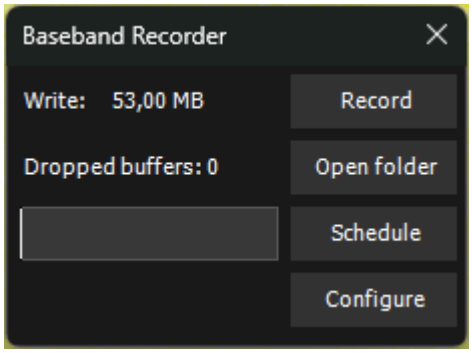


Рисунок 5.32 – Зовнішній вигляд вікна інструменту «Baseband: Simple Recorder»

Запис розпочинається після натискання на кнопку «Record», а файл зберігається в папку «C:\Users\UserName\Downloads\sdrsharp-x86\Audio\IQ\Date_Of_Record» у вигляді *.wav файлу.

Записані IQ-файли можна відкривати та аналізувати у різних професійних інструментах:

– GNU Radio – платформа для створення складних алгоритмів обробки сигналів. Можна побудувати власні декодери, фільтри, демодулятори для аналізу записаних даних;

– Inspectrum – спеціалізований інструмент для візуального аналізу IQ-файлів;

– Universal Radio Hacker (URH) – комплексний інструмент для реверс-інжинірингу бездротових протоколів;

– MATLAB/Octave – для математичного аналізу сигналів, спектрального аналізу, створення власних алгоритмів обробки;

Python (scipy, numpy, matplotlib) – для програмного аналізу, автоматизації обробки великих обсягів записів, машинного навчання.

В нашому експерименті ми будемо використовувати записаний сигнал в якості джерела вхідних даних. Це дасть нам змогу виконувати масштабування та дослідження модуляції отриманих сірих даних.

В програмі SDR# потрібно відкрити головне меню, обрати розділ «Source» в якому замість «RTL-SDR USB» обрати «Baseband File Player» (рис. 5.33).

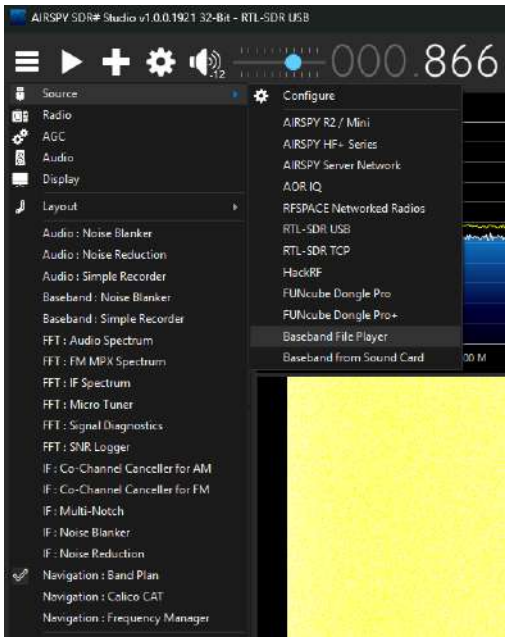


Рисунок 5.33 – Вибір інструменту «Baseband File Player» для дослідження записаного сигналу

Після обирання джерела вхідного сигналу зліва відкриється вікно зі спектром записаного сигналу (рис. 5.34).

Baseband File Player дозволяє відтворювати раніше записаний IQ-сигнал (baseband) так, ніби він надходить у програму безпосередньо з реального SDR-приймача. Фактично він підміняє апаратне джерело сигналу файлом, зберігаючи при цьому весь стандартний ланцюг обробки SDR#: спектр, водоспад, фільтри, демодулятори та плагіни працюють так само, як у режимі live-прийому. Це означає, що записаний ефір можна аналізувати офлайн, не маючи доступу до антени чи радіообладнання, і багаторазово повертатися до того самого фрагмента сигналу.

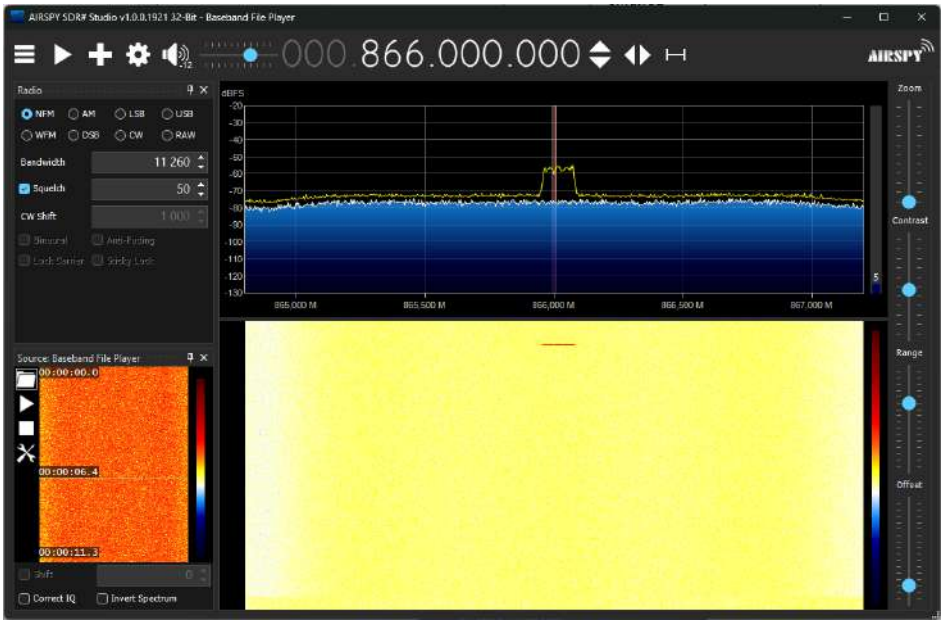


Рисунок 5.34 – Вікно «Baseband File Player» зі спектром записаного сигналу

Зазвичай Baseband File Player використовується для відкриття IQ-файлів у форматі WAV або RAW, які були збережені в режимі Baseband із заданою частотою дискретизації. Після завантаження файлу користувач може запускати і зупиняти відтворення, ставити його на паузу, перемотувати вперед і назад, а також зациклювати сигнал. Частота, зазначена під час запису, стає опорною для спектра і водоспаду, тому всі частотні співвідношення зберігаються коректно, ніби сигнал приймається наживо.

На рис. 5.35 подано місцезнаходження записаного LoRa сигналу.

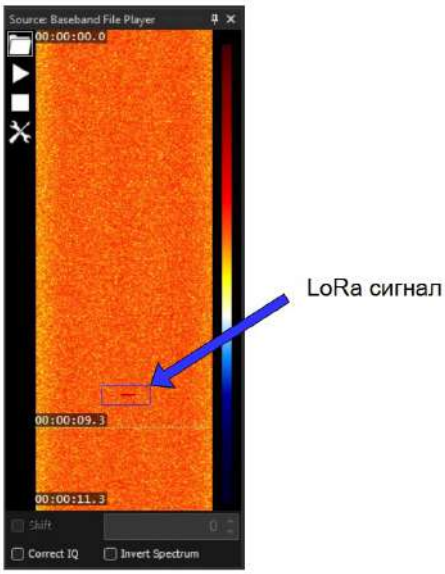


Рисунок 5.35 – Місцезнаходження записаного LoRa сигналу

Для детального дослідження потрібно обвести мишею область розташування LoRa сигналу. Це може потребувати декількох ітерацій, поки сигнал на буде чітко видимий у вікні програвача (рис. 5.36).

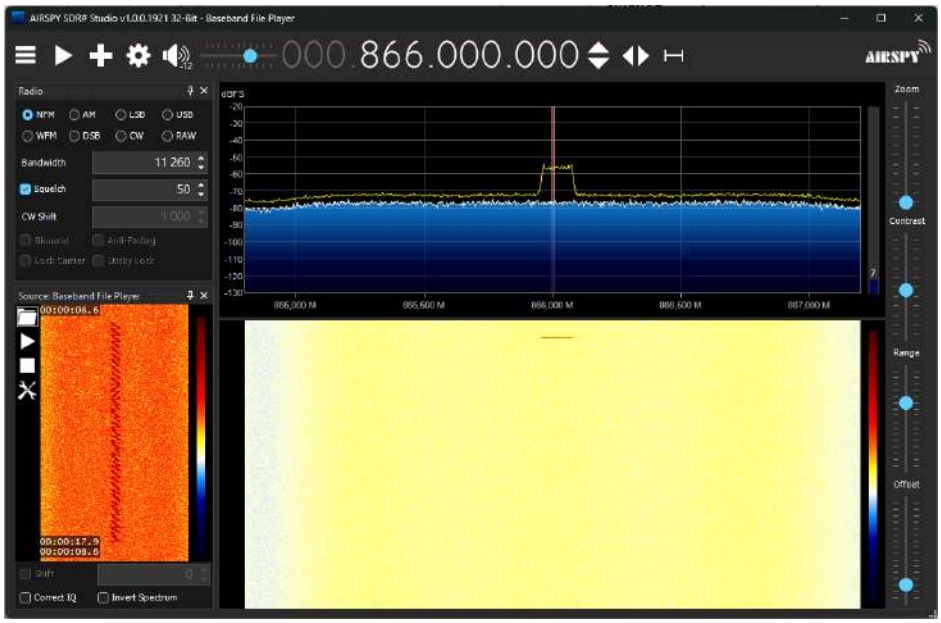


Рисунок 5.36 – Масштабування сигналу в програмі SDR#

Виконаємо дослідження сигналу за його візуальними ознаками (рис. 5.37).

Сигнал має характерну для LoRa часово-частотну структуру у вигляді послідовних діагональних ліній, що відповідають лінійним частотним чірпам.

Нахил ліній змінюється стрибкоподібно, формуючи «зигзагоподібний» рисунок, типовий для модуляції Chirp Spread Spectrum з дискретною зміною початкової частоти кожного символу.

Чірпи займають повну робочу смугу частот каналу, що вказує на використання стандартної LoRa-схеми з повним частотним розгортанням від нижньої до верхньої межі смуги. За виглядом спектрограми можна зробити висновок, що сигнал є стабільним по частоті: діагоналі рівні, без розмиття або викривлення, що свідчить про малий частотний дрейф та відсутність помітного Frequency Offset між передавачем і приймачем.

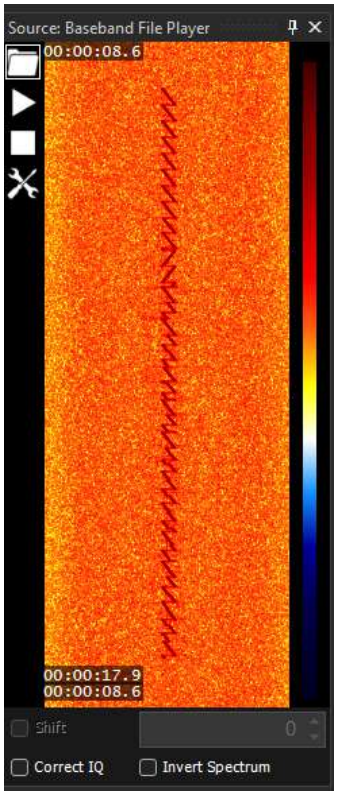


Рисунок 5.37 – Дослідження сигналу за його візуальними ознаками

Щільність чірпів у часі дозволяє припустити використання середнього або високого Spreading Factor. Візуально кількість символів на одиницю часу відповідає значенням SF ≈ 7…8, оскільки при SF = 11…12 чірпи були б значно довшими та менш численними.

Ширина частотного розгортання кожного чірпа, судячи з повного вертикального охоплення робочої смуги, відповідає типовому значенню BW ≈ 125 кГц, що є найпоширенішою конфігурацією для LoRa у діапазоні

868 МГц. Амплітуда сигналу відносно фонового рівня шуму є високою: чірпи чітко контрастують на фоні рівномірного шуму, що вказує на позитивне та значне відношення сигнал/шум, орієнтовно понад 20 дБ, достатнє для надійного декодування навіть при великих значеннях SF.

Структура сигналу не демонструє переривань або втрат символів, що свідчить про коректну передачу без колізій. Відсутність хаотичних частотних стрибків і чітка лінійність чірпів підтверджують, що це не FSK і не FHSS, а саме класичний LoRa-сигнал з CSS-модуляцією. Таким чином, за зображенням можна однозначно ідентифікувати LoRa-сигнал з орієнтовними параметрами: ширина смуги близько 125 кГц, Spreading Factor у діапазоні 7…8, стабільна центральна частота, високе SNR та тривалість пакета, характерна для швидкісної телеметричної передачі.

На водоспадній спектрограмі LoRa-сигналу (рис. 5.37) можливо частково інтерпретувати логічні частини кадру. LoRa не кодує Header, Payload і CRC як окремі візуально відмінні спектральні області, тому їх можна визначати лише непрямо, за зміною регулярності та тривалості чірпів, а не як чітко відокремлені ділянки, подібно до OFDM.

На верхній частині зображення спостерігається довга послідовність рівномірних, однаково нахилених діагональних чірпів, що займають усю робочу смугу частот і мають однакову структуру. Ця ділянка відповідає преамбулі LoRa, яка складається з N однакових up-chirp символів і використовується для синхронізації по часу та частоті. Візуально преамбула проявляється як найбільш регулярна і монотонна частина сигналу без зсувів початкової частоти між символами.

Безпосередньо після преамбули на зображенні помітна коротка зміна структури чірпів, де нахил або початкова частота виглядають зламаними або менш регулярними. Ця область відповідає Sync Word та SFD (Start Frame Delimiter), які формуються down-chirp та інверсними чірпами і служать маркером початку корисного кадру.

Далі, у середній та нижній частині зображення, спостерігається послідовність чірпів зі змінним вертикальним зсувом, але з однаковим нахилом, що є характерною ознакою даних LoRa, закодованих шляхом циклічного зсуву чірпа. Ця область включає Header і Payload, які на спектрограмі не мають чіткої межі між собою, оскільки Header є лише першими кількома символами Payload у режимі Explicit Header.

CRC, у свою чергу, не має жодної візуальної ознаки на спектрограмі, оскільки є лише додатковими символами наприкінці потоку даних і виглядає ідентично до корисного навантаження.

5.5.5 Популярні інструменти для декодування прийнятого LoRa-сигналу

Для декодування прийнятого сигналу можна використовувати різні інструменти. Розглянемо декілька з них:

– SDRangel;

– GNU Radio.

SDRangel є інтегрованим SDR-застосунком із графічним інтерфейсом, орієнтованим на практичну роботу з реальними сигналами в режимі реального часу. Архітектурно SDRangel побудований за принципом модульної обробки потоків даних, де IQ-потік, отриманий від SDR-пристрою (RTL-SDR, HackRF, LimeSDR, USRP та інших), може одночасно оброблятися кількома демодуляційними або аналітичними модулями.

Програма підтримує багатоканальний прийом, що дозволяє в межах однієї смуги спостерігати й аналізувати декілька сигналів паралельно. Для задач декодування LoRa особливо важливими є спектральний аналіз і водоспад, які в SDRangel реалізовані з високою роздільною здатністю в часі та частоті.

LoRa-сигнал, що використовує модуляцію CSS (Chirp Spread Spectrum), на водоспаді проявляється у вигляді характерних діагональних ліній, і SDRangel дозволяє чітко візуалізувати ці особливості, коректно виставляючи ширину смуги, частоту дискретизації та центр прийому.

Завдяки підтримці плагінів і мережевих інтерфейсів SDRangel може бути використаний як фронтенд для попереднього виявлення LoRa-сигналів, їх запису у вигляді IQ-даних та подальшої передачі на інші системи аналізу або декодування. Таким чином, SDRangel добре підходить для етапів моніторингу ефіру, ідентифікації сигналу та первинного аналізу LoRa.

GNU Radio, у свою чергу, є універсальним фреймворком для цифрової обробки сигналів, який надає користувачеві набір базових DSP-блоків і механізмів для побудови власних ланцюгів обробки. На відміну від SDRangel, GNU Radio не є просто програмою для прийому сигналів, а виступає як середовище розробки, у якому користувач сам визначає структуру приймального тракту.

Обробка даних у GNU Radio реалізується у вигляді графів потоків, де кожен блок виконує конкретну функцію: фільтрацію, перетворення Фур'є, детекцію, демодуляцію, синхронізацію або декодування. Для задач роботи з LoRa це має принципове значення, оскільки модуляція CSS вимагає специфічних алгоритмів: генерації та кореляції чірпів, оцінки Spreading Factor, синхронізації за преамбулою та точного визначення частотного зсуву.

У GNU Radio існують як готові модулі для роботи з LoRa (наприклад, gr-lora), так і можливість реалізації власних алгоритмів на Python або C++, що дозволяє глибоко досліджувати фізичний рівень протоколу. Крім того, GNU Radio зручно використовувати для офлайн-аналізу, коли IQ-дані LoRa записуються у файл, а потім багаторазово програються через один і той самий обробний ланцюг із різними параметрами.

5.5.6 Декодування прийнятого LoRa-сигналу за допомогою SDRangel

Після успішного захоплення радіочастотного сигналу наступним критичним етапом дослідження є його демодуляція та декодування для отримання корисного навантаження. У даному підрозділі розглянуто процес програмного аналізу LoRa-пакетів за допомогою універсальної SDR-платформи SDRangel. На відміну від стандартних приймачів, використання SDRangel дозволяє не лише візуалізувати спектр у реальному часі, а й застосовувати спеціалізовані плагіни для дешифрування протоколів з розширенням спектра (англ. Chirp Spread Spectrum, CSS), що є основою технології LoRa.

Після завантаження програми необхідно створити нове робоче середовище, обравши опцію «New» в головному меню в пункті «Workspace» (рис. 5.38).

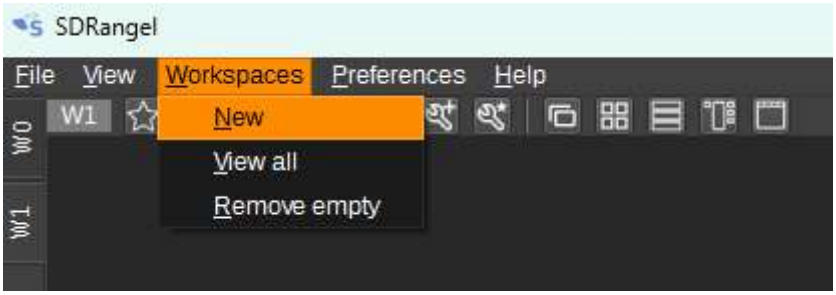


Рисунок 5.38 – Створення нового робочого середовища

В першу чергу необхідно додати джерело радіосигналу. В нашому випадку використаємо модуль RTL-SDR Blog. Для додавання в робоче середовище нового пристрою потрібно викликати діалогове вікно, натиснувши на кнопку «Add RX Device» (крок 1 на рис. 5.39). Далі відкриваємо випадний список з різними типами пристроїв (крок 2 на рис. 5.39). У списку доступних джерел вхідного сигналу обираємо «RTL-SDR xxxx» (крок 3 на рис. 5.39).

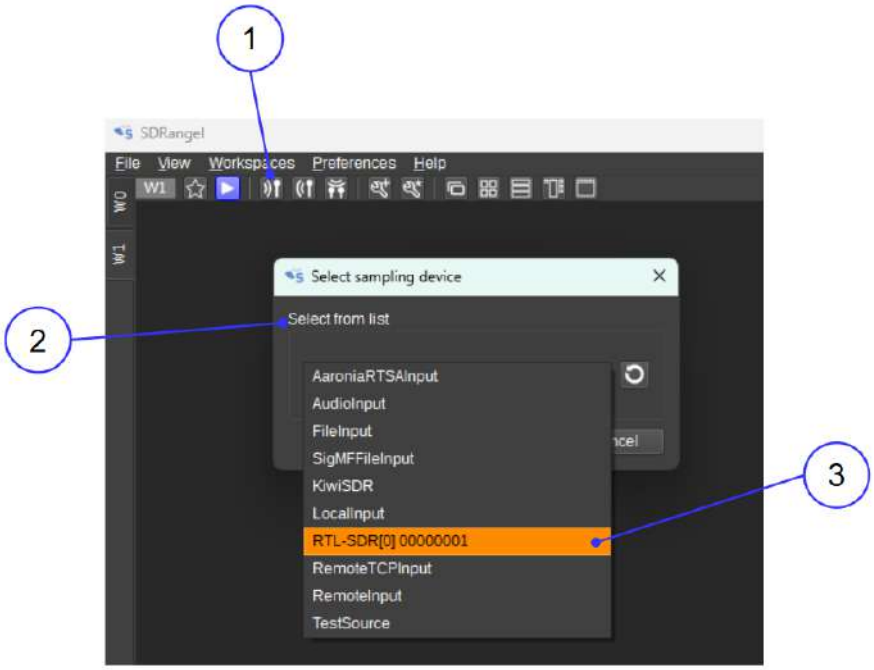


Рисунок 5.39 – Послідовність додавання нового джерела радіосигналу до проєкту

В робочому середовищі з’являться два вікна:

– вікно джерела сигналу;

– вікно основного спектру сигналу.

Інтерфейс вікна джерела сигналу подано на рис. 5.40.

В поданому інтерфейсі присутні такі інформаційні та керуючи компоненти:

1 – загальні параметри потоку;

2 – корекція локального генератора, що застосовується до локального генератора в ppm;

3 – параметри автоматичної корекції (DC – автоматичне видалення постійної складової; IQ – автоматичне встановлення балансу I/Q. Для ефективної корекції постійної складової необхідно ввімкнути IQ корекцію);

Рисунок 5.40 – Інтерфейс вікна джерела сигналу

3а – активація або деактивація Т-подібного елементу зміщення антени. Працює лише з адаптерами v3, інші його просто ігноруватимуть. Фактично чекбокс активує або деактивує контакт GPIO 0, який керує Т-подібним елементом зміщення на адаптерах v3;

4 – положення центральної частоти пропускання зі смугою відносно центральної частоти RTL-SDR:

– Cen – операція проріджування відбувається навколо центральної частоти RTL-SDR Fs;

– Inf – операція проріджування відбувається навколо Fs – Fc;

– Sup – операція проріджування відбувається навколо Fs + Fc.

5 – діалогове вікно відкриття режиму трансвертера. Ця кнопка відкриває діалогове вікно для налаштування параметрів перетворення частоти в режимі трансвертера;

6 – перемикання діапазону низької/високої частоти дискретизації. Коли кнопка увімкнена, частота дискретизації може змінюватися від 230 кВ/с до 300 кВ/с (у вибірках за секунду). Коли кнопка вимкнена, частота дискретизації може змінюватися від 950 кВ/с до 2400 кВ/с;

7 – перемикач частоти дискретизації пристрою / частоти дискретизації основної смуги частот. Ця кнопка використовується для перемикання частоти

дискретизації (8) між частотою дискретизації пристрою та частотою дискретизації основної смуги частот. Кнопка показує поточний режим:

– SR – режим введення частоти дискретизації пристрою. Частота дискретизації основної смуги частот – це частота дискретизації пристрою (8), поділена на коефіцієнт проріджування (9).

– BB – режим введення частоти дискретизації основної смуги частот. Частота дискретизації пристрою – це частота дискретизації основної смуги частот (8), помножена на коефіцієнт проріджування (9);

8 – частота дискретизації пристрою або частота дискретизації основної смуги частот у вибірках за секунду (В/с);

9 – коефіцієнт проріджування. Потік I/Q від АЦП RTLSDR знижується до степені двійки перед тим, як бути відправленим у смугу пропускання. Можливі значення – зростаючі степені двійки: 1 (без проріджування), 2, 4, 8, 16, 32, 64;

10 – режим прямої дискретизації. Цей прапорець використовується, щоб активувати спеціальну пряму дискретизацію RTLSDR. Це можна використовувати для налаштування на ВЧ-частоти;

11 – налаштування зсуву. Деякі радіочастотні інтерфейси, такі як застарілий E4000, реалізують цю функцію, і вона може суттєво зменшити центральний пік постійного струму без цифрової корекції. Це не працює для R820T та R820T2, які дуже популярні, на яких це не матиме жодного ефекту. Однак ці радіочастотні інтерфейси демонструють центральний пік постійного струму, набагато менший, ніж на E4000, і його можна легко виправити цифровим способом за допомогою елемента керування (3);

12 – керування шириною смуги пропускання фільтра тюнера. Може змінюватися від 350 кГц до 8 МГц;

13 – повзунок встановлює підсилення радіочастот у дБ. Значення визначені в пристрої RTL SDR і зазвичай такі: 0.0, 0.9, 1.4, 2.7, 3.7, 7.7, 8.7, 12.5, 14.4, 15.7, 16.6, 19.7, 20.7, 22.9, 25.4, 28.0, 29.7, 32.8, 33.8, 36.4, 37.2, 38.6, 40.2, 42.1, 43.4, 43.9, 44.5, 48.0, 49.6;

14 – автоматичне регулювання посилення (AGC). Прапорець AGC можна використовувати для ввімкнення або вимкнення AGC RTL2838. Це не залежить від налаштування посилення, оскільки це AGC діє після блоку посилення.

Докладно інтерфейс зміни загальних параметрів потоку подано на рис. 5.41.

Рисунок 5.41 – Інтерфейс зміни загальних параметрів потоку

В поданому інтерфейсі присутні такі інформаційні та керуючи компоненти:

1 – центральна частота прийому в кГц;

2 – кнопка запуску/зупинки пристрою;

3 – частота дискретизації пристрою або основної смуги. У режимі введення частоти дискретизації пристрою (7) це частота дискретизації I/Q основної смуги в кВ/с (у вибірках за секунду). Це частота дискретизації пристрою (8), поділена на коефіцієнт проріджування (9).

У режимі введення частоти дискретизації основної смуги (7) це частота дискретизації I/Q пристрою в кВ/с (у вибірках за секунду). Це частота дискретизації основної смуги (8), помножена на коефіцієнт проріджування (9).

Інтерфейс вікна основного спектра подано на рис. 5.42.

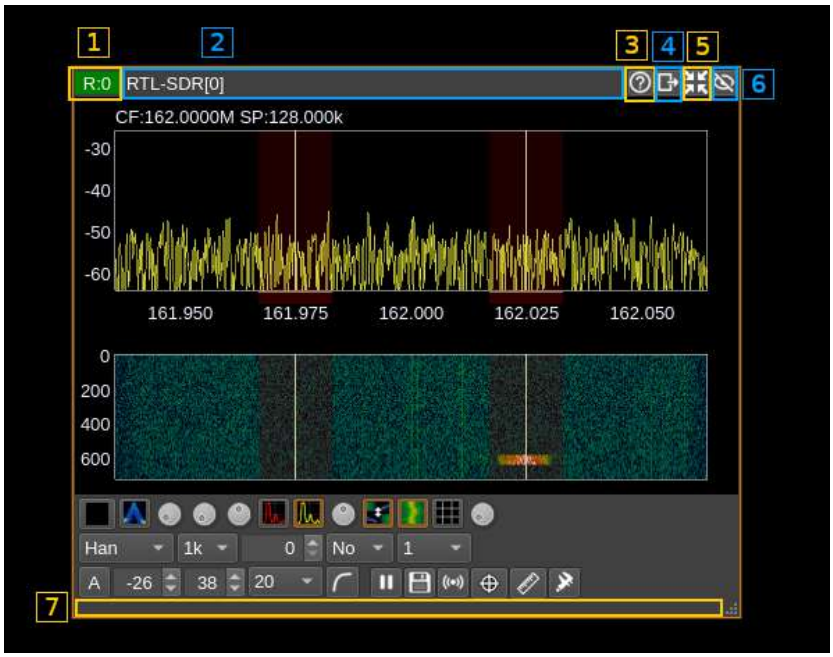


Рисунок 5.42 – Інтерфейс вікна основного спектра

В поданому інтерфейсі присутні такі інформаційні та керуючи компоненти:

1 – індекс пристрою, для якого відображається спектр. Він такий самий, як і у відповідному вікні пристрою. Формат такий самий, як і у вікні пристрою:

– R – приймач;

– T – передавач;

– M – MIMO;

2 – заголовок вікна показує тип пристрою та порядковий номер, для якого відображається спектр;

3 – довідка про елементи інтерфейсу;

4 – перехід до іншого робочого простору. Відкриває діалогове вікно для вибору робочого простору, до якого потрібно перемістити вікно пристрою. Нічого не відбувається, якщо вибрано той самий робочий простір;

5 – кнопка зменшення розміру вікна до мінімального;

6 – кнопка приховування даного вікна;

7 – область перетягування вікна.

Для декодування прийнятого LoRa-сигналу необхідно обрати відповідний канал (рис. 5.43).

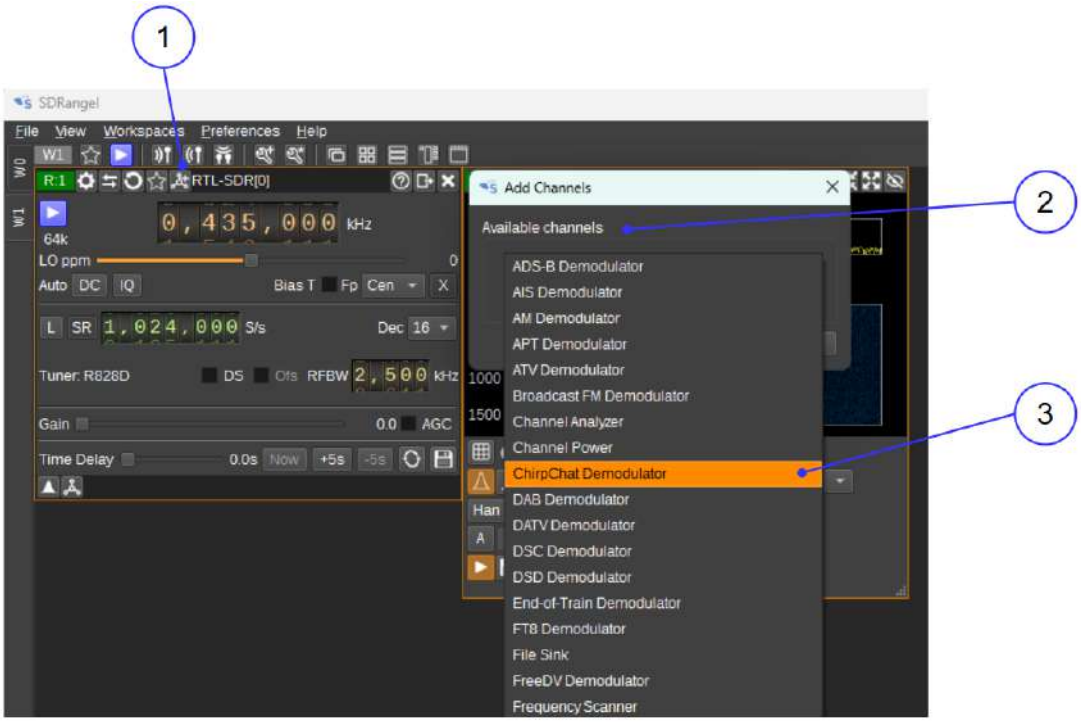


Рисунок 5.43 – Вибір каналу «ChirpChat Demodulator»

На першому кроці необхідно викликати діалогове вікно, натиснувши на кнопку «Channels» (крок 1 на рис. 5.43). Далі відкриваємо випадний список з різними типами каналів (крок 2 на рис. 5.43). У списку доступних каналів «ChirpChat Demodulator» (крок 3 на рис. 5.43).

На даному етапі інтерфейс програми SDRangel виглядатиме як подано на рис. 5.44.

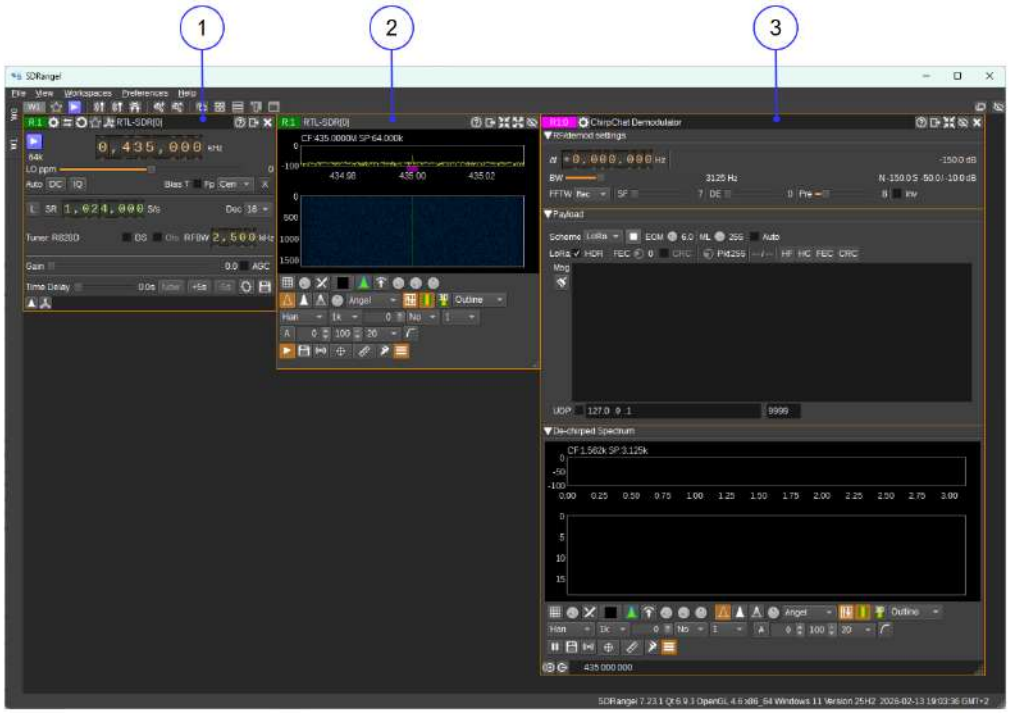


Рисунок 5.44 – Інтерфейс програми SDRangel з відкритими необхідними інструментами

В робочому просторі відкриті три вікна:

– вікно джерела сигналу;

– вікно основного спектра;

– вікно каналу демодуляції.

Виконаємо ряд налаштувань. У вікні каналу демодуляції потрібно обрати схему «LoRa» (рис. 5.45).

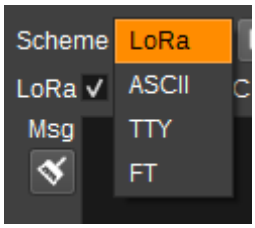


Рисунок 5.45 – Обирання схеми «LoRa» для демодуляції

У вікні джерела сигналу налаштовуємо частоту потоку 866 МГц (рис. 5.46, 1).

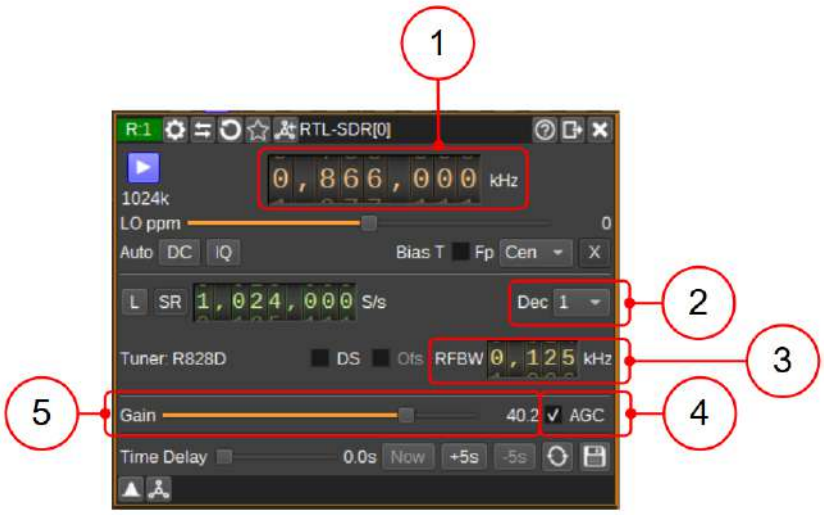


Рисунок 5.46 – Налаштування джерела сигналу

Коефіцієнт проріджування встановлюємо на позначку «1» (рис. 5.46, 2). Ширину смуги пропускання фільтра тюнера налаштовуємо в межах 125 КГц (рис. 5.46, 3). Вмикаємо автоматичне регулювання посилення (рис. 5.46, 4). Коефіцієнт підсилення радіочастот встановлюємо на 40,2 дБ (рис. 5.46, 5).

На рис. 5.47 подано приклад перехоплення та декодування LoRa сигналу.

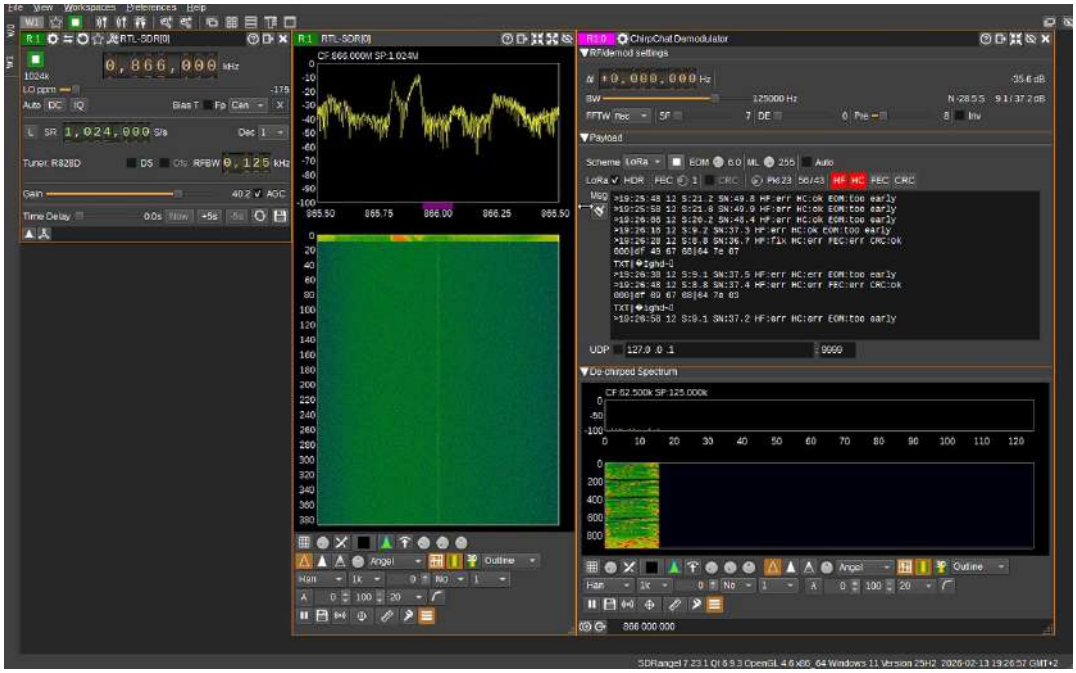


Рисунок 5.47 – Приклад перехоплення та декодування LoRa сигналу

За встановленою схемою декодування «LoRa» програма розпізнає структуру пакетів прийнятих повідомлень. На рис. 5.48 показано один з сеансів декодування.

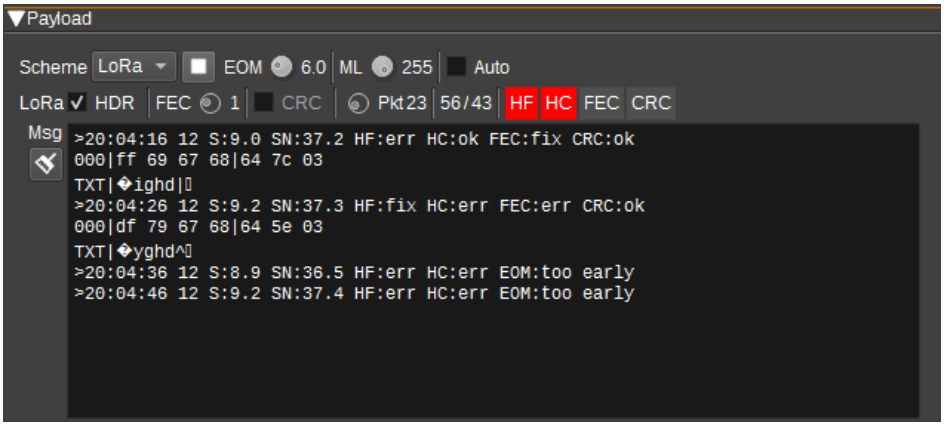


Рисунок 5.48 – Сеанс декодування

На даному рисунку подано фрагмент інтерфейсу SDRangel у режимі роботи канального плагіну LoRa Demodulator (ChirpChatDemodulator), а саме вкладку Payload, де відображаються результати демодуляції прийнятих LoRa-пакетів у реальному часі. У верхній частині панелі видно параметри декодування фізичного рівня: обрана схема LoRa, активований режим автоматичного визначення параметрів (Auto), встановлено максимальну довжину пакета (ML 255), а також увімкнена обробка заголовка (HDR), виправлення помилок (FEC) і перевірка контрольної суми (CRC). Поруч відображається службова інформація про пакет (Pkt 23 56/43), що вказує на порядковий номер пакета та статистику прийнятих/успішно декодованих кадрів.

Червоними індикаторами HF та HC позначено статус обробки заголовка кадру. HF (Header Flag або Header Frame status) сигналізує про наявність помилки в заголовку, тоді як HC (Header Check) відображає результат перевірки заголовкової частини після FEC-декодування. Нижче в текстовому полі Msg виводиться журнал прийнятих пакетів із часовими мітками. Наприклад, рядок виду «>20:04:16 12 S:9.0 SN:37.2 HF:err HC:ok FEC:fix CRC:ok» означає, що пакет був прийнятий о 20:04:16, параметр S відображає рівень сигналу або його відносну потужність, SN – співвідношення сигнал/шум (SNR). Далі подано статуси обробки: HF:err – помилка в заголовку на первинному етапі, HC:ok – заголовок після перевірки коректний, FEC:fix – помилки в даних були виявлені

та виправлені алгоритмом прямого виправлення помилок, CRC:ok – контрольна сума пакета правильна, тобто корисні дані відновлено успішно.

У наступному рядку подано шістнадцяткове представлення Payload, наприклад «000 | ff 69 67 68 64 7c 03», що є байтовою послідовністю корисного навантаження. Після цього виводиться інтерпретація в текстовому вигляді (TXT), де частина байтів відображається як ASCII-символи, якщо вони друковані. В інших записах видно випадки, коли FEC:err або HC:err, що означає неможливість виправлення помилок або некоректність заголовка, а також повідомлення «EOM:too early», яке вказує на передчасне завершення пакета – демодулятор виявив кінець кадру раніше очікуваної довжини, що може бути наслідком низького SNR, неправильного Spreading Factor або розсинхронізації.

Таким чином, даний інструмент дозволяє докладно роздивитись структуру LoRa пакетів та дає можливість досліднику тонко налаштовувати параметри вимірювальної системи, оптимізувати приймальний тракт за рівнем сигналу та співвідношенням сигнал/шум, коригувати ширину смуги, Spreading Factor і коефіцієнт кодування, а також оцінювати вплив завад і похибок синхронізації на достовірність декодування даних.

5.5.7 Декодування прийнятого LoRa-сигналу за допомогою GNU Radio

GNU Radio є універсальним середовищем цифрової обробки сигналів, яке дозволяє реалізувати повний ланцюг приймання, демодуляції та декодування LoRa-сигналів на фізичному рівні. На відміну від готових SDR-застосунків із фіксованою логікою роботи, GNU Radio надає користувачеві можливість самостійно сформувати структуру приймального тракту у вигляді графа потоків (Flowgraph), що складається з окремих функціональних блоків цифрової обробки сигналів. Такий підхід забезпечує гнучкість дослідження особливостей модуляції Chirp Spread Spectrum (CSS), яка використовується в LoRa, а також дозволяє змінювати параметри системи відповідно до конкретних умов експерименту.

Процес декодування LoRa-сигналу в GNU Radio починається з приймання комплексного IQ-потоку від SDR-приймача або з відтворення попередньо записаного файлу. Далі виконується попередня цифрова обробка, що включає фільтрацію, нормалізацію амплітуди та, за потреби, компенсацію частотного зсуву. Оскільки LoRa використовує лінійно змінювану частоту в межах заданої смуги пропускання, ключовим етапом є перетворення сигналу в часово-частотну

область та виконання операції де-чірпінгу – множення прийнятого сигналу на еталонний чірп із подальшим застосуванням швидкого перетворення Фур'є для визначення індексу символу.

Після синхронізації за преамбулою виконується визначення параметрів Spreading Factor, Bandwidth та Coding Rate. Отримані символи перетворюються в бітову послідовність, яка проходить етапи декодування з використанням алгоритмів прямого виправлення помилок і перевірки контрольної суми CRC. У результаті формується корисне навантаження пакета, яке може бути представлене у вигляді бінарних даних або текстової інформації залежно від структури переданого повідомлення.

В програмі GNU Radio версії 3.10.12.0 використовується вбудована бібліотека gr-lora_sdr, що була розроблена в Лабораторії телекомунікаційних схем EPFL [19].

Проєкт gr-lora_sdr – це повноцінна реалізація LoRa-фізичного рівня для середовища GNU Radio, доступна у вигляді out-of-tree модуля на GitHub (https://github.com/tapparelj/gr-lora_sdr) . Він містить усі необхідні блоки обробки сигналу для побудови LoRa-приймача (RX) і передавача (TX) у SDR-середовищі, включаючи синхронізацію, демодуляцію, обробку Hamming-кодування, деінтерлівінг, декодування і перевірку CRC.

Для декодування пакетів LoRa в даній схемі використаємо блок LoRa Rx (рис. 5.49).

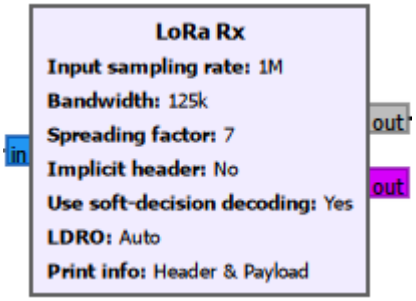


Рисунок 5.49 – Блок LoRa Rx для декодування прийнятих пакетів

Структура LoRa-RX реалізована не як один самодостатній компонент, а як набір взаємопов'язаних DSP-блоків, які разом формують приймальний ланцюг. Такий підхід відповідає концепції GNU Radio, де кожна операція цифрової обробки виділена в окремий блок, що у композиції формує повноцінний граф потоків.

Зокрема, реалізація включає: попереднє фільтрування і нормалізацію вхідного IQ-потоку з SDR-джерела, оцінку і корекцію Carrier Frequency Offset (CFO) та Sampling Time Offset (STO) для синхронізації на прийомі LoRa сигналів, пошук і вирівнювання за преамбулою чірпів, відокремлення корисних символів і їх подальшу демодуляцію шляхом FFT-аналізу сигналу протестованого для CSS модуляції LoRa. Ці операції реалізовані як окремі модулі або підблоки в пакеті gr-lora_sdr, які у компоновці забезпечують стійке декодування навіть за низькому відношенні сигнал/шум.

На рис. 5.50 подані налаштування блока LoRa Rx.

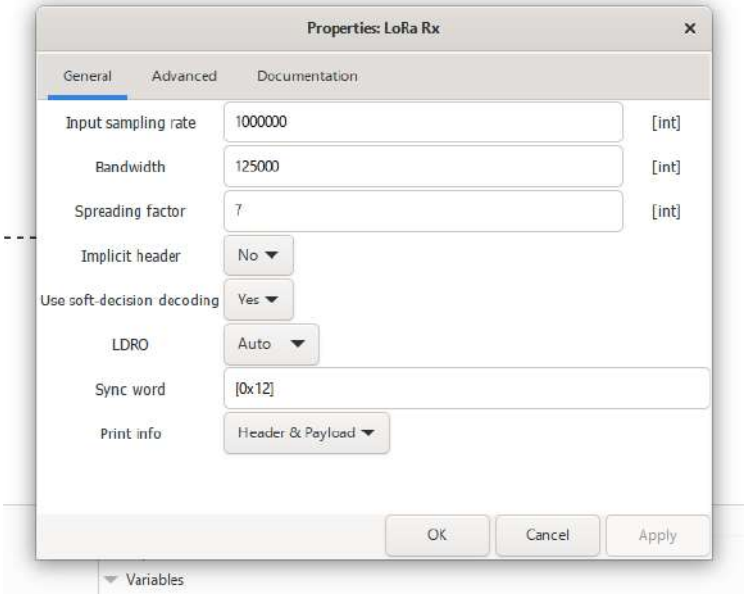


Рисунок 5.50 – Налаштування блока LoRa Rx

Як можна бачити з рисунку, параметр Input sampling rate встановлено 1000000 Гц, що визначає частоту дискретизації вхідного комплексного IQ-потоку. Це означає, що приймач очікує сигнал із частотою дискретизації 1 МГц, і всі внутрішні алгоритми синхронізації, фільтрації та FFT-демодуляції працюють, виходячи з цього значення. Частота дискретизації повинна бути узгоджена з налаштуванням SDR-джерела, оскільки невідповідність призведе до порушення масштабування символів і помилок синхронізації.

Параметр Bandwidth встановлено на рівні 125000 Гц, що відповідає стандартній смузі пропускання LoRa 125 кГц. Саме це значення визначає швидкість лінійної зміни частоти чірпа, тривалість символу та частотний інтервал між можливими символами після операції де-чірпінгу та FFT.

Відповідно до специфікації LoRa, при SF = 7 і BW = 125 кГц тривалість одного символу становить $Ts = 2^{SF} / BW$, що визначає часову структуру пакета та вимоги до синхронізації.

Параметр Spreading Factor встановлено рівним 7, що означає використання $2^7 = 128$ можливих частотних позицій символу в межах смуги. Це впливає на швидкість передачі даних, завадостійкість і тривалість символу. За значення SF = 7 забезпечується відносно висока швидкість передачі при помірній енергетичній ефективності. Даний параметр повинен відповідати налаштуванню передавача, інакше демодуляція буде неможливою через невідповідність кількості частотних відліків швидкого перетворення Фур'є (англ. Fast Fourier Transform, FFT).

Опція Implicit header встановлена в положення No, тобто використовується явний заголовок (Explicit Header Mode). У цьому режимі приймач очікує наявність службового заголовка в пакеті, який містить інформацію про довжину Payload, Coding Rate і наявність CRC. Це дозволяє автоматично визначати параметри конкретного пакета без їх жорсткого фіксування в приймачі.

Параметр Use soft-decision decoding активований (Yes). У цьому режимі демодулятор враховує амплітудну інформацію або метрики достовірності символів, а не лише жорсткі бінарні рішення, що підвищує ймовірність коректного відновлення даних за низькому співвідношенні сигнал/шум.

Параметр LDRO (Low Data Rate Optimization) встановлений у режим Auto. Таким чином, оптимізація для низьких швидкостей передачі активується автоматично залежно від поєднання Spreading Factor і Bandwidth. LDRO компенсує вплив дрейфу тактової частоти та нестабільності генератора при довгих символах, характерних для великих SF.

Поле Sync word має значення 0x12, що відповідає стандартному синхрослову LoRa WAN для публічних мереж. Синхрослово використовується для фільтрації пакетів і дозволяє приймачеві розпізнавати лише ті кадри, які належать до певного класу мережі або застосування.

Параметр Print info встановлено у режим Header & Payload. Це включає режим виведення в консоль GNU Radio інформації як про заголовок пакета, так і про корисне навантаження. Таким чином, це дозволяє контролювати процес синхронізації, декодування та перевірки CRC у режимі реального часу.

Блок, який виводить прийняті пакети в консоль називається Message Debug (рис. 5.51).

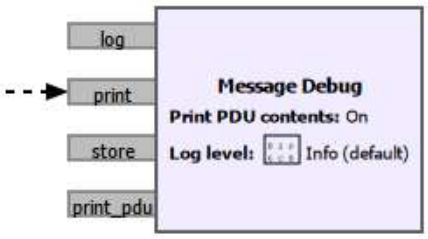


Рисунок 5.51 – Блок, що виводить прийняту інформацію в консоль

Даний блок використовує налаштування за замовченням (рис. 5.52).

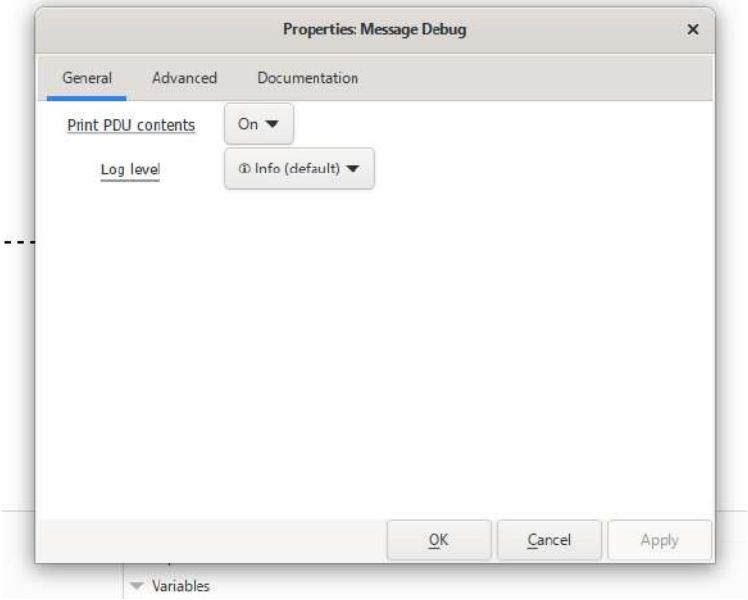


Рисунок 5.52 – Параметри блока Message Debug

Отримання пакетів в реальному часі відбувається за допомогою блока RTL-SDR Source, що також є вбудованим в програму GNU Radio (рис. 5.53).

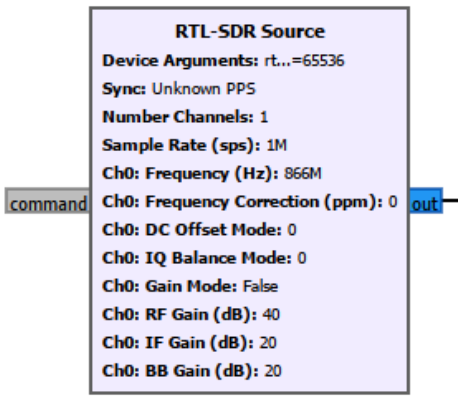


Рисунок 5.53 – Блок RTL-SDR Source

Параметри налаштування даного блока подані на рис. 5.54.

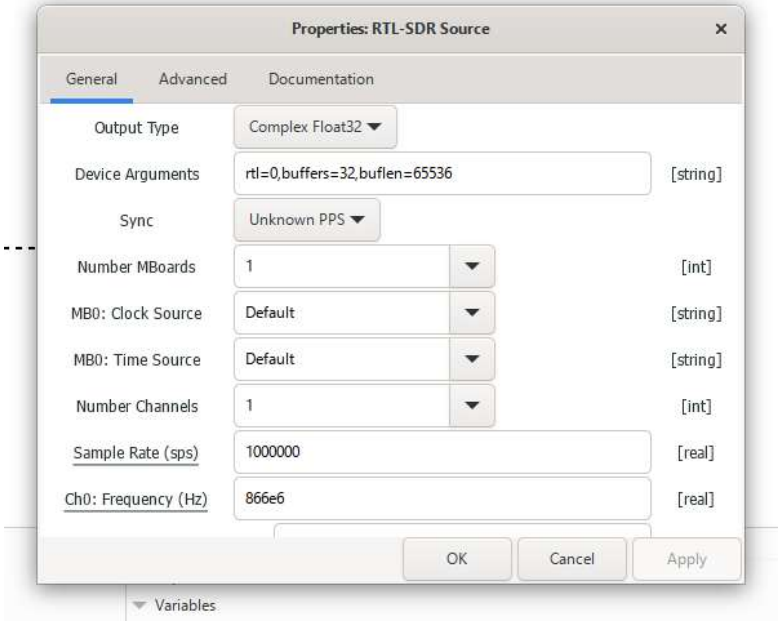


Рисунок 5.54 – Параметри налаштування блока RTL-SDR Source

Даний блок формує початковий потік даних у приймальному тракті та забезпечує оцифрування радіочастотного сигналу з подальшою передачею його у вигляді комплексних відліків у DSP-ланцюг GNU Radio.

Тип вихідних даних встановлено як Complex Float32 – представлення кожного відліку у вигляді комплексного числа з плаваючою комою одинарної точності (32 біти на дійсну та 32 біти на уявну складову). Такий формат є стандартним для подальшої цифрової обробки сигналів у GNU Radio, оскільки забезпечує достатню динамічну точність під час виконання фільтрації, FFT та демодуляції.

У полі Device Arguments задано параметри ініціалізації драйвера пристрою: «rtl=0,buffers=32,buflen=65536». Параметр rtl=0 вказує на використання першого доступного RTL-SDR приймача. Параметр buffers=32 визначає кількість внутрішніх буферів для передачі даних, а параметр buflen=65536 встановлює довжину кожного буфера. Ці параметри впливають на латентність та стабільність потокової передачі даних і можуть бути оптимізовані залежно від продуктивності системи та вимог реального часу.

Параметр Sync встановлено у значення Unknown PPS, що означає відсутність використання зовнішнього сигналу точного часу (Pulse Per Second, PPS) для синхронізації.

Кількість материнських плат (Number MBoards) дорівнює 1, що відповідає використанню одного фізичного SDR-пристрою. Джерело тактової частоти (Clock Source) та джерело часу (Time Source) встановлені у режим Default, тобто використовується внутрішній генератор пристрою без зовнішньої опорної частоти.

Параметр Number Channels встановлено у 1, що означає роботу в одноканальному режимі прийому. Це відповідає можливостям RTL-SDR, який підтримує один незалежний приймальний канал.

Частота дискретизації Sample Rate (sps) встановлена на рівні 1000000 відліків за секунду (1 МГц). Це визначає ширину спектра, що аналізується, і повинно бути узгоджене з параметрами наступних блоків обробки, зокрема демодулятора LoRa. При такій частоті дискретизації забезпечується достатній запас для обробки сигналу зі смугою 125 кГц із можливістю компенсації частотного зсуву та фільтрації.

Центральна частота прийому Ch0: Frequency (Hz) встановлена на рівні 866e6, тобто 866 МГц. Це відповідає одному з частотних каналів ISM-діапазону 868 МГц, який використовується технологією LoRa в Європі. Таким чином, блок налаштований на прийом сигналів у відповідному піддіапазоні, а весь спектр шириною приблизно 1 МГц навколо цієї частоти буде оцифрований і переданий у цифровий тракт.

Загальна схема декодування LoRa сигналу подана на рис. 5.55.

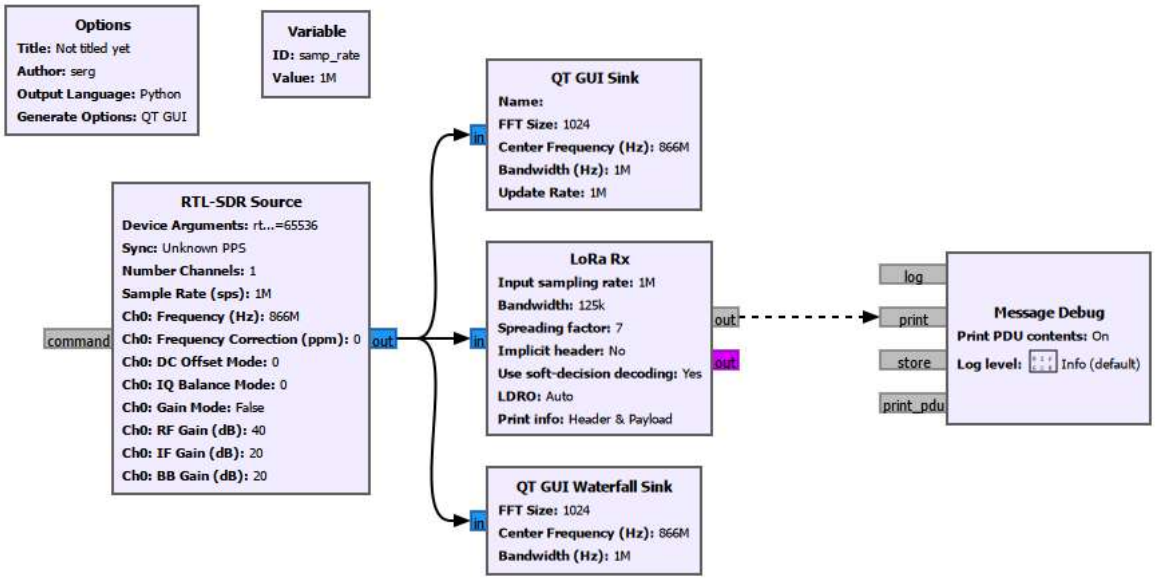


Рисунок 5.55 – Загальна схема декодування LoRa

Як можна бачити, крім описаних блоків RTL-SDR Source, LoRa Rx та Message Debug в схемі використані додаткові блоки QT GUI Sink та QT GUI Waterfall Sink.

Подана на рис. 5.55 схема (Flowgraph) у середовищі GNU Radio Companion реалізує повний тракт прийому, демодуляції та декодування радіосигналів LoRa у діапазоні 868 МГц з використанням програмно-визначеного радіообладнання RTL-SDR.

Flowgraph складається з чотирьох основних функціональних блоків: джерела радіочастотного сигналу (RTL-SDR Source), блоку декодування LoRa (LoRa Rx), блоків візуалізації спектру (QT GUI Sink та QT GUI Waterfall Sink) та блоку виведення декодованих даних (Message Debug). Система налаштована на прийом LoRa-сигналів з параметрами Spreading Factor 7, Bandwidth 125 кГц на центральній частоті 866 МГц, що відповідає європейському діапазону ISM для LoRa WAN-мереж.

Схема включає два паралельних блоки візуалізації, що отримують той самий потік IQ-відліків з RTL-SDR Source та надають різні форми представлення радіочастотного спектру для дослідника. Блок QT GUI Sink реалізує класичний спектральний аналізатор з графіком потужності сигналу у частотній області (Spectrum Analyzer View) (рис. 5.56).

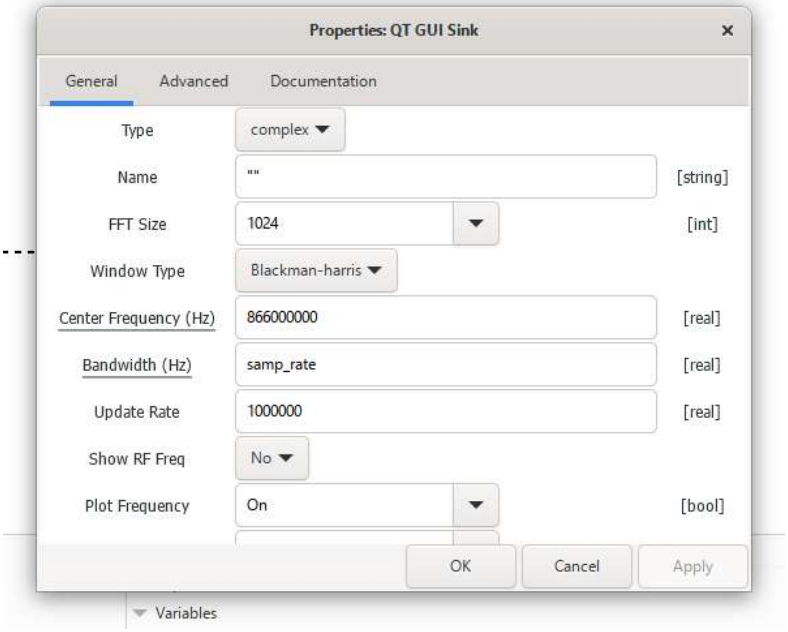


Рисунок 5.56 – Налаштування блока QT GUI Sink

Спектральний графік QT GUI Sink показує миттєвий розподіл енергії сигналу по частотах у вигляді графіка з горизонтальною віссю частоти та вертикальною віссю потужності (зазвичай у dBm або dB відносно повної шкали).

Параметр FFT Size встановлений на 1024, що означає використання 1024-точкового швидкого перетворення Фур'є для обчислення спектру. Цей розмір FFT визначає частотну роздільну здатність спектрограми: при Sample Rate 1 МГц роздільна здатність складає 1000000 Hz / 1024 ≈ 977 Hz на bin (смугу частот). Більший розмір FFT (наприклад, 4096 або 8192) дав би кращу частотну роздільну здатність (вужчі bins), що дозволило б точніше виміряти ширину смуги сигналу, але потребувало б більше обчислювальних ресурсів та зменшило б швидкість оновлення графіка.

Center Frequency встановлена на 866M (866 МГц), що відповідає центральній частоті прийому RTL-SDR та забезпечує правильну мітку на горизонтальній осі спектрограми.

Bandwidth встановлена на 1M (1 МГц), що відповідає Sample Rate та визначає діапазон частот, що відображається на графіку (від 865,5 МГц до 866,5 МГц при центральній частоті 866 МГц).

Update Rate встановлений на 1 МГц.

Для детального аналізу часової структури LoRa сигналу важливим є другий блок візуалізації – QT GUI Waterfall Sink, який реалізує спектрограму (Waterfall Display) (рис. 5.57).

Цей блок також має FFT Size 1024, Center Frequency 866M та Bandwidth 1M для узгодженості з основним спектральним аналізатором. Ключова відмінність полягає у формі візуалізації: замість графіка потужності в реальному часі, Waterfall відображує історію спектра у вигляді двовимірного зображення, де горизонтальна вісь представляє частоту, вертикальна вісь – час (минуле зверху, теперішнє знизу), а колір кожного пікселя відповідає рівню потужності сигналу на даній частоті у даний момент часу.

Саме на спектрограмі Waterfall найкраще видно характерні візуальні ознаки LoRa-сигналів: діагональні лінії up-chirps преамбули, down-chirps Start Frame Delimiter та «сходоподібну» структуру payload. За допомогою даної діаграми дослідник може швидко ідентифікувати присутність LoRa-передавання, навіть не дивлячись на декодовані дані, та оцінити параметри сигналу (SF за товщиною діагональних ліній, BW за вертикальним розміром, тривалість пакету за часовою протяжністю).

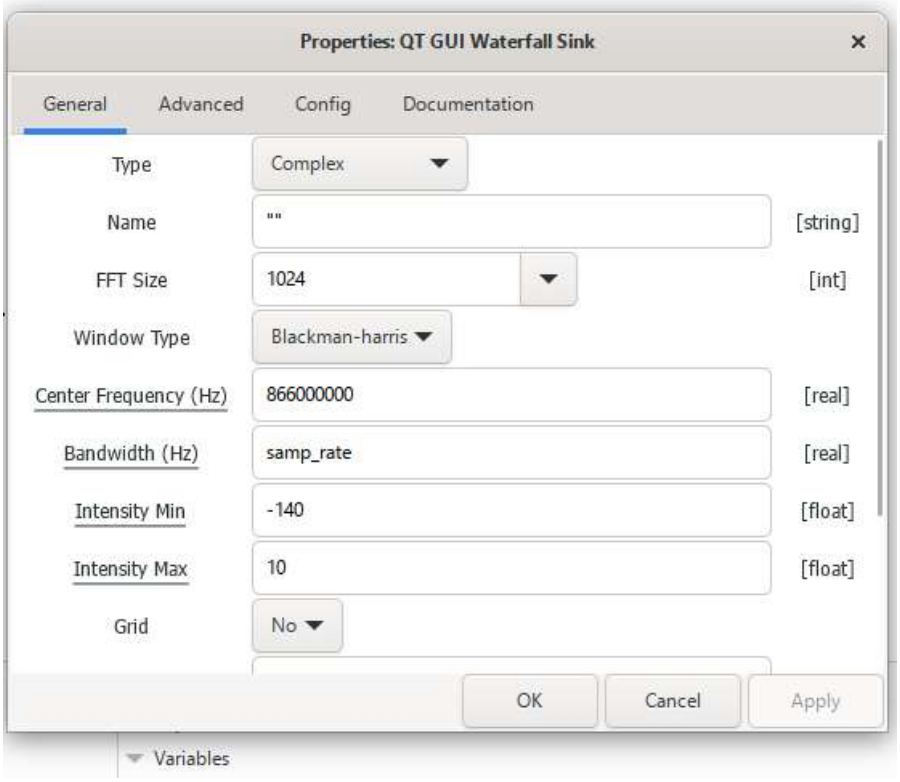


Рисунок 5.57 – Блок візуалізації QT GUI Waterfall Sink

Обидва блоки візуалізації працюють паралельно з блоком декодування, що дозволяє одночасно спостерігати візуальну картину радіочастотного середовища та отримувати декодовані дані.

На рис. 5.58 подано приклад роботи програми в режимі візуального дослідження параметрів LoRa сигналу.

Інтерфейс складається з двох основних графічних панелей: верхня панель показує спектрограму (Waterfall Display) у частотно-часовій області, а нижня панель відображає спектральний графік (Frequency Display) з розподілом потужності сигналу по частотах у реальному часі.

У верхній частині панелі видно тонкі лінії, що відповідають прийнятим LoRa-пакетам. Їх вузька ширина свідчить про високу швидкість передавання даних та параметр SF, що дорівнює, щонайменше 7.

В нижній частині рисунку видно спектрограму прийнятого сигналу. Спектральний графік показує широкий пік на центральній частоті 866,0 МГц з характеристиками, що відповідають LoRa-сигналу: ширина близько 300 кГц, плоска верхівка з нерівностями та дуже високий SNR близько 40…50 дБ. Цей пік представляє сильний LoRa-сигнал від близького передавача, захоплений у режимі Max Hold.

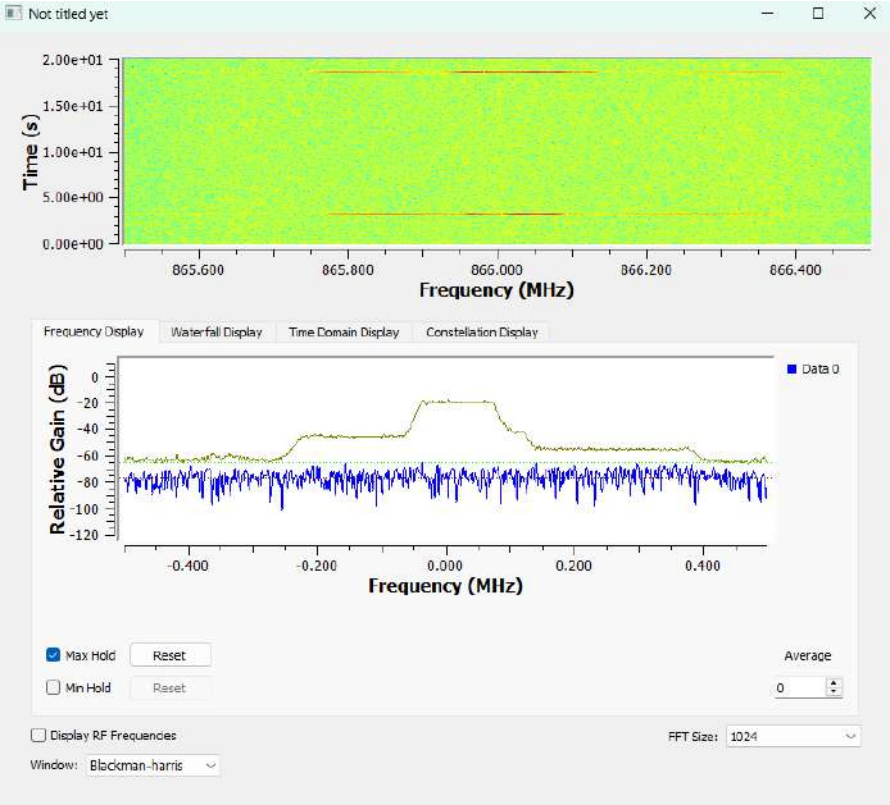


Рисунок 5.58 – Приклад роботи програми в режимі візуального дослідження параметрів LoRa сигналу

Декодоване повідомлення виводиться у консоль програми GNU Radio (рис. 5.59).

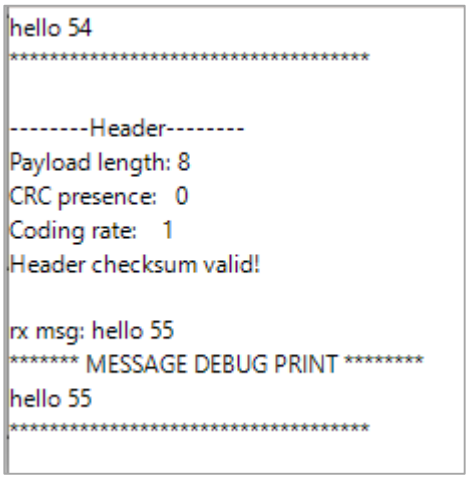


Рисунок 5.59 – Декодоване повідомлення у консолі програми GNU Radio

Таким чино, зібрана схема приймача LoRa сигналів з використанням блоку gr-lora_sdr успішно виконала задачу аналізу та декодування прийнятих пакетів LoRa.

Для успішного декодування невідомих сигналів необхідна верифікація та налаштування параметрів декодера (SF, BW, Sync Word) відповідно до очікуваних характеристик сигналу, що може бути здійснено шляхом систематичного сканування параметрів або детального аналізу записаних IQ-відліків у спеціалізованих інструментах.

5.6 Контрольні запитання та завдання

1. Які основні особливості бездротових сегментів АСУТП впливають на їхню кібербезпеку?
2. Які основні загрози характерні для радіочастотного середовища в АСУТП?
3. За якими ознаками класифікуються радіочастотні атаки?
4. У чому полягає принцип роботи спеціалізованих систем виявлення вторгнень типу WIDS?
5. Які функції виконують промислові системи радіомоніторингу?
6. У чому полягають переваги використання програмно-визначених радіосистем (SDR) для моніторингу?
7. Які можливості платформи WiFi Pineapple використовуються для аудиту безпеки бездротових мереж?
8. Яким чином здійснюється атака відмови в обслуговуванні за допомогою пристрою WiFi Deauther?
9. Які методи дозволяють виявляти атаки з використанням WiFi Pineapple та WiFi Deauther?
10. У чому полягає метод виявлення незареєстрованого передавача LoRa WAN та які інструменти застосовуються для декодування сигналів?

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАНЬ

1. P. Ackerman. Industrial Cybersecurity: Efficiently secure critical infrastruct. – Packt Publishing, 2017. – P. 456. ISBN 9781788395151.

2. Що таке Endpoint Detection and Response? Топ-3 найкращих EDR рішень. URL: https://softlist.com.ua/ua/news/chto-takoe-endpoint-detection (date of access: 05.05.2026).

3. Osterman Research. URL: https://ostermanresearch.com/ (date of access: 26.08.2024).

4. EPP. Системи захисту кінцевих точок Infosecurity. Infosecurity. Інформаційна безпека. URL: https://in4security.com/endpoint-protection-platform (date of access: 26.08.2024).

5. Учасники проектів Вікімедіа. Пісочниця (комп'ютерна безпека) – Вікіпедія. URL: https://uk.wikipedia.org/wiki/Пісочниця_(комп'ютерна_безпека) (date of access: 05.05.2026).

6. Ізольоване віртуальне середовище для надійного захисту програм. URL: https://iitd.com.ua/sandbox/ (date of access: 05.05.2026).

7. Node-RED та технологія промислового Інтернету речей : Навчальний посібник / І. Ш. Невлюдов, С. П. Новоселов, О. В. Сичова. – Харків: Видавництво Іванченка І. С., 2024. – 207 с. ISBN 978-617-8332-58-7. DOI: 10.30837/978-617-8332-58-7.

8. Промислові мережі та компоненти АСУТП : Навчальний посібник / І. Ш. Невлюдов, С. П. Новоселов, О. В. Сичова, С. І. Теслюк. – Харків: Видавництво Іванченка І. С., 2025. – 242 с. ISBN 978-617-8332-83-9. DOI: 10.30837/978-617-8332-83-9.

9. Wireshark – докладний посібник з початку використання. URL: https://hackyourmom.com/kibervijna/wireshark-dokladnyj-posibnyk-z-pochatku-vykorystannya/ (date of access: 05.05.2026).

10. Застосування цифрових двійників технічних засобів автоматизації для розроблення програмно-технічних комплексів АСУ ТП : Навчальний посібник / І. Ш. Невлюдов, С. П. Новоселов, О. В. Сичова. – Харків: Видавництво Іванченка І. С., 2023. – 267 с. ISBN 978-617-8059-95-8, DOI: 10.30837/978-617-8059-95-8.

11. Keysight Technologies. Enabling Technologies and Solutions for Design and Test. URL: https://www.keysight.com/us/en/assets/7018-05008/application-notes/5992-1175.pdf (date of access: 05.05.2026).

12. B.Snehal, P. Jadhav. Wireless Intrusion Detection System. *International Journal of Computer Applications*, 2010, Vol. 5. https://doi.org/10.5120/934-1312.

13. B. Devanathan, E. Ammu. A Comparison study of various Wireless Intrusion Detection Systems. *International Journal on Science and Technology*, 2025, Vol. 16(4). https://doi.org/10.71097/ijsat.v16.i4.9500.

14. J. Rugeles, E. Guillén, L. Cardoso. A Technical Review of Wireless security for the Internet of things: Software Defined Radio perspective. 2020, https://doi.org/10.48550/arXiv.2009.10171.

15. A. Makarenko. The use of SDR as sensor nodes of the intelligent system of management of radio emission data collection. *Scientific Notes of the State University of Telecommunications*, 2023, Vol. 3. https://doi.org/10.31673/2786-8362.2023.010202.

16. G. Michalis, A. Rousias, L. Kanaris, A. Kokkinis, P. Kanaris, S. Stavrou. A Distributed RF Threat Sensing Architecture. *Information*, 2024, Vol. 15(12):752. https://doi.org/10.3390/info15120752.

17. I. Nevludov, S. Novoselov, O. Sychova. Modeling and Practical Implementation of the Optimal Wireless Security Gateway for the Industrial Automation Network. *Serbian Journal of Electrical Engineering*, Vol. 19 (3), 2022, pp. 303-327. UDC: 004.738.5:004.71, DOI: https://doi.org/10.2298/SJEE2203303N.

18. SIGIDWIKI.COM – Signal Identification Guide. Digital Signals. URL: https://www.sigidwiki.com/wiki/Category:Digital (date of access: 05.05.2026).

19. J. Tapparel, A. Burg. Design and Implementation of LoRa Physical Layer in GNU Radio. *Proceedings of the 14 th GNU Radio Conference*, 2024.

20. Fortinet. 2026 Fortinet Global Threat Landscape Report. URL: https://www.fortinet.com/uk/resources/reports/threat-landscape-report (date of access: 05.05.2026).

Рекомендуємо інші наші книги:

Застосування цифрових двійників технічних засобів автоматизації для розроблення програмно-технічних комплексів АСУ ТП : Навчальний посібник / І.Ш. Невлюдов, С.П. Новоселов, О.В. Сичова. – Харків: Видавництво Іванченка І.С., 2023. – 267 с.

ISBN 978-617-8059-95-8
DOI: 10.30837/978-617-8059-95-8

У навчальному посібнику розглядаються питання практичного застосовування віртуальних макетів технічних засобів автоматизації, які використовуються для розроблення програмно-технічних комплексів сучасних АСУ ТП. Наведено приклади найпоширеніших технічних засобів автоматизації технологічних процесів і реалізації керування ними за допомогою релейно-контактної логіки.

Технології Інтернету речей в управлінні пристроями на мікроконтролерах: Навчальний посібник [Електронний ресурс] / І.Ш. Невлюдов, В.А. Андрусевич, С.П. Новоселов, О.Г. Резніченко. – Електронне видання. – Харків: ХНУРЕ, 2023. – 214 с. – pdf 2,88 Mb.

ISBN 978-966-659-364-4
DOI: 10.30837/978-966-659-364-4

У навчальному посібнику подані відомості про сучасну концепцію Інтернету речей, що дозволяє здійснювати передачу та обмін даними між фізичним світом і комп’ютерними системами в автоматичному режимі, за допомогою використання стандартних протоколів зв’язку. На реальних прикладах розглянуто процес підключення, моніторингу датчиків, віддаленого управління інтелектуальними виконавчими пристроями через Інтернет.

Основи розробки кросплатформного програмного забезпечення на Avalonia : Навчальний посібник / С.П. Новоселов, О.В. Сичова. – Харків: Видавництво Іванченка І.С., 2024. – 267 с.

ISBN 978-617-8332-57-0
DOI: 10.30837/978-617-8332-57-0

У навчальному посібнику наведено відомості для розробки додатків на Framework Avalonia. Розглядаються різні аспекти – від налаштування робочого середовища, встановлення .NET на платформі Linux до реалізації доступу до бази даних і відображення структурованої інформації. Посібник також містить інформацію щодо створення інтерфейсу користувача засобами Avalonia, взаємодії між візуальними елементами, використання елементів керування, роботи з XAML і командами Avalonia. Подано матеріал щодо прив'язки даних, роботи зі стильовими властивостями, маршрутизації подій, шаблонів даних і моделей відображення.

Промислові мережі та компоненти АСУТП : Навчальний посібник / І.Ш. Невлюдов, С.П. Новоселов, О.В. Сичова, С.І. Теслюк. – Харків: Видавництво Іванченка І.С., 2025 . – 242 с.

ISBN 978-617-8332-83-9
DOI: 10.30837/978-617-8332-83-9

У навчальному посібнику подано теоретичні основи цифровізації виробництва, принципи роботи віртуальних приладів і цифрових двійників, а також засоби промислової комунікації на базі протоколу Modbus. Окрему увагу приділено практичному застосуванню середовища програмування CODESYS для створення, налаштування та управління автоматизованими системами керування технологічними процесами.

Node-RED та технологія промислового Інтернету речей : Навчальний посібник / І.Ш. Невлюдов, С.П. Новоселов, О.В. Сичова. – Харків: Видавництво Іванченка І. С., 2024. – 207 с.

ISBN 978-617-8332-58-7
DOI: 10.30837/978-617-8332-58-7

Навчальний посібник охоплює такі теми, як впровадження Інтернету речей і промислового Інтернету речей у виробництво, розглядається використання віртуальних приладів та цифрових двійників технологічного обладнання АСУТП в рамках концепції Індустрії 4.0. Подана інформація щодо використання мережі зв'язку 6G для поєднання датчиків та виконавчих пристроїв в інтелектуальній кібер-фізичній системі, а також використання специфічних протоколів комунікації, таких як Modbus та MQTT, для обміну даними та управління промисловим обладнанням.

Технологія програмування промислових контролерів в інтегрованому середовищі CODESYS: Навчальний посібник / І.Ш. Невлюдов, С.П. Новоселов, О.В. Сичова. – Харків: ХНУРЕ, 2019. – 264 с.

ISBN 978-966-659-265-4
DOI: 10.30837/978-966-659-265-4

У навчальному посібнику подані відомості про сучасні технології організації виробництва, зокрема розглянуті перспективи переходу до Індустрії 4.0 у вітчизняних виробництвах. Розглянуті особливості побудови сучасних ПЛК. Дано практичні рекомендації щодо створення складних програмно-апаратних комплексів керування технологічними процесами.

Пневматичні пристрої та засоби автоматизації мехатронних систем: Навчальний посібник / І.Ш. Невлюдов, Л.О. Кривопляс-Володіна, С.П. Новоселов, О.В. Сичова. – Харків: ФОП Панов А.М., 2020. – 255 с.

ISBN 978-617-7859-58-0
DOI: 10.30837/978-617-7859-58-0

У навчальному посібнику подані відомості про сучасні технології організації виробництва, зокрема розглянуті особливості використання пневматичних систем, що разом з засобами автоматизації та ПЛК дозволяють створювати мехатронні роботизовані системи на виробництві. Розглянуті технології інтелектуальних приводів, датчиків і пневматичних пристроїв, а також системи управління за стандартами Індустрії 4.0. Дано практичні рекомендації щодо створення складних програмно-апаратних комплексів керування технологічними процесами.

Електропневмоавтоматичні приводи в автоматизованих системах керування: Навчальний посібник / І.Ш. Невлюдов, Л.О. Кривопляс-Володіна, С.П. Новоселов, О.В. Сичова. – Харків: ХНУРЕ, 2021. – 292 с.

ISBN 978-966-659-332-3
DOI: 10.30837/978-966-659-332-3

Навчальний посібник дає можливість студентам ознайомитися із основами пневматики, навчитися виконувати розрахунки схем пневмоприводів, вивчити характеристики пневматичних приводів та оволодіти сучасними інструментами підготовки технологічних програм керування ТП, та перспективними мовами програмування промислових контролерів, такими як ST, LD, FBD, SFC.

Навчальне видання

НЕВЛЮДОВ Ігор Шакирович
ФИЛИПЕНКО Олександр Іванович
НОВОСЕЛОВ Сергій Павлович
СИЧОВА Оксана Володимирівна

КІБЕРБЕЗПЕКА АВТОМАТИЗОВАНИХ СИСТЕМ УПРАВЛІННЯ ТЕХНОЛОГІЧНИМ ПРОЦЕСАМИ

Частина I

Навчальний посібник

Формат 60х84 1/16.
Ум. друк. арк. 14,4. Наклад 200 прим. Зам 30-06.

Видавництво
ФОП Іванченко І.С.
пр. Тракторобудівників, 89-а/62, м. Харків, 61135
тел.:+38(050/093) 40-243-50
Свідоцтво про внесення суб'єкта видавничої справи до державного реєстру видавців, виготовників та розповсюджувачів видавничої продукції ДК № 4388 від 15.08.2012 р.
www.monograf.com.ua