

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»

Факультет комп'ютерних наук та технологій

(повне найменування факультету)

Кафедра комп'ютерних систем та мереж

(повне найменування кафедри)

Пояснювальна записка

до дипломного проєкту (роботи)

бакалавра

(ступінь вищої освіти)

на тему «Розробка системи оповіщення та контролю за безпекою
автотранспортного засобу»

(назва теми)

Виконав: студент 4 курсу, групи КНТ-510сп

Спеціальності _____

123 «Комп'ютерна інженерія»

(код і найменування спеціальності)

Освітня програма (спеціалізація)

Комп'ютерна інженерія

ШЕВЧЕНКО Д.О.

(ПРИЗВИЩЕ та ініціали)

Керівник

ІЛ'ЯШЕНКО М.Б.

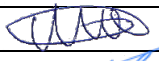
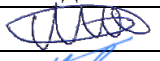
(ПРИЗВИЩЕ та ініціали)

Рецензент

КАЧАН О.І.

(ПРИЗВИЩЕ та ініціали)

6. Консультанти розділів проекту (роботи)

Розділ	ПРИЗВИЩЕ, ініціали та посада консультанта	Підпис, дата	
		завдання видав	прийняв виконане завдання
	ІЛЬЯШЕНКО М.Б., к. т. н., доцент		
Нормоконтроль	ЩЕРБАК Н.В., ст. викл.		

7. Дата видачі завдання «8» травня 2023 року.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломного проекту (роботи)	Строк виконання етапів проекту (роботи)	Примітка
	Аналіз технічного завдання	1 тиждень	виконано
	Аналіз структури роботи сигналізацій для автотранспорту	1 тиждень	виконано
	Вибір відповідної технології передачі інформації	1 тиждень	виконано
	Підбір комплектуючих елементів для створення системи, схеми підключення	2 тиждень	виконано
	Написання програмного коду	3 тиждень	виконано
	Проектування програмних засобів	3 тиждень	виконано
	Реалізація проекту	3 тиждень	виконано
	Проходження нормоконтролю	4 тиждень	виконано
	Захист роботи	4 тиждень	виконано

Студент(ка)


 (підпис)
Денис ШЕВЧЕНКО

(Ім'я ПРИЗВИЩЕ)

Керівник проекту (роботи)


 (підпис)
Матвій ІЛЬЯШЕНКО

(Ім'я ПРИЗВИЩЕ)

РЕФЕРАТ

ПЗ: 60 с., 20 рис., 3 табл., 24 джерел.

LORAWAN, ІНФОРМАЦІЙНА СИСТЕМА ,СУІБ, СИСТЕМА БЕЗПЕКИ

Метою дипломного проекту є розробка контролю цілісності автотранспортного засобу з функцією передачі даних стану бездротовою мережею LoRaWAN. Вибрано технологію бездротової передачі інформації, здійснено підбір комплектуючих елементів та здійснено реалізацію проекту з робочим програмним кодом. Розраховано основні параметри безпечної роботи в приміщенні та складено бізнес-план по об'єкту, що проектується.

Для виконання цього завдання розглянуто аналогічні системи сигналізацій, вибір технології бездротової передачі даних, підбір комплектуючих для забезпечення працездатності всіх необхідних функцій системи. Був написаний програмний код на C++ для платформи Arduino Uno, командами AT були задані параметри для роботи модулів передачі даних по LoRaWAN, а також код на мові Python для Raspberry Pi виконує функцію сервера.

ЗМІСТ

Вступ.....	6
1 Аналіз проблеми захисту автотранспорту у сегменті інформаційних систем	7
1.1 Сигналізація в автотранспорті	7
1.2 Поняття Інтернету речей	9
1.3 Бездротові технології Інтернету речей.....	13
2 Розробка системи оповіщення та контролю за безпекою автотранспортного засобу	20
2.1 Пристрій Інтернету речей.....	20
2.2 Сервер, що обробляє дані	38
2.3 Інтерфейс з користувачем.....	43
3 Реалізація проекту	45
3.1 Побудова моделі системи захисту інформації.....	45
3.2 Програмний код для модулів.....	49
Висновки	55
Перелік джерел посилання	56
ДОДАТОК А	58

Перелік графічного матеріалу:

Пл1 – Структура мікроконтролера;

Пл2 – Вибір протоколу передачі за характеристиками бездротових технологій;

Пл3 – Складання сигналізації, що встановлюється в автотранспорт;

Пл4 – Складання сигналізації, що виступає як сервер;

ВСТУП

На сьогоднішній день на покупку автомобіля можуть витрачатися значні суми грошей. Тому захист таких вкладень дуже актуальною темою. Існують різні способи зберегти цілісність транспортного засобу. Наприклад, вже на етапі виготовлення транспорту на заводі в нього монтуються відстежувальні датчики та сигналізації. Також автовласники набувають додаткового захисту у вигляді системи сигналізацій, що вбудовується. Сучасні системи повинні бути не лише прості у використанні, а й забезпечувати надійну охорону транспорту. З кожним разом охоронні системи стають все більш удосконаленими і мають різні доповнення. Це необхідно, оскільки зломщики та викрадачі автомобілів знаходять "недоліки" систем і використовують їх у своїх неправильних цілях.

Метою даної дипломної є розробка системи контролю цілісності автотранспортного засобу з функцією передачі даних стану за бездротовими технологіями. Для виконання цієї мети необхідно поставити завдання та їх реалізацію. Для початку необхідно провести аналіз існуючих систем сигналізацій та розібратися в їхній архітектурі та структурі. Так як передача даних буде здійснюватися за бездротовою технологією, то треба вибрати за якою, тобто провести аналіз та вибір конкретної бездротової мережі передачі інформації. Підбір обладнання для забезпечення цілісної передачі даних сприяє забезпеченню правильної роботи всіх пристроїв. Зрештою доцільно уявити реалізовану модель з повноцінною робочою системою.

1 АНАЛІЗ ПРОБЛЕМИ ЗАХИСТУ АВТОТРАНСПОРТУ У СЕГМЕНТІ ІНФОРМАЦІЙНИХ СИСТЕМ

1.1 Сигналізація в автотранспорті

Для початку розберемося з поняттям автомобільна сигналізація чи просто автосигналізація. Автосигналізація – це електронний пристрій, встановлений у транспорті та службовець для захисту автомобіля від угону, крадіжки частин самого транспорту або речей, що знаходяться усередині.

Сама сигналізація тільки інформує власника або оточуючих людей за допомогою звукових та/або світлових оповіщень, але не протидіє самому викраденню або крадіжці.

Сигналізацію можна класифікувати за деякими параметрами:

а) за видом зворотного зв'язку:

- 1) із зворотним зв'язком;
- 2) без зворотного зв'язку;

б) за способом передачі сигналу:

- 1) супутникові;
- 2) за допомогою GSM.

Моделі вищої цінової категорії вважаються сигналізаціями, що мають модулі GPS/GPRS. Їхня можливість полягає в керуванні функціями автосигналізації з мобільного пристрою через передачу SMS. До переваг наявності таких модулів можна віднести визначення розташування транспортного засобу за допомогою системи. Активна глобальна супутникова система, що надає доступ до місця розташування в усьому світі без обмежень. Технічні характеристики необхідні забезпечення працездатності системи наявність 24 супутників, що йдуть по круговій орбіті на висоті 19400 км з періодом звернення 11 годин 15 хвилин. Для визначення місцезнаходження потрібен зв'язок із чотирма супутниками [1].

Додаткові функції даної системи: дзвінки на певні номери телефонів,

заздалегідь встановлених, прослуховування салонів автомобілів, діагностичні повідомлення і т.д.

Найбільшу популярність набула система сигналізації з GPS, оскільки триває безперервний моніторинг автомобіля. GPS трекер є контролером, який встановлюється в машину, приймаючи і передаючи дані. Обробка даних відбувається через глобальні системи GPS. У контролері встановлюється звичайна SIM-карта з активованим номером. Ця система працює лише з тими номерами телефонів, які були зареєстровані раніше. Після підтвердження система починає в реальному часі надсилати повідомлення з даними про місцезнаходження автомобіля. Дані можна відстежувати як на мобільному пристрої, так і на персональному комп'ютері.

Докладно розберемо систему. У складові системи входять переважно 5 компонентів:

- приймач (GPS);
- супутниковий передавач;
- антена (вбудована та/або зовнішня);
- акумуляторна батарея;
- флеш пам'ять.

Дана система дуже проста і зручна тим, що, крім основних функцій, до неї можна прикріпити ще додаткові можливості. Наприклад, прослуховування салонів та оточення поблизу машини, тривожна кнопка дозволить в екстрених випадках відправити SMS на заздалегідь записаний номер телефону, резервна батарея допоможе сигналізації працювати, якщо запалення вимкнено, а акумулятор від'єднано, віддалене блокування двигуна, контроль активності автомобіля за допомогою сформованих таблиць та звітів.

Проте система дуже залежить від трьох складових: супутника, сервера обробки даних, оператора зв'язку. Правильна та стабільна робота безпосередньо залежить від них. Зазвичай проблеми виникають із боку супутникового чи стільникового зв'язку. Припустимо, автомобіль заїхав у тунель або якусь криту споруду і дані вже не зможуть передаватися належним чином. Ще однією

проблемою може бути глушники сигналів GSM, тому що припиняється робота з Інтернетом. Важливим фактором є наявність коштів на SIM-карті модуля передачі даних по стільниковому зв'язку. Незручність полягає у постійному контролі балансу на карті та правильності роботи операторів [2].

1.2 Поняття Інтернету речей

Інтернет речей (англ. internet of things, IoT) – мережа пов'язаних між собою фізичних пристроїв, які мають інтегровані технології, що забезпечують взаємодію із зовнішнім середовищем, приймають та передають дані. Простіше кажучи, це мережа, в якій люди можуть взаємодіяти з пристроями, а пристрої можуть спілкуватися між собою, реагувати на зміну оточення та приймати рішення без участі людини. IoT-пристрої працюють самостійно, хоча люди можуть налаштовувати їх або надавати доступ до даних. IoT-системи працюють у режимі реального часу і зазвичай складаються з мережі розумних пристроїв та хмарної платформи, до якої вони підключені за допомогою WiFi, Bluetooth чи інших видів зв'язку. Спочатку пристрої збирають дані - наприклад, про температуру в квартирі або частоту серцебиття користувача, потім ці дані відправляються в хмару. Там програмне забезпечення їх обробляє. Що станеться, якщо спекотного літа температура в кімнаті буде більшою, ніж та, в якій комфортно власнику квартири? Система автоматично ввімкне кондиціонер, і він буде працювати доти, доки не встановиться потрібний рівень температури.

Інтернет речей тісно пов'язаний із «розумним» будинком. Внаслідок технологій та пристроїв, розроблених відомими компаніями Google, Amazon, Apple та іншими люди можуть здійснювати онлайн-покупки, регулювати температуру в кімнаті, включати світло і музику, віддаючи голосові команди віртуальним помічникам. Більше не треба хвилюватися, що забули вимкнути вдома воду у ванній або газовій плиті. За допомогою простого телефону можна

контролювати дистанційно всю обстановку будинку і навіть навколо нього, наприклад, увімкнувши відео, яке записує камери відеоспостереження, встановлені по периметру будинку. У систему може бути вбудоване програмне забезпечення, яке за допомогою машинного зору контролюватиме, запам'ятовуватиме і розпізнаватиме надалі всіх, хто потрапляє в об'єктив камери. Якщо виявляться порушення, система здатна сама вжити необхідних заходів для забезпечення безпеки будинку.

IoT може охоплювати усі галузі. Наприклад, він може допомогти компаніям автоматизувати процеси та знизити трудовитрати. Це сприяє підвищенню якості послуг, що робить процес виробництва та логістику недорогим, мінімізує витрати на вантажоперевезення і виключає вплив людського фактора. Впровадження інтернету речей в електроенергетиці покращує контрольованість підстанцій та ліній електропередачі за рахунок дистанційного моніторингу, а в охороні здоров'я дозволяє перейти на новий рівень діагностики та лікування захворювань. У сільському господарстві «розумні» ферми та теплиці самі дозують добрива та воду, можуть контролювати потрібну температуру та вологість у приміщеннях, а також створювати штучне світло, необхідне рослинам та тваринам. Інтернет речей активно використовують нафтогазові та гірничодобувні галузі. Наприклад, застосування поглибленої аналітики з бурових свердловин допомагає нафтогазовій промисловості збільшити обсяги видобутку на вже відпрацьованих родовищах. А IoT у транспорті – це сам транспорт, електронні табло, навігатори, системи безпеки, камери спостереження, які взаємодіють між собою.

Основна проблема, з якою пов'язане стрімке зростання IoT – безпека. Кіберзлочинці постійно намагаються зламувати пристрої, де зберігаються бази даних людей, перехоплювати доступ до відеокамер, інтелектуальних систем керування автомобілем, здійснюють кібератаки, підвантажують віруси на зламані пристрої та навіть створюють масштабні аварії на виробництвах. Тому учасникам ринку IoT-ринку треба вміти захищати свої системи. Також залучення та як наслідок участь все більшої кількості пристроїв до Інтернету речей може призвести до втрати робочих місць. Наприклад, IoT-системи замінять фахівців з

технічного обслуговування, ремонту та встановлення обладнання [3].

Нинішній Інтернет включає державні, загальноорганізаційні та персональні комп'ютерні мережі. За допомогою протоколу IP (Internet Protocol) можна створити пристрої в одну єдину систему з різною топологією і різною архітектурою. Пристрій, що увійшов до мережі, автоматично отримує свою індивідуальну IP-адресу. Він може бути як статичним, і динамічним. Так само Інтернет речей включає в себе цілу низку систем, які пов'язані між собою і виконують певні функції. Наприклад візьмемо систему “розумного” будинку, де є кілька мереж, але кожна виконує своє завдання: автоматичне включення світла, управління безпекою, контроль систем охолодження і опалення та багато іншого. Непростим завданням є об'єднати всі ці мережі в одну мережу, оскільки вони можуть працювати за різними стандартами. Однією з проблем у наданні IP-адрес учасникам системи Інтернету речей була обмеженість в адресах. Так, наприклад, у четвертій версії протоколу IP (IPv4) можна використовувати лише 4,29 мільярда адрес. Вже до сьогодні відчувалася нестача цієї кількості. Цю складність допомогла вирішити шоста версія протоколу IP (IPv6). З подальшим глобальним використанням приладів Інтернету речей їм буде призначено адресу цієї версії. Таким чином, для IoT пристроїв можливо використовувати більше 340×1036 IP-адрес. З подальшим глобальним використанням приладів Інтернету речей їм буде призначено адресу цієї версії. Таким чином, для IoT пристроїв можливо використовувати більше 340×1036 IP-адрес. З подальшим глобальним використанням приладів Інтернету речей їм буде призначено адресу цієї версії. Таким чином, для IoT пристроїв можливо використовувати більше 340×1036 IP-адрес.

Один із ключових факторів у використанні Інтернету речей це технології, на яких він базується. В основі представлені наступні технології IoT:

можливість виявлення. Будь-який об'єкт, підключений до системи пристроїв, повинен ідентифікувати себе, інакше він просто не зможе бути учасником мережі. Пристрої можуть отримати особливі виявлення автоматично у різний спосіб: до кожного приладу додається радіочастотна мітка, інфрачервона мітка, оптична

(штрих-код, QR-код та ін). Але на забезпечення унікальності ідентифікаторів різних типів необхідно здійснити роботу з їхньої стандартизації;

засоби виміру. Метою технології є перетворення зібраної інформації про навколишнє середовище для подальшої передачі даних на пристрої обробки інформації. Цю роль можуть виконувати різні датчики: вологість, температура, освітленість і т.д., а також цілі вимірювальні комплекси. Для самостійної роботи даних пристроїв бажано забезпечити їхнє заряджання від альтернативних джерел енергії, наприклад, сонячних батарей, і не турбуватися про зміну батарей або акумуляторів протягом тривалого часу. Ця умова потрібна, так як більшість датчиків будуть працювати постійно і складно встежити за всіма, якщо в системі не кілька, а кілька тисяч приладів;

спосіб передачі. Передавання даних можна організувати будь-якою з доступних технологій. Якщо передача даних буде здійснюватися через провідні мережі, зазвичай використовують лінії електропередачі. Але якщо розглядати бездротовий спосіб, то особлива увага приділяється високій надійності та захисту даних при передачі;

метод обробки даних. В одній з великих компаній Microsoft вважають, що основна частина IoT – це не датчики та засоби передачі даних, а хмарні системи, що мають високу пропускну здатність та вміння негайно вживати заходів у певних ситуаціях. Припустимо, за показаннями датчиків з'ясовувати, що в будинку вже п'ять хвилин нікого немає, а входні двері залишилися відчиненими;

виконавчі устрою. Це такі пристрої, які можуть перетворити цифровий сигнал, отриманий від інформаційних мереж, і виконати певні дії, які від них очікується. Тобто, якщо власнику будинку захочеться підняти температуру на один градус, він може налаштувати потрібні параметри на телефоні і передати ці дані на пристрій, що змінює температуру в будинку [4].

Очікувані прогнози від Gartner (дослідницької та консалтингової компанії, що спеціалізується на ринках інформаційних технологій) вказують, що до 2021 року до інтернету буде підключено 25 млрд пристроїв. Кіберзлочинці продовжуватимуть атакувати їх, тому що IoT-система це швидкий спосіб

поширити шкідливе ПЗ. Прості користувачі, компанії та цілі міста все частіше застосовуватимуть інтелектуальні технології, щоб заощадити час і гроші. Наприклад, холодильники зможуть попереджати про швидке псування продуктів, світлофори із вбудованими відеодатчиками регулюватимуть дорожній рух залежно від трафіку. Один із нюансів впровадження Інтернету речей – це подача живлення на пристрої. Надалі необхідно передбачити автономність пристроїв, тобто мати можливість отримувати енергію із навколишнього середовища, без участі людини. Зараз, однак, ключова проблема застосування IoT - відсутність єдиних стандартів.

Так як на сьогоднішній день існує різноманітність пристроїв пов'язаних з Інтернетом речей з'явилася і проблема того, як системи повинні спілкуватися між собою, на якій мові має відбуватися розуміння один одного. І тому існує поняття як протокол IoT.

Протокол IoT – комплект стандартів для зв'язку пристроїв, їхньої правильної та ефективної функціональності [5].

1.3 Бездротові технології Інтернету речей

При створенні мережі виникає важливе питання, як будуть пов'язані всі пристрої між собою. Рішення залежить від завдань об'єкта. Основними параметрами при виборі проекту є:

- а) дистанція використання (поширення працездатності здійснюватиметься у межах офісу чи цілого міста);
- б) частота, при якій не буде перешкод та шумів, що впливають на роботу мережі;
- в) пропускна здатність необхідна для передачі даних та частота поновлення даних;
- г) подача енергії йде від мережі чи є акумулятори;
- д) безпека при передачі, обробці та зберіганні даних [6].

Кожні існуючі бездротові пристрої передачі даних мають такі параметри як швидкість, радіус дії та енергоефективність, але одночасно відповідати необхідним вимогам можуть лише дві характеристики з трьох (рисунок 1.1).

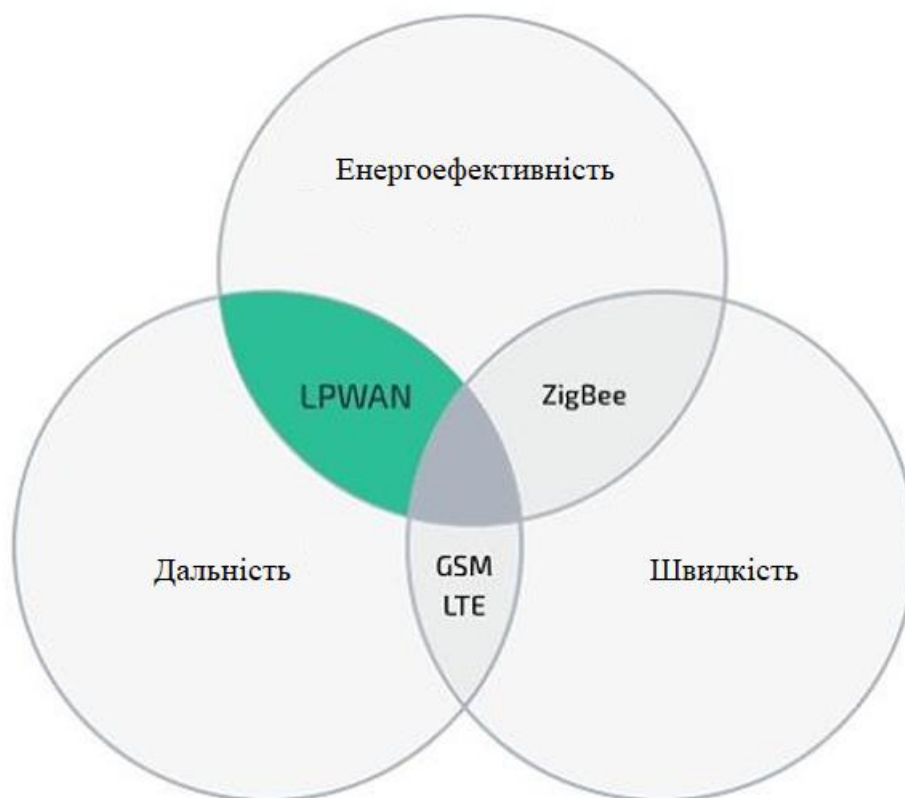


Рисунок 1.1 - Вибір протоколу передачі за характеристиками бездротових технологій

Вкрай важливо, щоб технологія була відкритим глобальним стандартом, а не запатентованою технологією, щоб гарантувати низьку вартість, а також максимально розширити вибір користувачів та постійні інновації.

Декілька технологій ближньої дії, зокрема Wi-Fi, Bluetooth і Zigbee, коштуватимуть набагато дешевше. Однак, будучи працездатними лише на малій відстані, вони не можуть забезпечити покриття, необхідне для таких додатків, як автомобілі, датчики стеження та багато інших (рисунок 1.2). Натомість вони обмежені комп'ютерами, підключеними до домашнього чи офісного середовища.



Рисунок 1.2 - Порівняння дальності роботи різних мереж

Так як автомобіль із сигналізацією не прив'язаний до одного місця, а вільно переміщається містом, а можливо і між містами, розберемо протоколи передачі даних з далеким радіусом дій. Однією з таких технологій є бездротова мережа LPWAN. Вона надає ефективну передачу даних великі відстані.

LPWAN (Low-power Wide-area Network; мережа широкого радіусу дії з низьким рівнем енергоспоживання) - технологія передачі даних з низьким енергоспоживанням, створена для бездротової передачі телеметрії між датчиками машинної інформації на великі відстані. Цей протокол був створений як рішення для Інтернету речей, що активно зростає. Для повної функціональності всіх пристроїв у мережі потрібний безпечний та ефективний спосіб передачі даних. LPWAN підходить за цими критеріями. В основі передачі даних по мережі LPWAN лежить принцип роботи мережі. Датчики, що приймають сигнал, передають його радіоефіром. Станції приймають та оцифровують сигнали і далі направляють їх на сервер. На серверах дані обробляються та передаються у звичному та зручному вигляді для користувачів. Зворотний канал зв'язку дозволяє контролювати систему з відривом [7].

На даний момент є кілька технологій, в основі яких лежить протокол LPWAN:

- Sigfox;
- LoRaWAN;
- Weightless;
- Ingenu;
- NB-IoT.

Технологія Sigfox була заснована в 2010 році з метою зв'язати кожен об'єкт у нашому фізичному світі з цифровим всесвітом. Два французькі засновники побудували глобальну мережу, присвячену Інтернету речей, засновану на малій потужності, великій дальності та невеликих даних, що пропонує наскрізну послугу підключення. В даний час Sigfox є у більш ніж 70 країнах.

Цей стандарт ґрунтується на наступних принципах:

- надання мінімального каналу для передачі невеликих повідомлень, який доповнюватиме інші протоколи зв'язку (WiFi, Bluetooth, супутниковий зв'язок, 3G, 4G, 5G);
- налаштування резервного каналу у разі збою основних каналів зв'язку або у випадках, коли мережа виходить з ладу після стихійного лиха чи зловмисної дії;
- підвищення та забезпечення безпеки мереж та обміну даних;
- скорочення енергоспоживання.

SigFox в даний час використовує найпопулярніший європейський діапазон ISM на 868 МГц, а також 902 МГц в США, залежно від певних регіональних правил. Система розгорнута з використанням можливостей сучасних стільникових мереж. Система здатна надсилати близько 140 повідомлень на день. Кожне повідомлення може використовувати 12 байт корисної інформації. У ці байти входять інформація про стан пристроїв, які відправляють відомості, наприклад, коефіцієнт енергоспоживання, місцезнаходження, сигналізація або інший вид сигналу про стан системи. Для збереження споживання енергії пристроєм, що передає сигнал, можна встановити надсилання повідомлень за певними діями або таймером, запрограмованим раніше. У такому разі система перебуватиме в сплячому режимі, поки її не “активує” якусь подія. Тоді вона передасть дані на

енергоспоживання, адаптованість в налаштуванні антени, кооперація з іншими діючими трансіверами. Головними недоліками можуть бути: низька швидкість передачі даних, залежне положення від стільникової інфраструктури, низька стійкість до перешкод [8].

Weightless – це технологія, що забезпечує бездротовий зв'язок для малопотужних глобальних мереж (LPWAN), спеціально розроблених для Інтернету речей. Ця мережа може працювати як у діапазоні нижче 1 ГГц, так і ліцензованому спектрі. Weightless SIG (Special Interest Group) – це некомерційна глобальна організація стандартизації, створена для координації дій, необхідних для надання кращої у світі технології підключення до IoT. Ця діяльність включає в себе розробку остаточної технології відкритого стандарту LPWAN для підключення до Інтернету, управління поточною еволюцією, інноваціями та оновленнями до стандарту та управління тестуванням, сертифікацією та ліцензуванням технології. До переваг відносяться: відкритий стандарт, далекий радіус дії, велика проникаюча здатність передачі сигналу в міській забудові, низьке енергоспоживання, підтримка особистих та громадських мереж, висока безпека. Мінусами даного протоколу є невисока швидкість передачі, недостатність доступного апаратного забезпечення, рідкісні оновлення документації [9].

NB-IoT (Narrow Band Internet of Things) – стандарт стільникового зв'язку для обладнання з невеликими обсягами передачі даних. Розроблений для з'єднання систем Інтернету речей до цифрових мереж зв'язку широкого діапазону, такі як датчики, що використовуються в медичних цілях, лічильників у будинках, конструкцій розумного будинку та ін. Ця мережа працює на приладах мережі та на обладнанні GSM. До плюсів NB-IoT відносяться: низька витрата енергії, підключення великої кількості пристроїв, низька вартість. Мінусами можна вважати недостатню підтримку мобільності, низьку можливість прийому передачі даних, при режимі енергозбереження потенційні значні затримки з'єднань у зв'язку з тим, що кінцевий механізм, перебуваючи в режимі низької витрати енергії, виявляється недоступним з боку мережі (сервера додатків).

LoRaWAN (Long Range wide–areanetworks, глобальна мережа великого радіусу дії) - основний апаратний протокол LoRa, який потрібен для контролю статусу та управління спілкуванням між LPWAN-шлюзами та кінцевими пристроями. Технологія ґрунтується на топології за типом "зірка". Найбільше пристроїв передають дані через бездротовий зв'язок не так на один шлюз, а одночасно кілька.

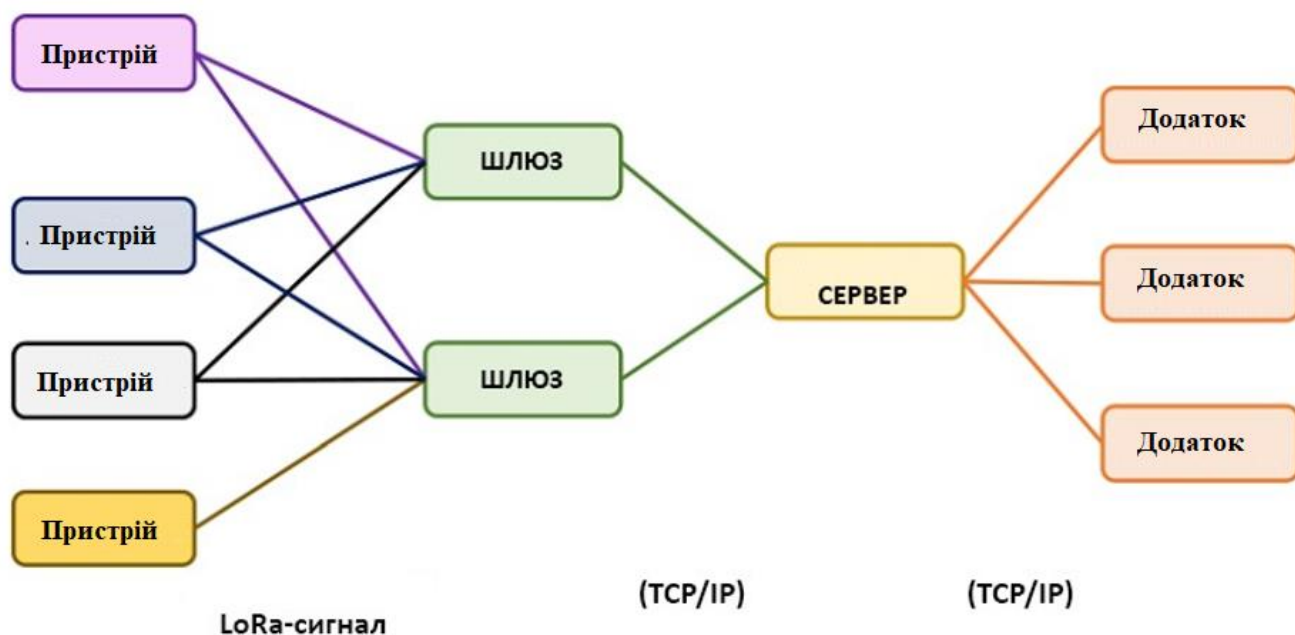


Рисунок 1.3 - Принцип роботи мережі LoRa

Це зручно тим, що при пересуванні пристрою, за яким йде стеження, він не має прив'язки до одного єдиного шлюзу. Значить, якщо об'єкт пішов з поля зору одного шлюзу, він автоматично відображається в зоні іншого. Цей метод дозволяє гарантувати цілісну передачу інформації [11] (рисунок 1.3).

2 РОЗРОБКА СИСТЕМИ ОПОВІЩЕННЯ ТА КОНТРОЛЮ ЗА БЕЗПЕКОЮ АВТОТРАНСПОРТНОГО ЗАСОБУ

2.1 Пристрій Інтернету речей

Систему, створювану в даній дипломній роботі, можна розділити на кілька складових частин: розробка інтерфейсу для зв'язку з користувачем, налаштування сервера та написання програми для обробки та зберігання отриманих даних, розробки IoT пристроїв, які прийматимуть інформацію з датчиків і передаватимуть на сервер, а після обробки виконувати команди, отримані з нього.

Пристрій Інтернету речей є засобом зв'язку між датчиком і сервером, а основною частиною пристрою є мікроконтролер. Сам собою мікроконтролер це мікросхема, з допомогою якої можна керувати різними пристроями (рисунок 2.1). Він направлятиме і контролюватиме потік інформації, зчитуючи дані з датчика і передаючи її далі через модулі зв'язку. Для проектування знадобиться навичка роботи з мікроконтролером, а саме його потрібно вибрати, написати програму, прошити мікроконтролер, підключити потрібні модулі, налагодити та протестувати програму. Для повноцінного функціонування мікроконтролера потрібно його правильно запитати, захистити від коротких замикань, підключити програматор до нього, стабілізувати напругу, підключити зовнішній частотний резонатор і світлодіод для зручного налагодження.



Рисунок 2.1 – Зовнішній вигляд мікроконтролера ATmega328p

Під час роботи мікроконтролер зчитує команди з порту введення або пам'яті та виконує їх. Кожна окрема команда характеризується системою команд, яка закладена в архітектуру мікроконтролера (рис. 2.2).

Розглянемо докладніше основні структурні елементи мікроконтролера. Головним елементом будь-якого мікроконтролера є центральний процесор (ЦПУ). Він має доступ до пам'яті, виконує обчислення, керує периферійними пристроями та обробляє переривання. Ядро процесора складається з арифметико-логічного пристрою (АЛП). Воно підтримує арифметичні та логічні операції між регістрами або між константою та регістром. Після виконання операції регістр стану оновлюється для відображення інформації про результат операції. АЛУ задіє регістри загального призначення (РОН, General Purpose Registers - GPR). Регістри загального призначення призначені звернення одночасно до оперативної пам'яті пристрою ОЗУ, де тимчасово зберігаються дані, і до виконання різних арифметичних, логічних операцій. Цей варіант роботи є особливістю мікроконтролера, тому що дозволяє йому працювати набагато ефективніше [14].

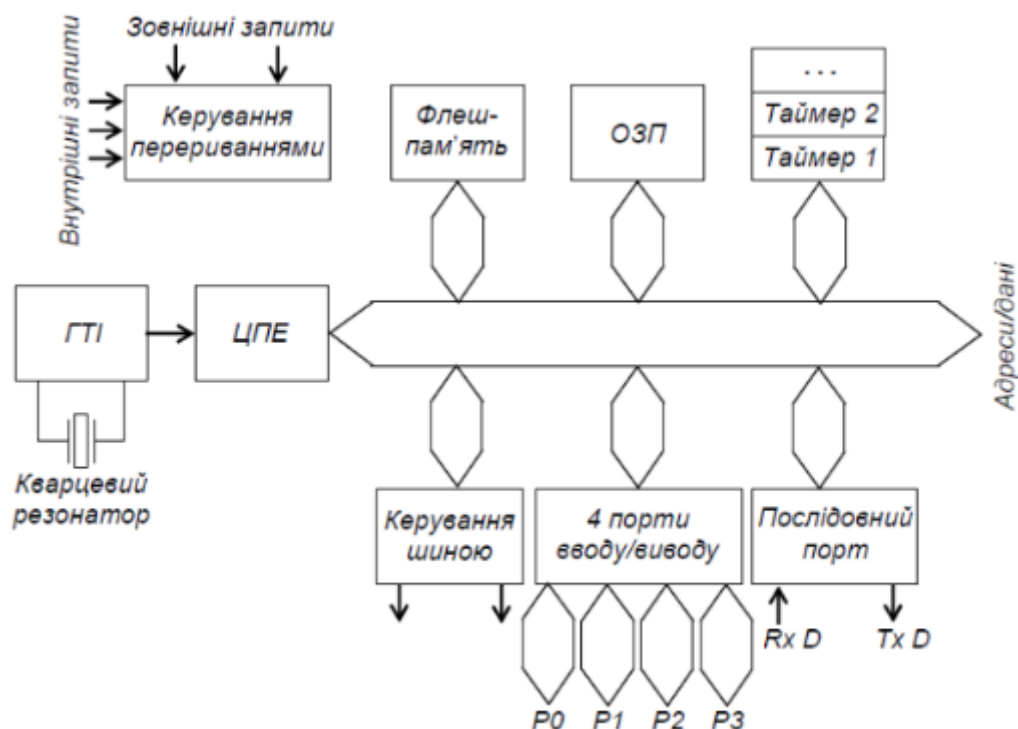


Рисунок 2.2 – Структура мікроконтролера

Якщо говорити про пам'ять мікроконтролера, її можна розділити на два види: пам'ять програми і пам'ять даних. Пам'ять програм - це постійна пам'ять (ПЗУ), необхідна для запису програмного коду і реалізується за технологією Flash-пам'яті. Є незалежним, тобто при відключенні живлення програмний код, записаний раніше, збережеться і при включенні пристрою почне працювати за заданим алгоритмом. Пам'ять даних – призначена зберігання даних, задіяних програмами. До них можуть належати різні таблиці, графіки, константи. Вона буває двох видів: оперативна пам'ять (RAM) та енергонезалежна пам'ять даних EEPROM. На відміну від EEPROM, де інформація може зберігатися у разі відключення живлення на пристрій, всі дані, записані раніше на ОЗП, будуть стерті.

РОН пов'язаний з лічильником команд, в якому зберігається адреса програми, що виконується на даний момент. Під час виконання коду значення лічильника збільшується. Основне призначення лічильника команд – запам'ятовування поточної адреси виконання програми у разі переривання роботи мікроконтролера, а щодо відновлення його працездатності, продовження обробки коду з останнього місця зупинки. Тому робота лічильника пов'язана з Flash-пам'яттю мікроконтролера. До цієї пам'яті може також звертатися регістр команд. Регістр команд – регістр, у якому тимчасово зберігається код операції. Існує також декодер команд, який конвертує програмний код з регістру команд у відповідні сигнали керування для виконання інструкцій. У регістрі стану міститься інформація про результат останньої виконаної арифметичної інструкції.

У мікроконтролері бувають інші допоміжні пристрої, звані периферією. До них можна віднести аналоговий компаратор, таймери, АЦП, ЦАП, порти вводу/виводу.

Порти входу/виходу – периферійний елемент до пін яких можна підключити різні датчики та пристрої, що виконують.

Аналогово-цифровий перетворювач (АЦП) необхідний перекладу аналогового сигналу в цифровий на вході передачі. Зворотну функцію виконує цифро-аналоговий перетворювач (ЦАП), конвертуючи сигнал з цифрового вигляду

в аналоговий.

Блок переривань дозволяє мікроконтролеру контролювати певні події у фоновому режимі реагуючи на сигнали, що надходять з вхідних елементів або за якоюсь внутрішньою подією. При кожному окремому перериванні формується та зберігається окрема підпрограма на згадку про мікроконтролера. При перериванні, якщо необхідно, зупиняється виконання основного коду програми і пріоритет ставиться виконання підпрограми. До кожної події може бути заданий рівень пріоритету самостійно. Після виконання завдань підпрограми відбувається поновлення роботи основної програми завдяки лічильнику команд, який зберіг попередні значення.

Послідовний інтерфейс SPI (Serial Peripheral Interface) здійснює передачу даних між різними пристроями, наприклад цифровий потенціометр. У цьому випадку визначається тип ведучого (Master) пристрою та веденого (Slave). Після цього Master робить запит на передачу даних. Передача даних ділиться на три провідники: від веденого до провідного MISO (Master Input Slave Output), від ведучого до провідного MOSI (Master Output Slave Input) та тактовий сигнал CLK (Clock). Програмування мікроконтролера також здійснюється через інтерфейс за допомогою підключення до нього програматора.

В одне із завдань мікроконтролера входить порівняння двох аналогових величин. Цим процесом займається аналоговий компаратор. При надходженні двох аналогових сигналів на даний елемент воно порівнює їх і результат порівняння записує певної комірки пам'яті. Надалі цей сигнал можна проаналізувати та подати команду для виконання подальших функцій. Наприклад, при вимірюванні напруги на блоці живлення було зафіксовано низьке значення і за допомогою аналогового компаратора запустилася індикація оповіщення червоним світлодіодом.

Модулі таймера та лічильника використовуються для виконання операцій синхронізації або підрахунку мікроконтролера. До них відносяться позначки часу, інтервали та затримки вимірів, підрахунок подій тощо.

Внаслідок описаного вище мікроконтролер – це пристрій, за допомогою

якого можна створити прилад керування. Написавши програмний код, можна отримати доступ до контролю складових елементів та створення алгоритмів роботи всередині мікроконтролера.

Мова програмування мікроконтролерів залежить від пам'яті. Зазвичай ця цифра значиться від 2 до 128 Кб. Здебільшого код пишеться мовою C, C++. Але якщо ця цифра менша, то використовуються низькорівневі мови програмування, наприклад асемблер.

Мікроконтролери зазвичай не використовуються власними силами. До них підключають різні монітори або екрани, клавіатуру, датчики для отримання інформації та виконання наступних команд. Як готова платформа, де вже передбачені всі тонкощі роботи з мікроконтролером, розглянемо платформу Arduino UNO (рисунок 2.3).

Плата Arduino дуже зручна у створенні технологій IoT, оскільки має можливість зчитування вхідних даних, наприклад, світло на сенсорі, відбиток пальця, повідомлення з телефону, і конвертує його у вихідний сигнал – активувати мотор, закрити двері, увімкнути світлодіод тощо. Можна встановити команди платі що робити, написавши набір інструкцій мікроконтролеру. Для цього використовується мова програмування Arduino та програмне забезпечення IDE. Завдяки простому і доступному інтерфейсу користувача Arduino використовується в тисячах різних проектів і додатків. Програмне забезпечення Arduino є простим у використанні для початківців, але досить гнучким для досвідчених користувачів. Воно підтримує роботу в операційних системах Mac, Windows та сімейства Linux.

Всі плати Arduino повністю відкриті, тому користувачі мають можливість самим доопрацьовувати пристрої самостійно та адаптувати до контрактних потреб. Програмне забезпечення теж має відкритий вихідний код, завдяки чому ним можуть користуватися люди в усьому світі.

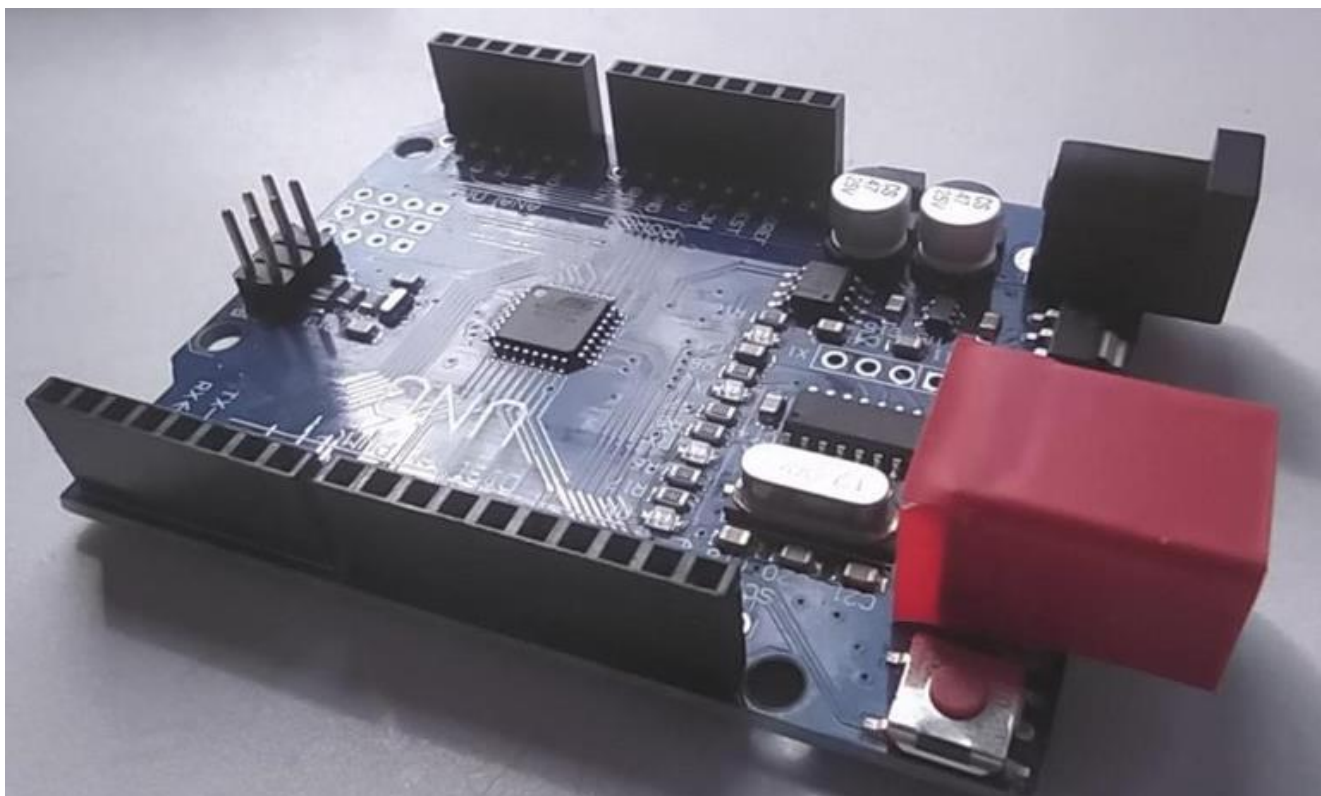


Рисунок 2.3 – Arduino UNO

Переваги плати Arduino:

- кросплатформність. Програмне забезпечення Arduino (IDE) працює в операційних системах Windows, Macintosh OS X і ОС сімейства Linux, тоді як більшість IDE, що спеціалізуються на мікроконтролерах, таких як Keil uVision, Atollic TrueSTUDIO, eclipse, обмежені Windows;
- простота використання. Зрозуміле середовище програмування – Arduino Software (IDE) проста у використанні для новачків, але досить гнучка для досвідчених користувачів;
- програмне забезпечення з відкритим вихідним кодом і програмне забезпечення, що розширюється. ПЗ випускається у вигляді інструментів з відкритим вихідним кодом, доступних для розширення досвідченими програмістами. Мова може бути розширена за допомогою бібліотек C++, і люди, які бажають зрозуміти технічні деталі, можуть перейти від Arduino до мови програмування AVR C, на якій він заснований;
- ціна. Плати Arduino відносно недорогі, порівняно з іншими платформами

мікроконтролерів. Найдешевша версія модуля Arduino може бути зібрана вручну [15].

На платі Arduino Uno є сам мікроконтролер, порти вводу/виводу, роз'єм USB, роз'єм живлення від зовнішнього джерела, що інформують світлодіоди, кнопка скидання (рисунк 2.4).

Подати живлення на плату можна двома способами: через порт USB або через порти GPIO. При включенні від зовнішнього джерела живлення через GPIO дроти підключаються до висновків “GND” “VIN”. Подача напруга повинна бути в діапазоні від 7 до 12 В. Якщо напруга буде менше, то платформа може нестабільно працювати, що призведе до поломки пристрою. На випадок проходження великих струмів або короткого замикання через роз'єм USB у платформі передбачено запобіжник.

Він спрацює за наявності струму близько 500 мА і розімкне ланцюг до тих пір, поки не відновиться нормальне значення струму.

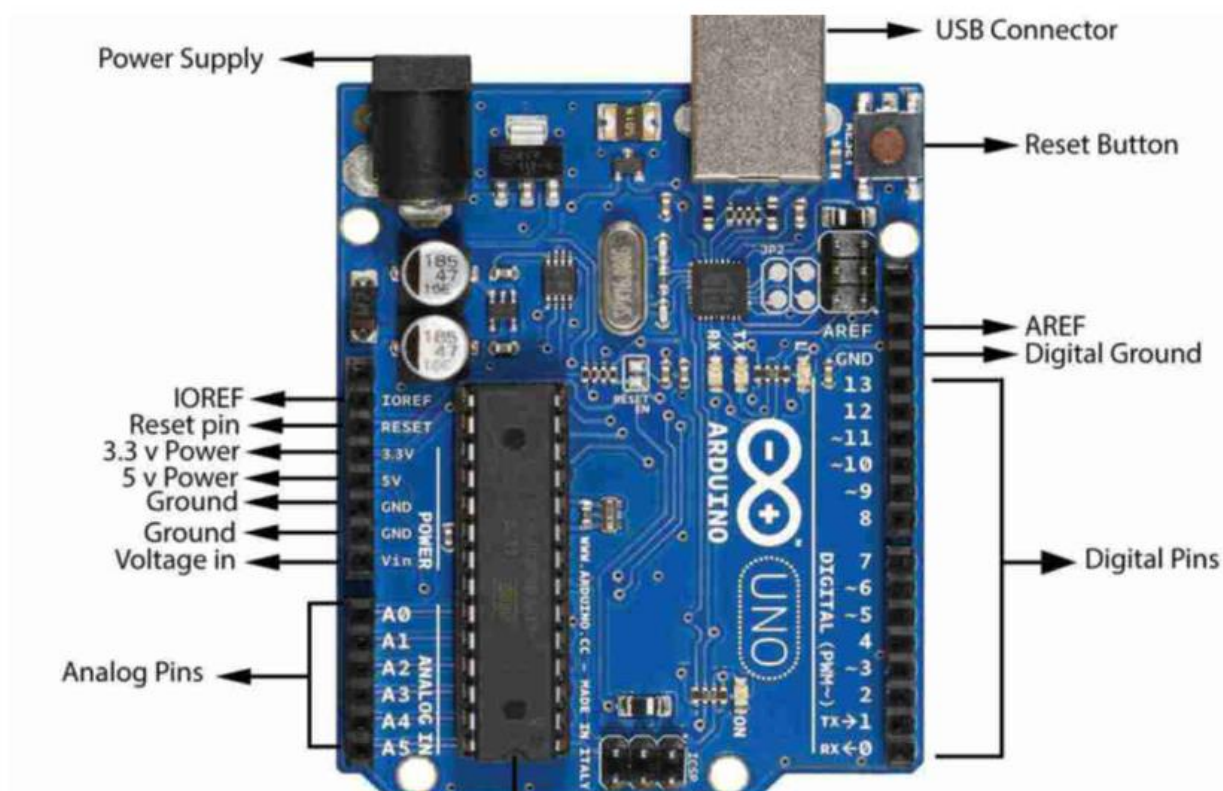


Рисунок 2.4 – Розташування елементів на платі Arduino UNO

У таблиці 2.1 наведено технічні характеристики Arduino UNO. Функції портів групи “Power” на платі:

- VIN – одержання напруги від зовнішнього джерела живлення;
- 5V – видача напруги до 5 В, використовується для живлення зовнішніх пристроїв, що підключаються;
- 3V3 – видача напруги до 3,3 В, використовується для живлення зовнішніх пристроїв, що підключаються;
- GND - загальний висновок заземлення;
- IOREF – використовується для інформування про робочу напругу для пристроїв та автоматичне перемикання на потрібне джерело живлення 5 В або 3,3В.

Плата підтримує три види пам'яті, що вбудовані в сам мікроконтролер: Flash, ОЗУ, EEPROM. Flash-пам'ять має об'єм 32 Кб для зберігання коду програми. В оперативній пам'яті зберігаються змінні, які створюються в процесі виконання коду, але після втрати живлення видаляються. Об'єм даної пам'яті становить 2 Кб. На відміну від ОЗУ, змінні в пам'яті EEPROM, об'ємом 1 Кб, не втрачаються при вимкненому живленні. Головний недолік цієї пам'яті – обмеженість перезапису даних, за твердженням виробника туди можна записати 100 000 разів. Для роботи з цією пам'яттю необхідно встановити спеціальні бібліотеки, доступні в Arduino IDE.

Цифрові порти вводу/виводу на платі пронумеровані числами від 0 до 13. Кожен із цих портів має своє призначення. На пінах 0 та 1 налаштований послідовний асинхронний інтерфейс UART передачі даних через висновки RX та TX відповідно. Піни 2 та 3 налаштовані на виклик переривання за певним сигналом. Піни з номерами 3,5,6,9,10,11 використовуються як ШІМ, тобто виведення аналогових значень цифровими сигналами. На висновках 10 (SS), 11 (MOSI), 12 (MISO), 13 (SCK) налаштований периферійний послідовний інтерфейс SPI. До виходу 13 підключений світлодіод, який на платі підписаний буквою L, що інформує рівень сигналу.

Таблиця 2.1 - Технічні характеристики

Характеристика	Значення
Мікроконтролер	ATmega328
робоча напруга	5 В
Вхідна напруга	7–12 В
Кількість цифрових вводів/виходів	14
Кількість аналогових вводів	6
Постійний струм, що подається на порти вводу/виводу	20 мА
Flash-пам'ять	32 Кб
ОЗУ	2 Кб
EEPROM	1 Кб
Світлова індикація подій	Обмін даними (RX, TX), подача живлення, цифровий висновок 13

Для роботи з аналоговими пристроями на платі є спеціальні аналогові піни, що нумеруються як А0-А5. Додаткові функції виконують висновки А4 (SDA) та А5 (SCL). За допомогою них здійснюється зв'язок з пристроями через два дроти за синхронним протоколом I2C.

Додатковий вихід AREF на платі застосовується для налаштування опорної напруги аналогових сигналів. Вихід RESET здійснює функцію перезавантаження контролера при низькому сигналі з його пін [16].

Програмування платформи здійснюється у спеціальній програмі на ПК.

Для цього на комп'ютер необхідно завантажити програму Arduino IDE. Після відкриття програми в налаштуваннях вибрати плату, з якої відбувається робота. Мова програмування платформи Arduino пишеться мовою C++. Зручність роботи у цьому середовищі обумовлено наявністю готових бібліотек застосування різних функцій платформи. Запис коду на платформу здійснюється шляхом підключення плати до комп'ютера через USB.

Оскільки у цій роботі було обрано бездротова технологія LoRa, то вибір модуля для приєма–передачі даних будувався цьому головному пункті – підтримка стандарту передачі LoRaWAN. Як пристрій було обрано модуль RYLR896 (рисунок 2.6).

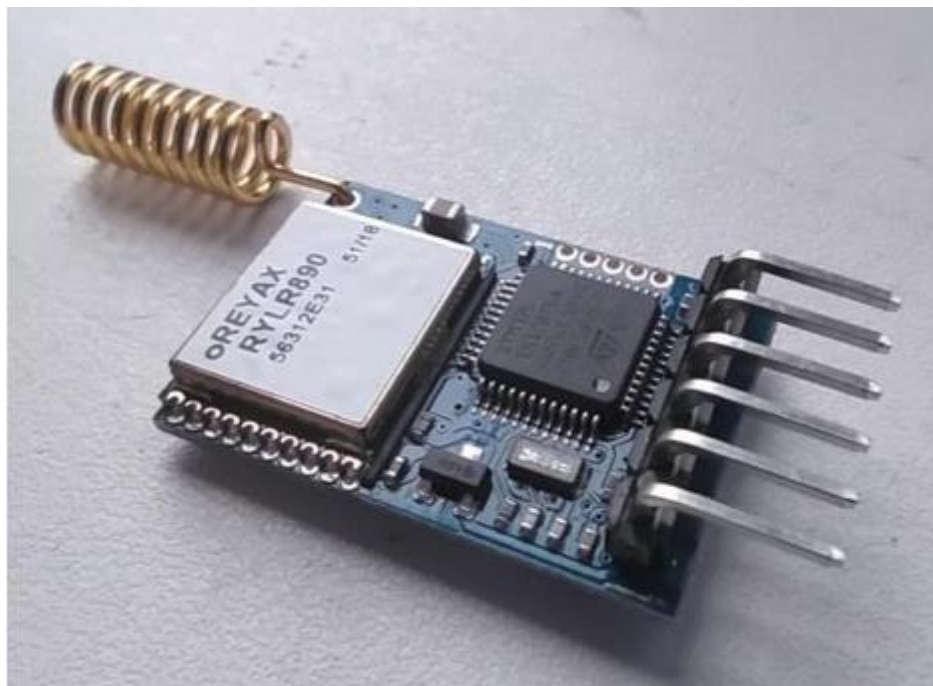


Рисунок 2.6 – Модуль RYLR896

Основну зручність модуля становлять наявність вбудованих вже в плату приймально-передаючого чіпа SX1276 та антени. На платі встановлено мікроконтролер STM32L, який реалізує протокол взаємодії трансівером SX1276 за стандартом SPI. Також від мікроконтролера відходять висновки UART інтерфейсу з протоколом, що базуються на AT-командах. Розташування пінів зображено рисунку 2.7, а призначення висновків у таблиці 2.2.

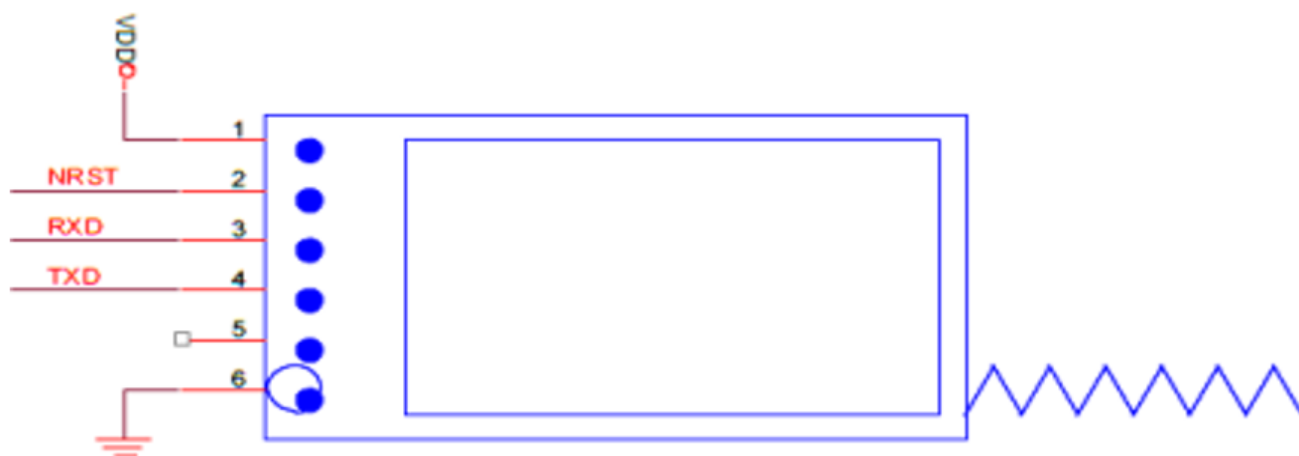


Рисунок 2.7 – Розпинання модуля

RYLR896 Таблиця 2.2 – Призначення висновків

№	Назва	Функції
1	VDD	Виведення джерела живлення
2	NRST	RESET (за низького значення напруги)
3	RXD	Прийом даних UART
4	TXD	Передача даних по UART
5	-	-
6	GND	Виведення заземлення

До функцій приладу належать шифрування даних алгоритмом AES128. Режим роботи передбачає, що у приймачі і передавачі встановлюються ключі і якщо вони збігаються, то приймач приходить неспотворені повідомлення.

Живлення на модуль розраховане на 3.3 В. На платі немає вбудованих стабілізаторів, тому при подачі більшого значення напруги просто приведе її до неробочого стану. Пристрій може бути в трьох режимах роботи: режим прийому, передачі і сплячому, при цьому споживанні енергії у них різні. Споживаний струм у режимі прийому даних становить 16.5 мА, у стані передачі - 43 мА, а в режимі сну - 0.5 мкА. Робочий діапазон частот варіюється від 820 МГц до 1020 МГц. Дальність роботи складає 15 км. Треба врахувати, що дана дистанція буде актуальною, якщо передача даних буде йти в ідеальних умовах по прямій без перешкод і перепадів висоти. У міській забудові дана довжина знижується до 4,5 км. У модуль вбудована пам'ять EEPROM, яка дозволяє запис та стирання до 300000 разів.

Кожен модуль необхідно налаштувати для того, щоб приймаючий та передавальний пристрої "спілкувалися" тільки між собою. Для керування модулем використовуються набір AT-команд. Послідовність використання налаштувань:

– команда "AT+ADDRESS". Використовується при логічній ініціалізації пристроїв для прийому та передачі даних. Кількість адрес можливо від 0 до 65535 (за замовчуванням використовується 0). Цей запис зберігається у пам'яті EEPROM;

– команда "AT+NETWORKID". Використовується для групування

пристроїв у мережі LoRa. В одній мережі може бути до 65 536 пристроїв. Якщо ID у приймаючого пристрою буде відрізнятися від ID передавального, дані не зможуть дійти до адресата. Діапазон мереж може бути від 0 до 16, проте 0 зазвичай використовується як публічна адреса. Цей запис також зберігається у пам'яті EEPROM;

– команда "AT+BAND". У цьому налаштуванні задається центральна частота передачі даних. На обох пристроях (приймачі та передавачі) повинна бути виставлена годин і той же діапазон частот. Діапазон робочих частот варіюється від 820 МГц до 1020 МГц. Неліцензована частота, де може працювати модуль, становить 868 МГц;

– команда "AT+PARAMETER". Ця команда використовується для налаштувань 4 параметрів: коефіцієнт поширення, пропускна здатність, швидкість кодування, довжина преамбули. Так само, як і для інших налаштувань, дана конфігурація повинна бути налаштована на пристроях, що передає і приймає, однаково. При повторному увімкненні пристрою жодна з наступних установок не збережеться в пам'яті модуля.

Коефіцієнтом поширення можна регулювати ступінь розширення спектру від 7 до 12. Відповідно, чим вищий коефіцієнт, тим вища чутливість і більша дальність. Але з цього збільшується час передачі пакета даних (рисунк 2.8). За промовчанням використовується коефіцієнт 12.

Параметром пропускної спроможності характеризується смуга частот, що використовується для прийому передачі даних. Значення цього параметра впливає на чутливість і час, чим смуга частот вже, тим довше відбувається передача даних (рисунк 2.9). За замовчуванням використовується смуга частот 125 КГц.

Бездротова мережа LoRa використовує набори алгоритмів сигнальної обробки. Одним з таких алгоритмів є алгоритм пошуку та усунення помилок у прийнятих спотворених пакетах. На прилад приймача приходить корисна, а й надмірна інформація. Швидкість кодування використовується для обробки надлишкових даних. Параметр набуває значень від 1 до 4, де 1 – 25% надлишкових даних, 2 – 50%, 3 75%, 4 – 100%. Звичайно це позначається на часі

передачі (рисунок 2.10). За промовчанням стоїть параметр 1.

Значення довжини преамбули дозволяє зменшити втрату даних, тобто, чим вищий коефіцієнт, тим менша можливість втрати, тим довше часу передачі (рисунок 2.11). За замовчуванням використовується значення 4; команда "AT+SEND". Дозволяє надіслати пакет із даними іншому передавачу. У цій команді необхідно вказати адресу одержувача, довжину рядка, що передається, і сам рядок. Протокол AT- команд, вбудований у модуль передачі даних, базується на передачі саме рядків, а чи не двійкових даних.

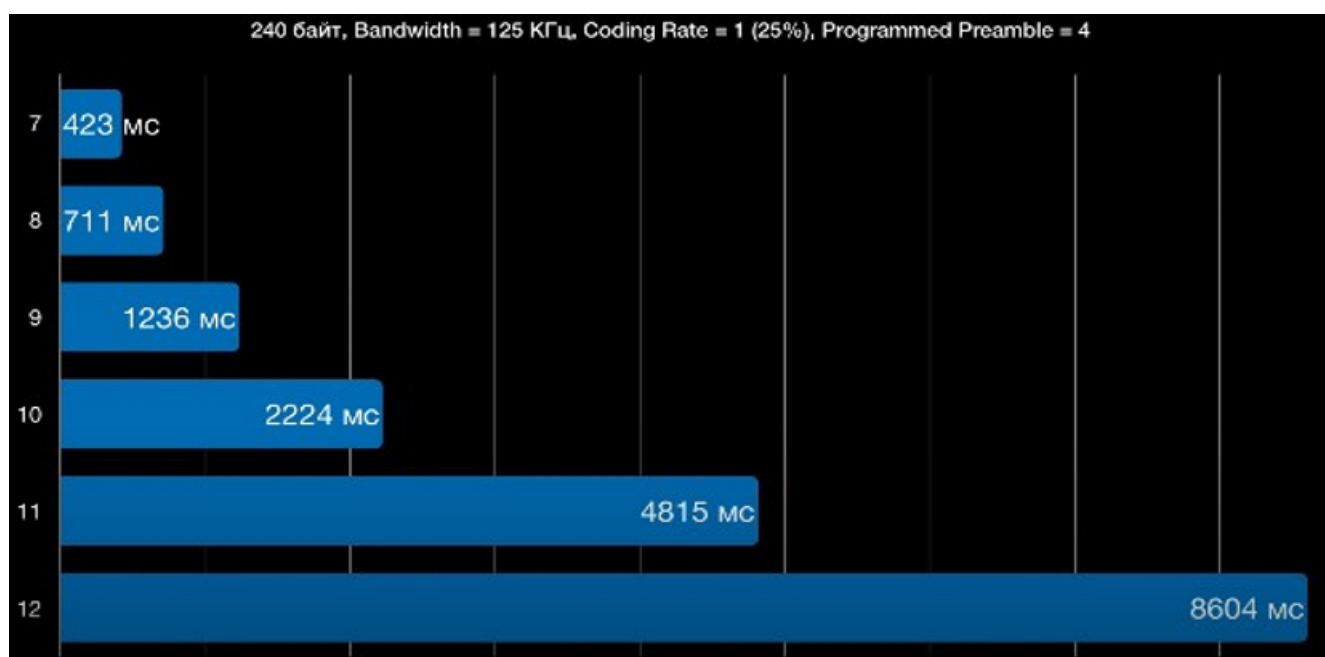


Рисунок 2.8 – Значення часу, що залежать від коефіцієнта розповсюдження

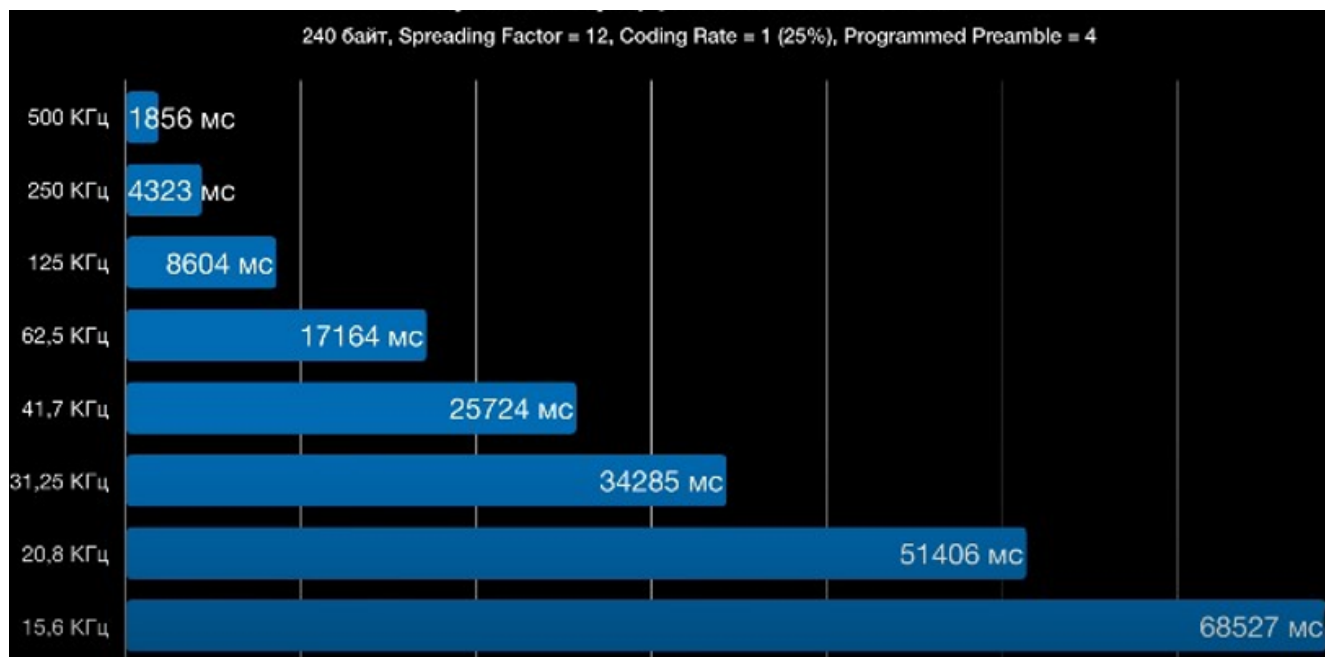


Рисунок 2.9 – Значення часу, що залежать від пропускної спроможності

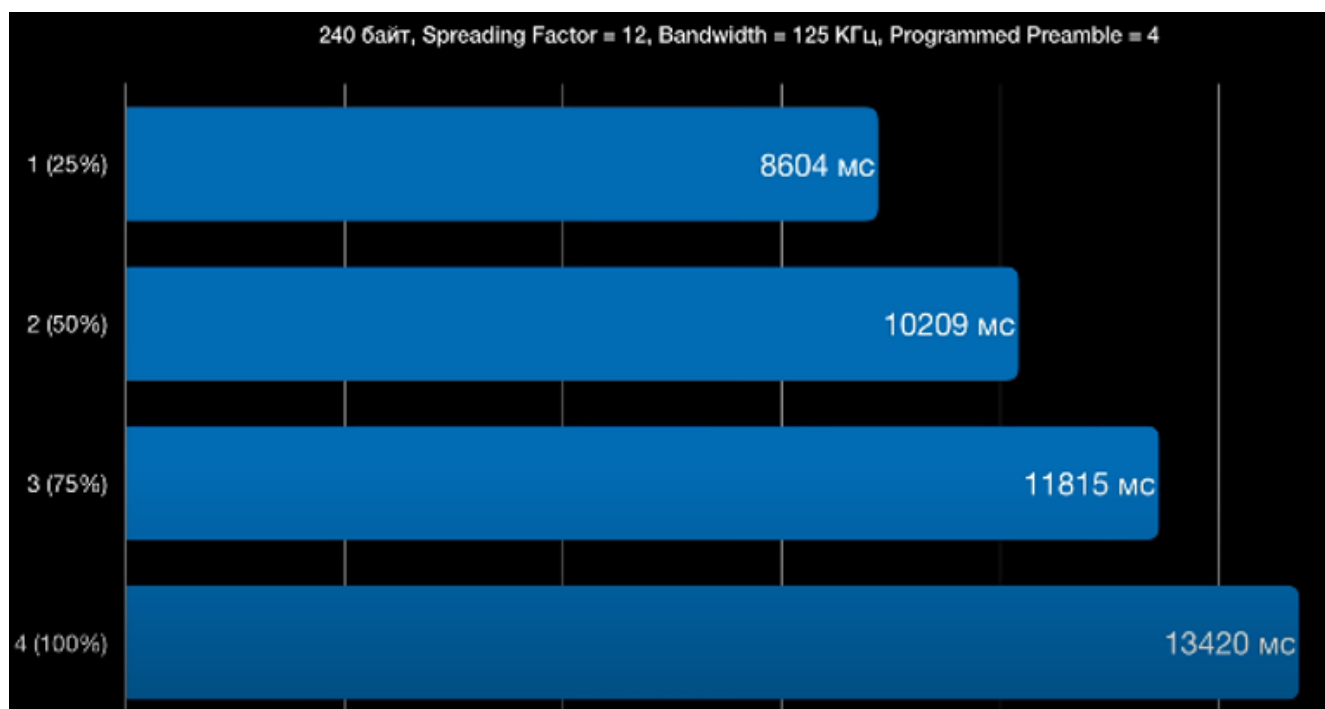


Рисунок 2.10 – Значення часу, що залежать від швидкості кодування

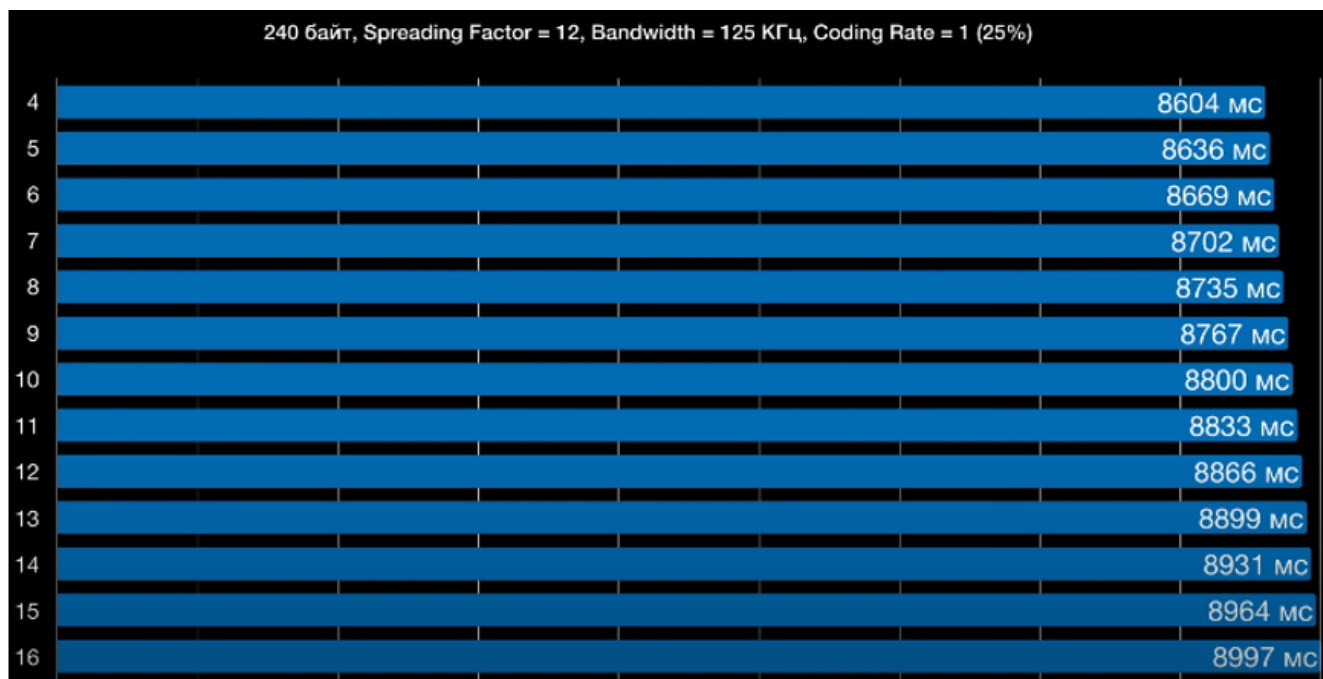


Рисунок 2.11 – Значення часу, що залежать від довжини преамбули

Команда "+RCV". Коли радіомодуль отримує дані, то на UART відправляє наступний рядок, в якому відображається адреса модуля, що відправив, довжина рядка, сам рядок і числові параметри RSSI і SNR. RSSI – рівень сигналу прийому. Негативна величина, що вимірюється в децибелах, чим ближче вона до нуля, тим краще і вищий рівень сигналу. SNR - відношення сигнал-шум [17].

Як згадувалося раніше, модуль вбудований чіп SX1276 (рисунок 2.12). Він включає модем з розширеним спектром LoRa, який здатний досягати значно більшої дальності, ніж існуючі системи. Живлення чіпа походить від напруги 3,3 Ст.

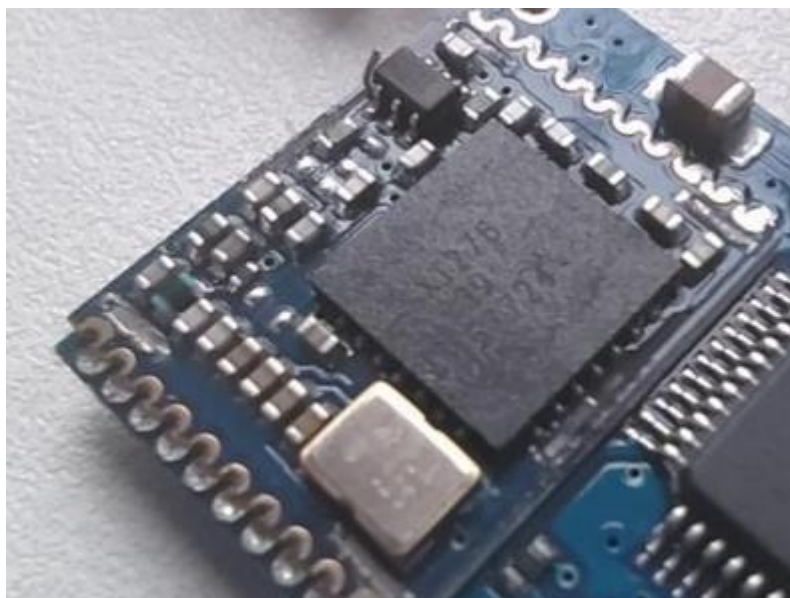


Рисунок 2.12 – Трансівер SX1276

Підключення модуля до Raspberry Pi призвело до необхідності включення до системи додаткового пристрою. Так як на сьогоднішній день в основному всі сучасні комп'ютери взаємодіють із зовнішніми пристроями за допомогою USB інтерфейсу. Однак у мікроконтролерів найпоширенішим інтерфейсом є UART. Для того, щоб сполучити між собою персональний комп'ютер із мікроконтролером існують спеціальні перетворювачі. Один із таких перетворювачів є перехідник FT232RL. Як видно на рисунку 2.13, з одного боку, у пристрою висновки для підключення до мікроконтролера, а з іншого висновки для підключення USB. В основі функціональності даного приладу лежить мікросхема FT232R, яка виконує всі необхідні завдання взаємозв'язку двох інтерфейсів.



Рисунок 2.16 – Схема підключення пристроїв із живленням мікроконтролера

Після підключення перетворювача до комп'ютера пристрій буде визначено, але не класифіковано належним чином. Для усунення цієї проблеми необхідно завантажити драйвер на ПК з офіційного сайту виробника. За підсумками установки драйвера, обладнання сприйматиметься комп'ютером і повноцінно функціонуватиме, виконуючи свої завдання.

2.2 Сервер, що обробляє дані

Всі дані, які надходять на модуль RYLR896, повинні бути оброблені, записані в базу даних і використані в подальшому. Пристрій, що виконує це завдання, називають сервером. Як сервер може використовуватися пристрій Raspberry Pi 3 B+, повноцінний одноплатний мінікомп'ютер (рисунок 2.17). У даному проекті Raspberry Pi, використовуючи модуль RYLR896, отримуватиме від системи сигналізації пакети даних, обробляти їх, конвертуючи пакет даних у людину-читану інформацію та відправляти безпосередньо користувачеві.

Серцем міні комп'ютера є 64-бітний 4-х ядерний процесор ARM Cortex-A53, з тактовою частотою 1,2 ГГц на ядро (рисунок 2.18). Також він має графічний 2-х ядерний процесор VideoCore IV, який підтримує стандарти OpenGL 2.0, OpenVG, MPEG-2, VC-1, а також здатний кодувати і декодувати, виводити у Full HD відео з частотою 30 кадрів в секунду. І процесор і графіка об'єднані в одному чіпі Broadcom BCM2837.

Raspberry Pi має 1 Гбайт оперативної пам'яті. Ця пам'ять ділиться із графічною підсистемою. Операційна система встановлюється на карту пам'яті microSD, слот для якої вбудований у плату. Мінімальний розмір карти пам'яті 4 Гбайт, рекомендується використовувати карти 10 класу швидкості. Поряд із слотом карти пам'яті знаходиться контролер бездротових інтерфейсів Wi-Fi та Bluetooth, які також об'єднані в одному чипі Broadcom BCM43438.



Рисунок 2.17 – Raspberry Pi 3 B+

Міні комп'ютер має різні інтерфейси для підключення периферії. На платі є 4 роз'єми виходу USB 2.0. Поруч із ними розташований мережевий адаптер Ethernet, що працює зі швидкістю 100 Мбіт на секунду. За їхню роботу відповідає USB Хаб LAN 9514-JZX. Для підключення навушників та колонок передбачено роз'єм Jack 3,5 мм. Він також поєднаний з композитним RCA-відеовимодом і може використовуватися для виведення відео у форматах PAL або NISC. HDMI вихід для виведення графіки на монітор чи телевізор. Роз'єм microUSB для підключення живлення. Живиться плата від 5 В, струм не менше 2 А (рекомендована сила струму блоку живлення, що видається, становить 2,5 А). Поруч розташовуються

два світлодіоди: червоний показує наявність живлення, зелений – звернення до microSD. Також на платі розпаяно два роз'єми CSI-2 для підключення камери та DSI для підключення дисплея.

На лицьовій стороні плати знаходяться 40-піновий роз'єм GPIO (інтерфейс введення/виведення загального призначення). Він необхідний для підключення зовнішніх пристроїв та взаємодії з ними. Протоколи спілкування: UART (Serial), I2C/TWI, SPI із селектором між двома пристроями. Більш детальну розпинку висновків GPIO можна побачити в таблиці 2.3

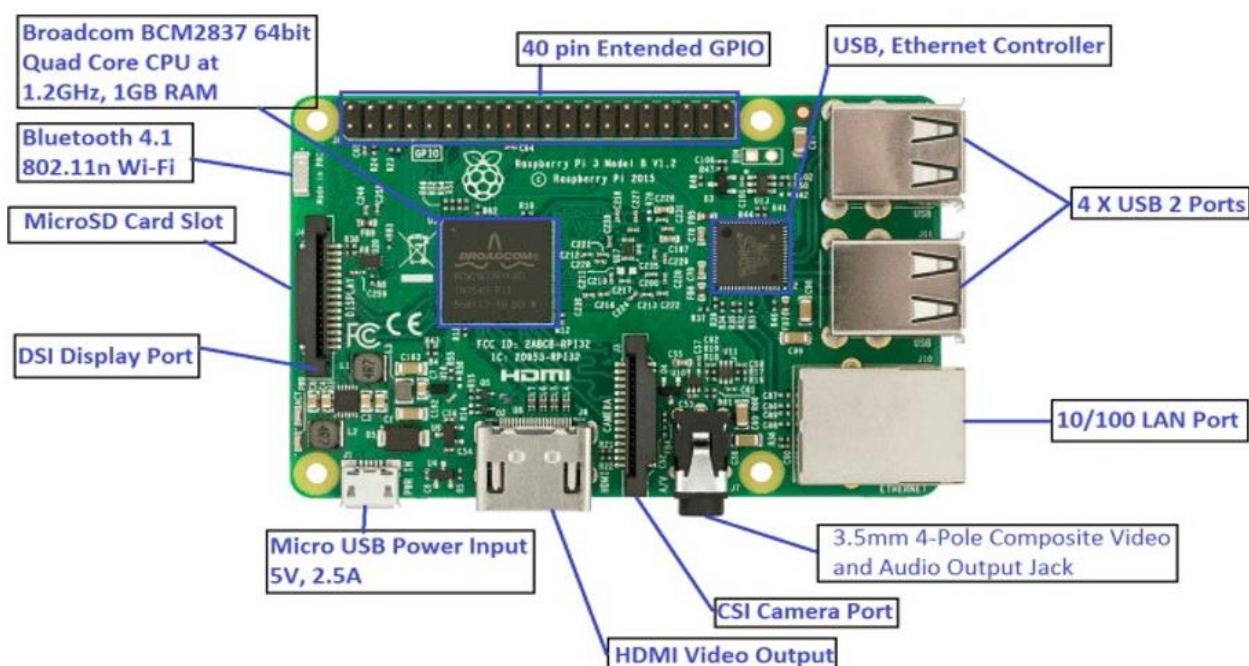


Рисунок 2.18 – Розташування елементів Raspberry Pi

Для роботи з міні-комп'ютером необхідно встановити на неї операційну систему. Операційна система, що рекомендується, використовується для Raspberry Pi - Raspbian. Ця безкоштовна операційна система на основі Debian. Щоб завантажити образ операційної системи, необхідно записати його на microSD карту через інший комп'ютер із пристроєм для читання SD-карт. Сам образ можна завантажити з офіційного сайту www.raspberrypi.org. Raspbian містить встановлені програми для освіти, програмування та загального користування: аналоги офісних програм Microsoft, поштові клієнти, середовища програмування, ігри, службові

програми та ін.

Після завантаження необхідної версії образу операційної системи, необхідно розпакувати його з архіву, вставити microSD карту в приймач. За допомогою програми Win32 Disk Imager слід записати розпакований образ на карту. Після того, як образ буде змонтований на SD-карту, її можна вже вставити в сам Raspberry Pi. Паралельно доцільно підключити HDMI провід для виведення зображення на монітор, а також клавіатуру та мишу для керування курсором. Після цього живлення подається на пристрій. Перший запуск комп'ютера буде тривалішим, решта завантажень відбуватиметься значно швидше.

Операційні система на базі Linux, у тому числі і сам Raspbian, вже давно обзавелися зручним графічним Windows-подібним інтерфейсом, а також управління курсором мишки, набір з клавіатури, реакції на подвійні кліки, контекстне меню, що випадає, кнопка "ПУСК" та ін.

Перше, що потрібно зробити при початковому запуску комп'ютера налаштувати систему. Взаємодія користувача з операційною системою може бути як за допомогою графічної оболонки, так і допомоги командного рядка терміналу. Зміна мови операційної системи відбувається за допомогою введення команди "sudo raspi-config" у термінальному рядку. Команда sudo дозволяє здійснити виклик меню налаштувань із root правами. Без цього система видасть повідомлення про помилку нестачі прав. Також вікно налаштувань можна викликати через меню "Пуск". Після вибору потрібної мови та часової зони, необхідно перезавантажити пристрій, щоб налаштування набули чинності.

Перше, що потрібно зробити при початковому запуску комп'ютера налаштувати систему. Взаємодія користувача з операційною системою може бути як за допомогою графічної оболонки, так і допомоги командного рядка терміналу. Зміна мови операційної системи відбувається за допомогою введення команди "sudo raspi-config" у термінальному рядку. Команда sudo дозволяє здійснити виклик меню налаштувань із root правами. Без цього система видасть повідомлення про помилку нестачі прав. Також вікно налаштувань можна

викликати через меню “Пуск”. Після вибору потрібної мови та часової зони, необхідно перезавантажити пристрій, щоб налаштування набули чинності.

Таблиця 2.3 – Розпинування GPIO Raspberry Pi

№ контакту процесора Broadcom	Призначення	№ піна на роз'ємі		Призначення	№ контакту процесора Broadcom
	Харчування 3,3 В	1	2	Харчування 5 В	
2	I ² C (SDA1)	3	4	Харчування 5 В	
3	I ² C (SCL1)	5	6	GND	
4	GPCLK0	7	8	TXD0	14
	GND	9	10	RXD0	15
17	SPI1 (CE1 – Chip Enable) GPIO (GEN0)	11	12	SPI1 (CE0) PWM0	18
27	GPIO (GEN2)	13	14	GND	
22	GPIO (GEN3)	15	16		23
	Харчування 3,3 В	17	18		24
10	SPI0 (MOSI - Master Out Slave In)	19	20	GND	
9	SPI0 (MISO – Master In Slave Out)	21	22		25
11	SPI0 (SCLK)	23	24	SPI0 (CE0)	8
	GND	25	26	SPI0 (CE1)	7
	ID (SD)	27	28	ID (SC)	
5	GPCLK1	29	30	GND	
6	GPCLK2	31	32	PWM0	12
13	PWM1	33	34	GND	
19	SPI1 (MISO) PWM1	35	36	SPI1 (CE2)	16
26		37	38	SPI1 (MOSI)	20
	GND	39	40	SPI1 (SCLK)	21

Алгоритм налаштування Wi-Fi нічим не відрізняється від налаштування його на простому комп'ютері з Windows. Натиснувши на іконку Wi-Fi, потрібно

вибрати мережу та ввести пароль від неї. Після підключення можна увійти до Інтернету через браузер [18].

Увімкнувши інтерфейс SSH, можна отримати віддалений доступ до Raspberry Pi з іншого комп'ютера. Для цього в `raspi-config` необхідно налаштувати дозвіл на використання цієї функції (за умовчанням вона вимкнена). Також вкрай рекомендується перед дозволом використання поставити надійний пароль на операційну систему.

Для спілкування з модулем RYLR896 використовується бібліотека для роботи з послідовним портом – `PySerial`.

2.3 Інтерфейс з користувачем

Для використання Raspberry Pi як сервер використовується програма, написана мовою програмування Python, яка через бібліотеку `pyserial` буде спілкуватися з модулем LoRa і приймати повідомлення. Python – високорівнева мова програмування. Він підтримується майже на будь-яких відомих платформах: Windows, Unix - подібних системах (включаючи Linux), macOS і т.д.

Мова програмування Python універсальна, тому часто використовується у вирішенні багатьох завдань. Найчастіше він використовується для створення простих скриптів, веб-розробці, а також у напрямку Data Science.

Синтаксис мови дуже простий та зручний у використанні. Стандартна бібліотека має вбудовані типи даних, такі як булеві, рядки, Unicode-рядки, цілі числа, числа з плаваючою точкою, комплексні числа і т.д. Також у цій мові можуть бути використані рядки, кортежі, словники, множини та ін. Усі значення виступають як об'єкт, включаючи функції, методи, модулі, класи.

Синтаксична особливість мови програмування Python полягає в тому, що при написанні коду немає використання лапок для позначення початку та кінця

операції як у Паскалі, чи фігурних дужок та крапки з комою для відділення блоку функції як у C/C++. Натомість використовуються відступи (пробіли та табуляція). Таке рішення дозволяє спростити написання програмного коду, тобто мінімізувати набір рядків і символів, а також надати чіткості структури коду.

Для написання коментаря в коді використовується знак “#” – якщо коментар написаний на один рядок та трьома подвійними лапками на початку та наприкінці (“”) – якщо коментар розташовується на кілька рядків. Під час присвоєння змінної використовується знак “=”, а порівняння знаки “==”. Працюючи зі збільшенням чи зменшенням значення змінної використовуються оператори “+=” і “-=” відповідно. У Python використовуються оператори циклу (while), порівняння (for) та умови (if). Оголошення функцій відбувається за допомогою слова “def”. Для підключення додаткових бібліотек використовується команда “import”. При використанні конкретної функції з наявної бібліотеки можна використовувати команду “from (назва_бібліотеки) import (назва_функції)”.

Для написання та перевірки працездатності програмного коду є кілька додатків IDE, серед яких найпопулярніша PyCharm. Система аналізу коду даного середовища розробки надає автодоповнення функцій, змінних і команд та інших даних, оперативне виправлення помилок, а також підказки для належного використання шуканих операцій та багато іншого.

Як додаток, який приймає повідомлення на телефон власника автотранспортного засобу, було обрано мобільний додаток для спілкування – Telegram. Ця програма може бути встановлена на всі підтримувані телефони Android та IOS. Програма зручна тим, що в ній можуть спілкуватися не тільки різні користувачі програми, але й створювати спеціальні боти для надання інформації засобами спеціального API. Такий бот буде використаний як оповіщувач власника.

На початку необхідно створити бота у спеціальному редакторі, задавши йому ім'я. Автоматично роботі присвоюється унікальний токен. Він необхідний

подальшої роботи з telegram-ботом. Усі правила взаємодії встановлені у його API – наборі команд. Щоб отримати інформацію від конкретного робота необхідно формувати і надсилати POST запити. Порядок роботи відбувається в такий спосіб: програмний код, написаний мовою Python, відправляє запит на сервер telegram-a за допомогою бібліотеки requests через метод post. Сервер, отримавши дані, обробляє їх та відправляє відповідь назад клієнту.

При використанні та роботі з API у Python використовується бібліотека Requests. Вона дозволяє у зручному форматі працювати із запитами та отримувати відповіді.

3 РЕАЛІЗАЦІЯ ПРОЕКТУ

3.1 Побудова моделі системи захисту інформації

Насамперед було змонтовано роботу передавального модуля при подачі сигналу на пристрій Arduino. Як імітація злому виступає звичайна кнопка. Щоб зменшити обсяг займаного місця було прийнято рішення припаяти один із модулів та перехідник для регулювання напруги до монтажної плати. На звороті були припаяні ніжки для підключення до Arduino Uno. В результаті вийшла додаткова плата - Shield (рисунок 3.1).

Побудова моделі системи захисту інформації є важливим кроком для забезпечення безпеки організації та її інформаційних ресурсів. Модель системи захисту інформації являє собою концептуальну схему, яка описує різні компоненти, процеси та заходи безпеки, необхідні для забезпечення конфіденційності, цілісності та доступності інформації.

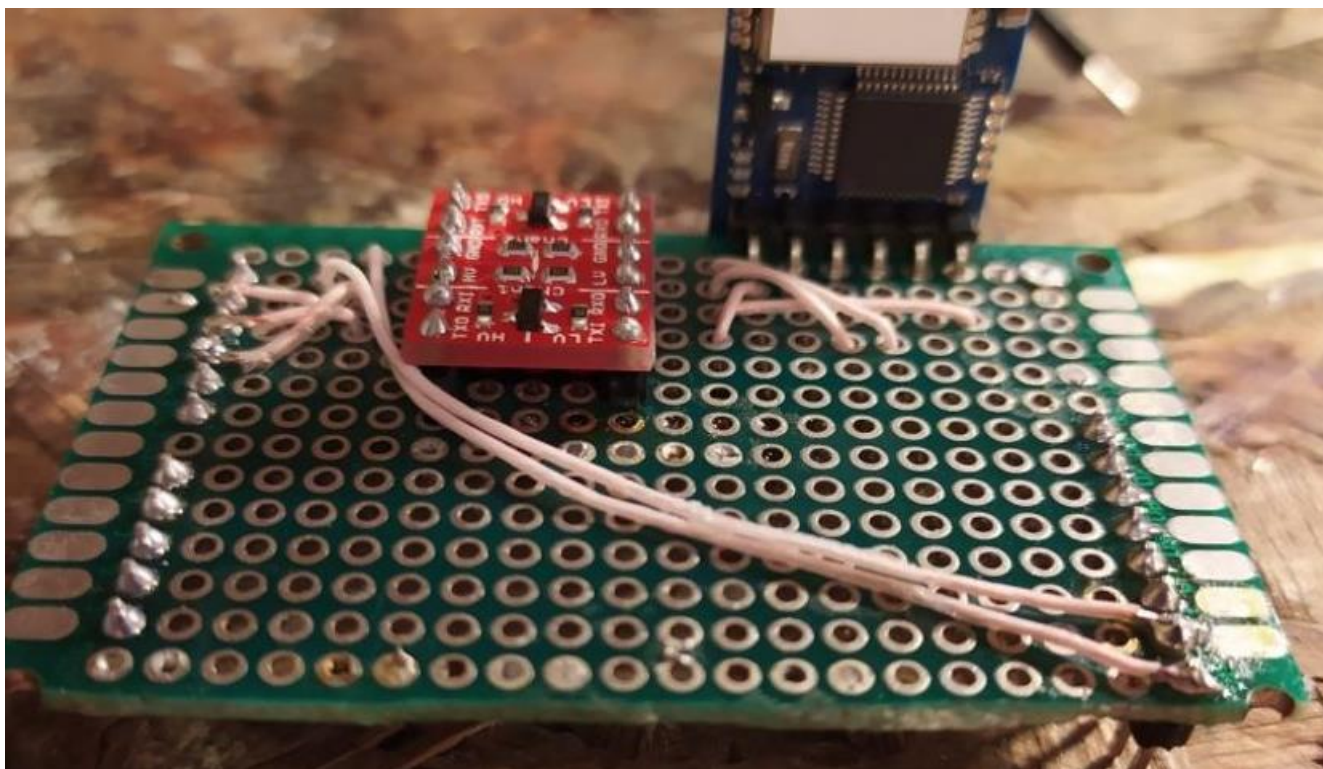


Рисунок 3.1 – Shield для Arduino Uno

Повне складання сигналізації наведено рисунку 3.2.

Shield для Arduino Uno - це додаткова плата, яка призначена для розширення функціональності Arduino Uno. Він розроблений спеціально для використання з Arduino Uno і дає змогу додати нові можливості та функції до вашого проєкту.

Shield являє собою плату, яка монтується безпосередньо на Arduino Uno, використовуючи роз'єми та контакти плати. Він розширює функціональність Arduino Uno, надаючи додаткові входи, виходи, комунікаційні інтерфейси та інші можливості.

Залежно від конкретної моделі та виробника, існує безліч різних типів і варіацій Shield для Arduino Uno, кожен з яких призначений для певних завдань і сфер застосування. Ось деякі приклади популярних типів Shield.

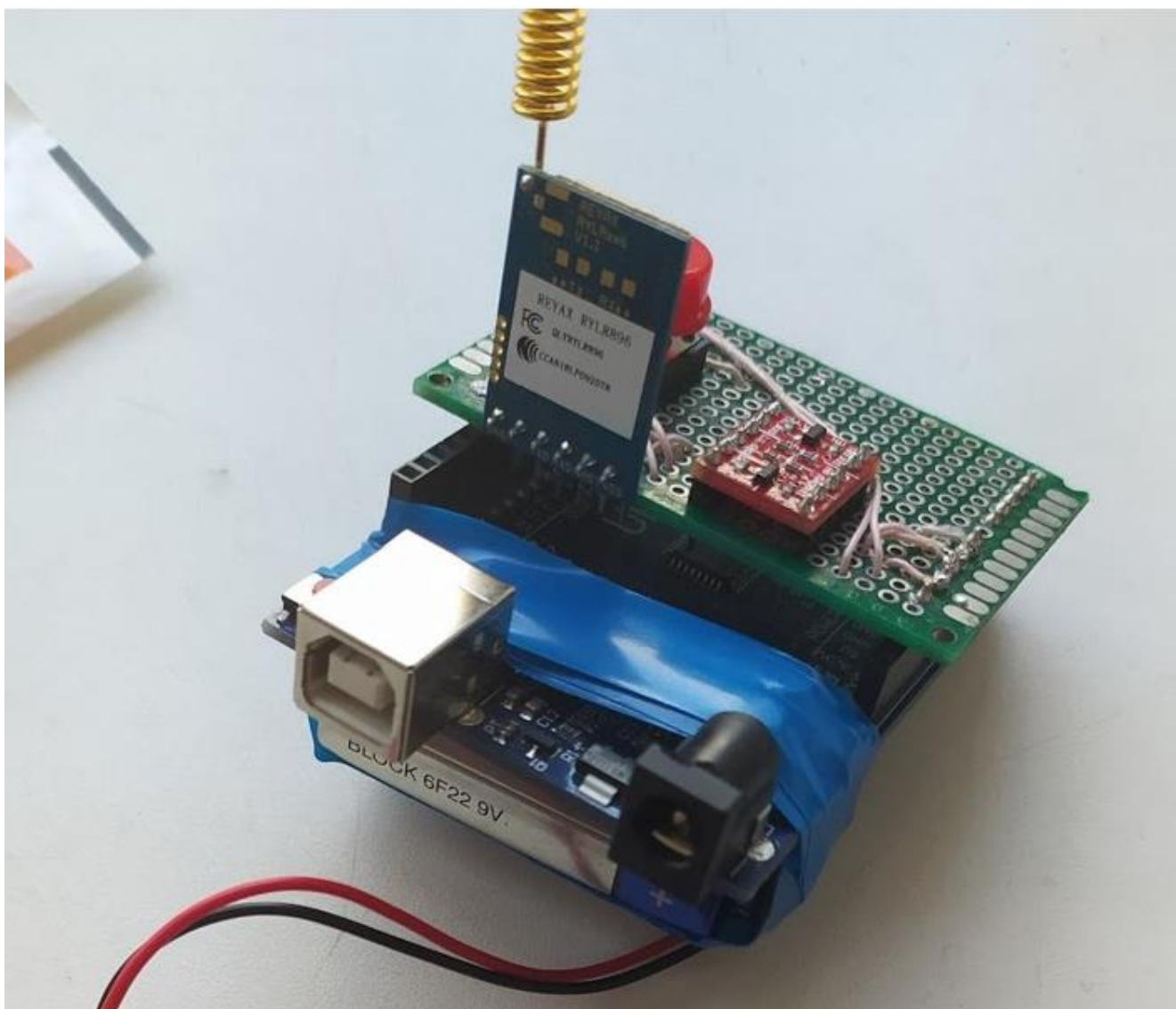


Рисунок 3.2 - Складання сигналізації, що встановлюється в автотранспорт

Аналогічним чином збирається додаткова плата з приймаючого модуля, яка встановлюватиметься на Raspberry Pi (рисунки 3.3, 3.4).

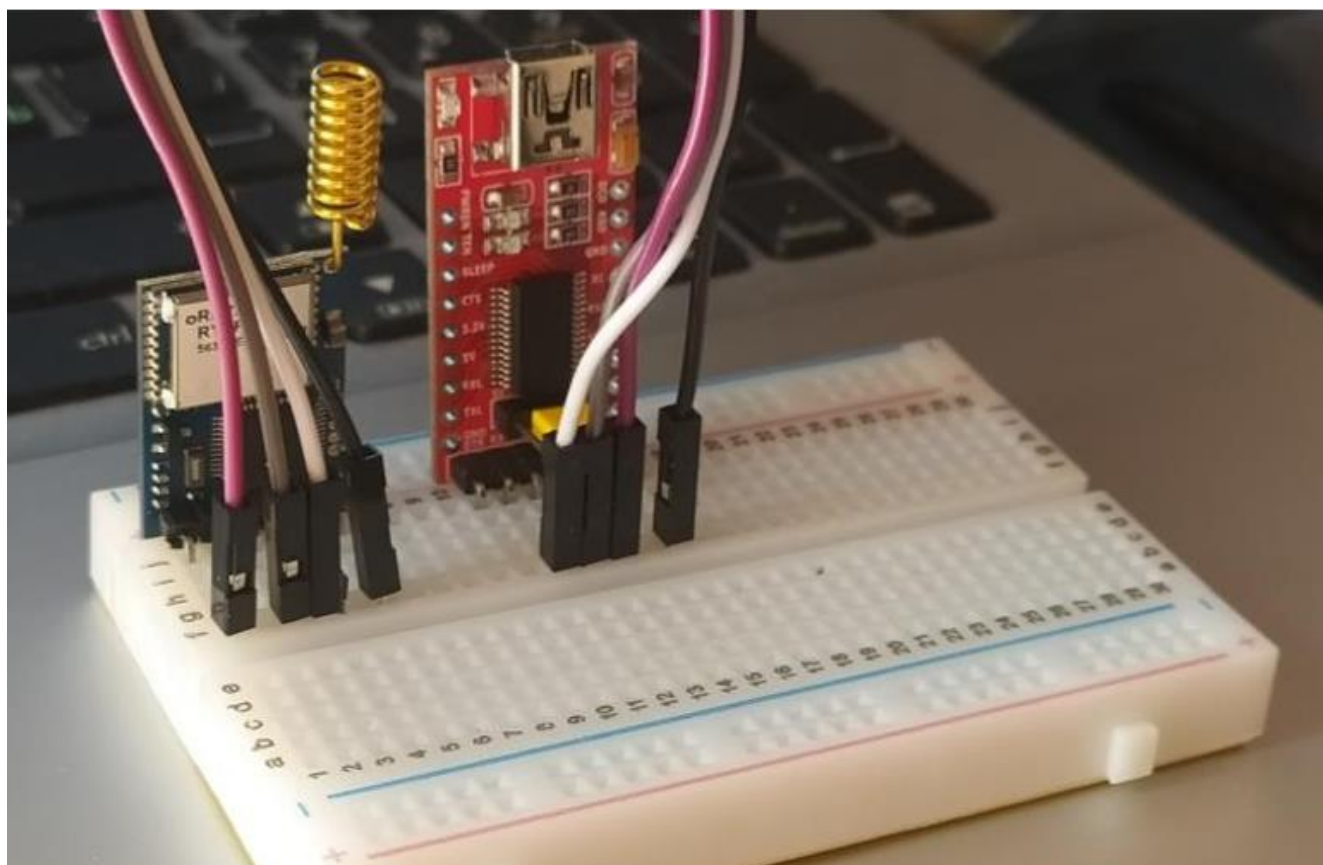


Рисунок 3.3 – Додаткова плата для Raspberry Pi



Рисунок 3.4 – Складання сигналізації, що виступає як сервер

Напишемо програмний код для спілкування модулів LoRa. Принцип їх роботи заснований наУ командах. Для початку налаштуємо їх параметри, адреси та частоти щоб вони спілкувалися між собою. Відкривши “Монітор порту” в Arduino IDE, можна перевірити роботу цих модулів. Відправник та приймаючий при правильному встановленні параметра відповідає “+OK”.

3.2 Програмний код для модулів

Програмний код для модулів написаний мовою C++ у програмі Arduino IDE. Для перевірки коректної роботи коду будемо використовувати світлодіод, встановлений на Arduino Uno. Напишемо код для тестування програми та подивимося на миготіння світлодіода (рисунок 3.5). Оскільки UART порт використовується для спілкування модуля і мікроконтролера, ми не можемо використовувати його налагодження. Тому для перевірки правильності роботи програми на мікроконтролері використовуємо миготіння світлодіодом як сигнал, що інформує нас про досягнення тієї чи іншої ділянки програмного коду.

```
void blink(uint8_t cnt) { // Моргание светодиодам для отладки программы
  for(uint8_t i = 0; i < cnt; i++){
    digitalWrite(LED_BUILTIN, HIGH);
    delay(250);
    digitalWrite(LED_BUILTIN, LOW);
    delay(50);
  }
  delay(200);
}
```

Рисунок 3.5 – Код для миготіння світлодіодом

Після перевірки пишемо сам код програми. По-перше, відкриваємо канал зв'язку по послідовному порту. Потім надсилаємо налаштування AT команд. На завершення світлодіод повинен блимати двічі (рисунок 3.6).

```
void setup() {  
  pinMode(LED_BUILTIN, OUTPUT);  
  Serial.begin(115200); // Установка скорости общения  
                        //по последовательному порту  
  
  Serial.println("AT+ADDRESS=1");  
  receive();  
  Serial.println("AT+NETWORKID=1");  
  receive();  
  Serial.println("AT+BAND=868500000");  
  receive();  
  Serial.println("AT+PARAMETER=10,7,1,7");  
  receive();  
  Serial.println("AT+SEND=2,5,HELLO");  
  receive();  
  blink(2); // Конфигурация успешна  
}
```

Рисунок 3.6 – Налаштування параметрів передачі

Як говорилося раніше, при правильній роботі модулів відповідь повинна приходити у вигляді + OK. Тому далі в коді вказуємо чи коректна відповідь щоразу, коли її отримуємо (рисунок 3.7).

```

void receive() {
    String answer = Serial.readString();
    if(get_result(answer) != RES_OK) {
        deinit();
    }
}

```

Рисунок 3.7 – Перевірка правильності відповіді

Код для приймаючого модуля буде мати функцію надання потрібного статусу для відповіді модулю-відправнику (рисунок 3.8).

```

uint8_t get_result(String str){ // Получить статус ответа
    if (str == "+OK\r\n"){
        return RES_OK;
    }
    else if (str == ""){
        return RES_NONE;
    }
    else{
        return RES_ERROR;
    }
}

```

Рисунок 3.8 – Надання статусу відповіді приймаючим модулем

```

void deinit(){ // Остановка программы
    blink(5);
    while(true){
        // end of program
    }
}

```

Рисунок 3.9 – Вимкнення мікроконтролера

Повний код програми для Arduino IDE можна побачити в додатку А.

Програмний код для сервера Raspberry Pi буде написаний мовою Python.

Для початку підключаємо бібліотеки для правильної роботи програми (рисунки 3.10).

```
import serial # бібліотека для роботи з послідовний порт
import time # бібліотека для роботи з часом
from catagram import TelegramBot # бібліотека для роботи з telegram API
```

Рисунок 3.10 – Підключення бібліотек

Далі створюємо об'єкти порту та telegram-бота (рисунки 3.11).

```
# Об'єкт послідовного порту
port = serial.Serial("/dev/ttyUSB0", baudrate=115200, timeout=3.0)
# Об'єкт телеграм бота
bot = TelegramBot()
```

Рисунок 3.11 – Створення об'єктів

Тепер напишемо програмний код, який буде відправляти дані в telegram-бот (рисунки 3.12).

```

import requests

class TelegramBot(object):

    def __init__(self):
        super(TelegramBot, self).__init__()
        self.chat_ids = [
            386830039,
            426538860
        ]

    def url_and_bot_token(self):
        return 'https://api.telegram.org/bot579887627:AAHfX87fq8VRVU777sY7d9GQXocZUbV_qvI'

    def send_img(self, img, bot_message):
        status = ''
        for chats in self.chat_ids:
            status += self.send_img_to(img, bot_message, chats)
            img.seek(0)
        return status

    def send_text(self, bot_message):
        status = ''
        for chats in self.chat_ids:
            status = self.send_text_to(bot_message, chats)
        return status

    def send_text_to(self, bot_message, bot_chatID):
        url = self.url_and_bot_token() + '/sendMessage'
        data = {'chat_id': str(bot_chatID),
                'parse_mode': 'Markdown',
                'text': bot_message,
                'disable_notification': True
               }
        response = requests.post(url=url, data=data)
        return response.json()

    def send_img_to(self, img, bot_message, bot_chatID):
        data = {'chat_id': bot_chatID,
                # 'caption': bot_message,
                'parse_mode': 'Markdown',
                'disable_notification': True}
        files = {'photo': img}
        url = self.url_and_bot_token() + '/sendPhoto'
        response = requests.post(url=url, data=data, files=files)
        return response.json()

```

Рисунок 3.12 – Код для надсилання повідомлень у telegram

Надсилаємо команди модулю (рисунок 3.13). Також, як і для Arduino налаштовуємо модуль і чекаємо на відповідь. Якщо відповідь отримана, то відправляємо її в telegram-бот (рисунок 3.14).

Повний основний програмний код для telegram-бота наведені у додатках Б та В відповідно.

```
def push_command(cmd): # отправить команду модулю
    string = 'AT+{}\\r\\n'.format(cmd)
    port.write(bytes(string, 'ascii'))
    rcv = port.readline()
    print(rcv)
```

Рисунок 3.13 – Функції надсилення команд модулю

```
# Настраиваем модуль
push_command('ADDRESS=2')
push_command('NETWORKID=1')
push_command('BAND=868500000')
push_command('PARAMETER=10,7,1,7')

while True:
    time.sleep(1)
    rcv = port.readline() # Читаем буффер
    if(rcv): # Если сообщение есть
        print(rcv)
        rcv = rcv.decode("ascii") # Дешифруем сообщение
        if(rcv.startswith('+RCV')): # Если сообщение начинается с
            rcv = rcv[5:-2:] # +RCV, то делим сообщение
            answer = rcv.split(',') # Именно третий пункт ответа
            print(answer[2]) # содержит текст сообщения
            bot.send_text(answer[2]) # Отправляем сообщение в телеграм
```

Рисунок 3.14 – Налаштування та надсилення повідомлень у telegram

ВИСНОВКИ

У цій дипломній роботі було побудовано модель контролю безпеки автотранспортного засобу.

Для виконання цього завдання було розглянуто аналогічні системи сигналізацій, вибір технології бездротової передачі даних, підбір комплектуючих для забезпечення працездатності всіх необхідних функцій системи. Був написаний програмний код на C++ для платформи Arduino Uno, командами AT були задані параметри для роботи модулів передачі даних по LoRaWAN, а також код на мові Python для Raspberry Pi виконує функцію сервера.

Розроблена система може бути використана для будь-якого виду транспорту: особистого, громадського, вантажного тощо.

Так як радіус дії одних лише модулів, згідно з експериментальними значеннями, невеликий, яка має базові станції LoRaWAN по всьому місту Алмати. Таким чином, передача сигналу досягатиме майже всього міста.

Ще однією перспективою розвитку системи є налаштування зворотного зв'язку. Тобто, власник машини зможе не тільки дізнаватися про спрацювання сигналізації в машині, але й давати сигнал транспорту для виконання дій для захисту автомобіля (наприклад, заблокувати двигун, включити камеру, встановлену в автомобілі, і дізнатися, що відбувається поряд з машиною та багато іншого).

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. ISO/IEC 27001 – Інформаційні технології. Методи забезпечення безпеки. Системи управління інформаційною безпекою».
2. ISO/IEC 27002 – Інформаційні технології. Методи забезпечення безпеки. Практичні правила управління інформаційною безпекою»
3. ISO/MEK 15408-2002. Інформаційна технологія. Методи та засоби забезпечення безпеки. Критерії оцінки безпеки інформаційних технологій. - М.: Видавництво стандартів, 2002.
4. ISO/MEK 51583-2000 «Захист інформації. Порядок створення автоматизованих систем у захищеному виконанні»
5. Добрушський С. UEBA, або поведінкова аналітика. Базова функція всіх систем безпеки майбутнього // Інформаційна безпека, 2017. - № 4. URL: http://www.itsec.com/articles2/Inf_security/ueba--ili-povedencheskaya-analitika-bazova-funkciya-vsikh-sistem-bezpeki/ (дата звернення 07.04.23)
6. Мельник Г. Модель оцінювання рівня інформаційних ризиків у корпоративних системах // Вісник Київського національного університету ім. Т.Г. Шевченка, 2015. - № 6 (171). - С. 54-60
7. Система захисту інформації від несанкціонованого доступу "Вартовий NT". Опис застосування. URL: <http://доступ>
8. Сканер безпеки SecPointPenetrator. URL: <http://ftp.technoserv.com/off-line/it/products/secur/secpoint/secpoint.pdf> (дата звернення 07.04.23)
9. Засоби адаптивного керування безпекою. URL: <http://www.rnt.com/price/> (дата звернення 07.04.23)
10. Щорічний звіт Cisco з кібербезпеки за 2021 р.: топ-менеджери у сфері безпеки роблять ставку на автоматизацію, машинне самонавчання та штучний інтелект. URL: https://www.cisco.com/c/com_ua/about/press/press-releases/2022/03-12.html (дата звернення: 07.04.23)

11. Miller, DR Security Information and Event Management (SIEM) implementation / DR Information Technology. Information Security. Information Assurancy. URL:<http://www.isaca.org>. (дата звернення: 07.04.23)
12. Jones JA An Introduction to FAIR / JA Jones - Trustees of Norwich University, 2005. - 67 p.
13. Zadeh LA Fuzzy sets / LA Zadeh. - Information and Control, 1965. - №8. - P. 338-353.
14. Zadeh LA On optimalний контроль і linear programming / LA Zadeh, BH Whalen. - IRE Trans. Automaticcontrol, Ac-7, 1962. - P. 45-46.
15. Zimmermann H.-J. Fuzzy Sets, Decision Making and Expert Systems / H.-J. Zimmermann. - Kluwer: Dordrecht, 1987. - 335 p.
16. Відкритий стандарт бездротової мережі передачі Weightless. URL:www.weightless.org(27.04.23)
17. Подкаліцький Л. Архітектура мікроконтролерів AVR: Пер. з англ. - 2019.
18. Що таке Arduino. URL:www.arduino.cc(05.02.23)
19. Arduino UNO. Технічні характеристики. URL:www.arduino.com (05.02.23)
20. Lora AT COMMANDGUIDE.REYAX TECHNOLOGY CO., LTD, 2018 року. Raspberry Pi Documentation. URL:www.raspberrypi.org(07.04.23)

ДОДАТОК А

Лістинг А1 – Код прграми

```
#define BUTTON_PIN 3
enum RESULTS {
    RES_OK,
    RES_NONE,
    RES_ERROR
};
boolean button_flag = 0;
void setup() {
    delay(1000);
    pinMode(LED_BUILTIN, OUTPUT);
    pinMode(BUTTON_PIN, INPUT_PULLUP);
    Serial.begin(115200); // Встановлення швидкості спілкування
                        //по послідовному порту
    Serial.println("AT+ADDRESS=1");
    receive();
    Serial.println("AT+NETWORKID=1");
    receive();
    Serial.println("AT+BAND=868500000");
    receive();
    Serial.println("AT+PARAMETER=10,7,1,7");
    receive();

    blink(2); // Конфігурація успішна
}

void loop() {
    boolean button = !digitalRead(BUTTON_PIN);
    if (button == 1 && button_flag == 0)
    {
        button_flag = 1;
        Serial.println("AT+SEND=2,5,HELLO");
        receive();
    }
}
```

```

    blink(2);

    if (button == 0 && button_flag == 1)
    {
        button_flag = 0;
    }
}

void receive() {
    String          answer          =
    Serial.readString();
    if(get_result(answer)          !=
    RES_OK){ deinit();
    }
}

uint8_t  get_result(String
str){    if      (str      ==
"+OK\r\n"){          return
RES_OK;

    }
    else if (str == ""){
        return RES_NONE;
    }
    else {
        return RES_ERROR;
    }
}

void deinit(){
    blink(5);
    while  (true)
    {
        // end of program
    }
}

void  blink(uint8_t  cnt)  {
    for(uint8_t i = 0; i < cnt;
i++){

```

```
    digitalWrite(LED_BUILTIN, HIGH);  
    delay(250);  
    digitalWrite(LED_BUILTIN, LOW);  
    delay(50);  
}  
delay(200);  
}
```

Лістинг А.2 – код серверної частини

```

import serial # бібліотека для роботи з послідовним портом
import time # бібліотека для роботи з часом
from catagram import TelegramBot # бібліотека для роботи з telegram#
port = serial.Serial("/dev/ttyUSB0", baudrate=115200,
timeout=3.0)
# Об'єкт послідовного порту
port = serial.Serial("/dev/ttyUSB0", baudrate=115200,
timeout=3.0) # Об'єкт телеграм бота
bot = TelegramBot()
def push_command(cmd): # відправити команду модулю string =
'AT+{} \r\n'.format(cmd)
port.write(bytes(string, 'ascii')) rcv = port.readline()
print(rcv)
# Настроюємо модуль push_command('ADDRESS=2')
push_command('NETWORKID=1') push_command('BAND=868500000')
push_command('PARAMETER=10,7,1,7')
while True: time.sleep(1)
rcv =port.readline()# Читаємо буферif(rcv):# Якщо
повідомлення є
print(rcv)
rcv =rcv.decode("ascii")# Дешифруємо
повідомленняif(rcv.startswith('+RCV')):# Якщо повідомлення
починається з
rcv =rcv[5:-2:]# +RCV, то ділимо повідомлення answer
=rcv.split(',')# Саме третій пункт відповіді
print(answer[2])# містить Текст повідомлення
# Надсилаємо повідомлення до телеграм status =
bot.send_text(answer[2]) print(status)

```

Лістинг А.3 – код клієнтської частини

```

import requests

class TelegramBot(object):

    def __init__(self): super(TelegramBot, self). __init__
()self.chat_ids = [
    386830039,
    426538860
]

    def url_and_bot_token(self):
    return 'https://api.telegram.org/bottokenXX'

    def send_img(self, img, bot_message): status = ''
    для чатів у self.chat_ids:
    status += self.send_img_to(img, bot_message, chats)
img.seek(0)
    return status

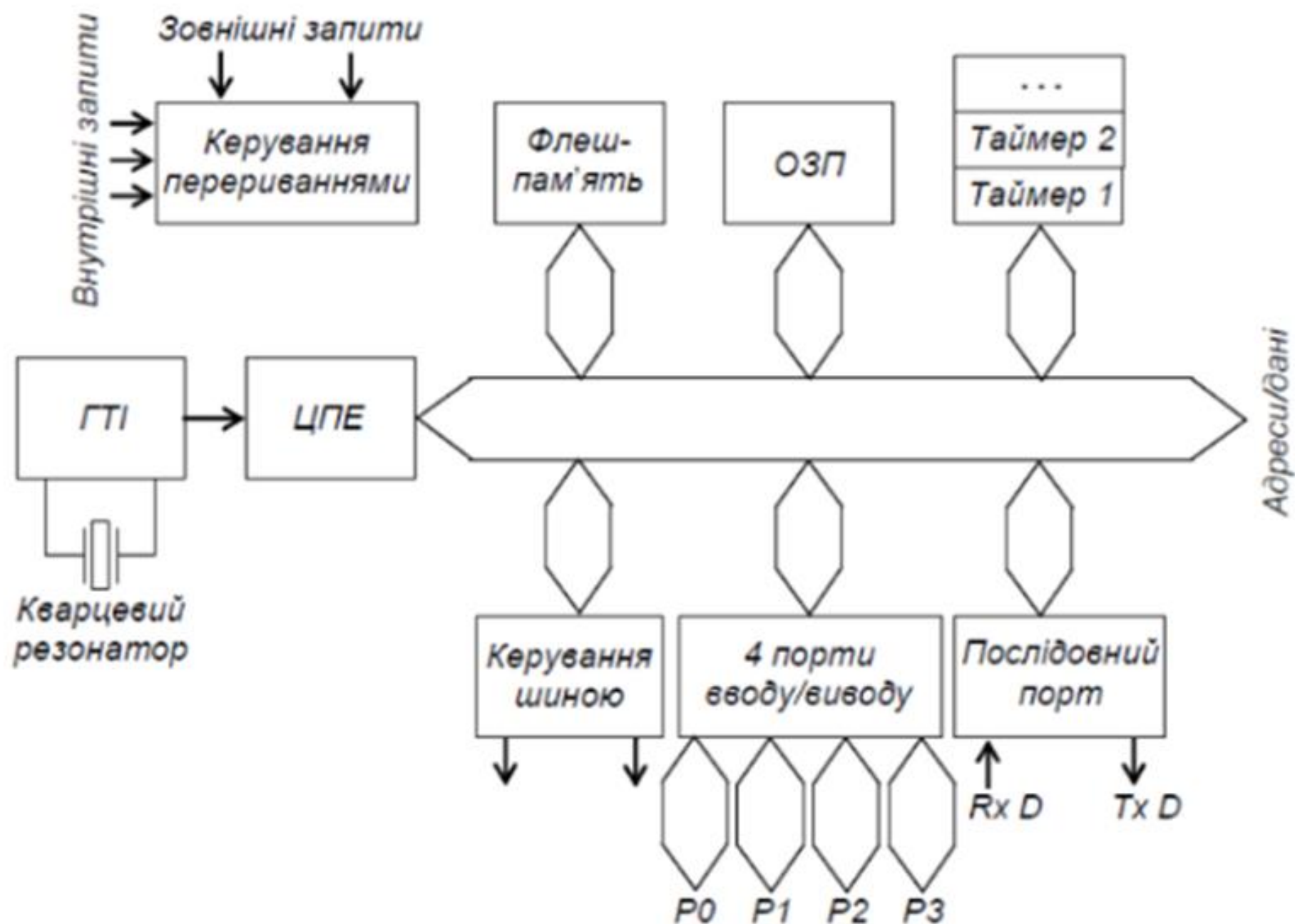
    def send_text(self, bot_message): status = ''
    для чатів у self.chat_ids:
    status = self.send_text_to(bot_message, chats) return status

    def send_text_to(self, bot_message, bot_chatID): url =
self.url_and_bot_token() + '/sendMessage' data = {'chat_id':
str(bot_chatID),
    'parse_mode': 'Markdown', 'text': bot_message,
'disable_notification': True
    }
    response = requests.post(url=url, data=data) return
response.json()

    def send_img_to(self, img, bot_message, bot_chatID): data =
{'chat_id': bot_chatID,
    # 'caption': bot_message,
    'parse_mode': 'Markdown', 'disable_notification': True}
    files = {'photo': img}
    url = self.url_and_bot_token() + '/sendPhoto'
    response = requests.post(url=url, data=data, files=files)
return response.json()

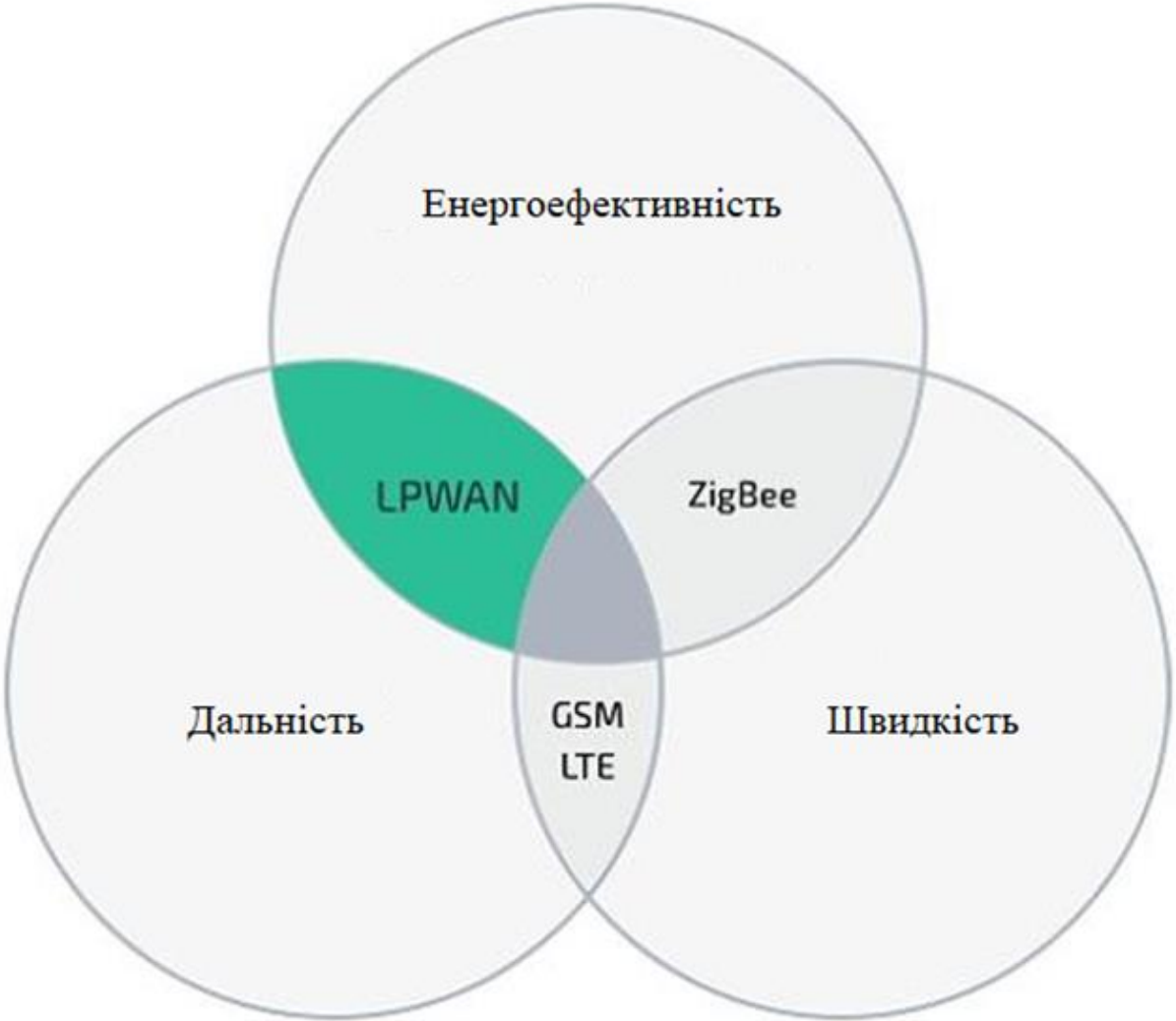
```

Структура мікроконтролера

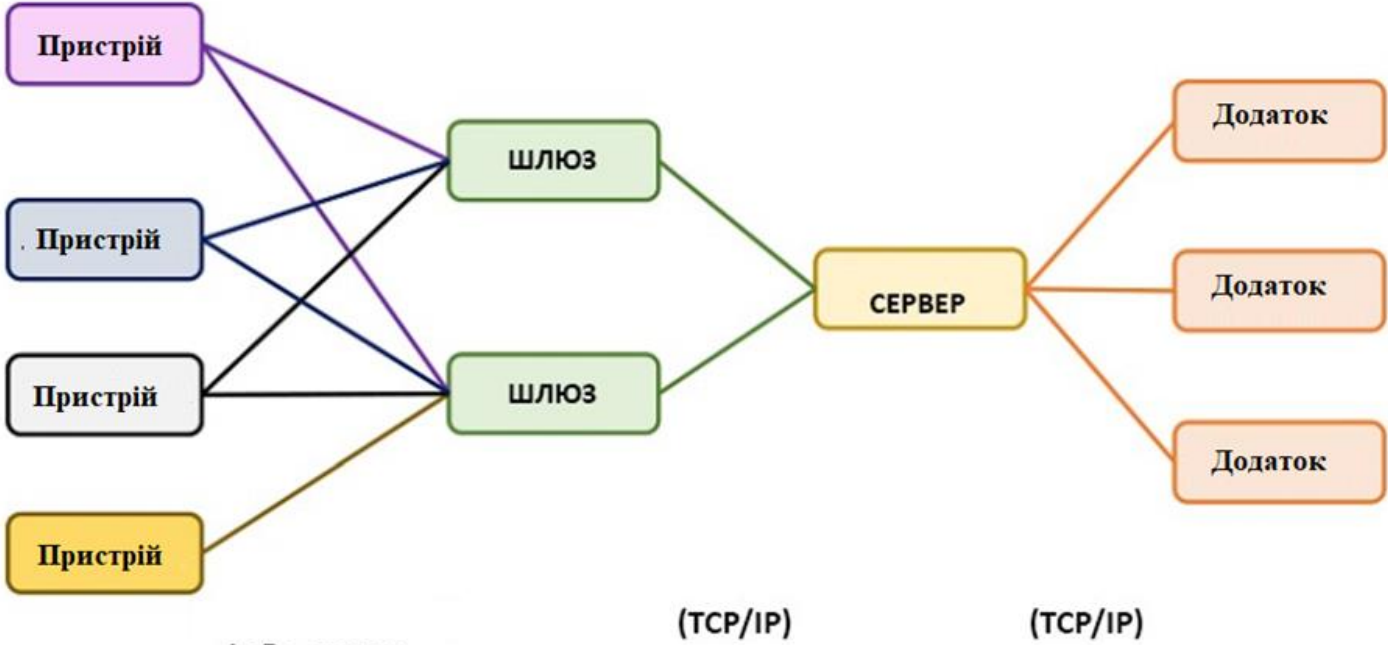


					13.02070849.51044 ПЛ1						
					Структура мікроконтролера Розробка системи оповіщення та контролю за безпекою автотранспортного засобу	Лит.		Маса		Масит.	
Зам.	Лист.	№ докум.	Підп.	Дата							
Розроб.		Шевченко Д.О.									
Перев.		Ільяхенко М.Б.									
Т.контр.											
						Лист 1		Листів 1			
Н.контр.		Щербак Н.В.				НУ «Запорізька політехніка» КНТ-510сп					
Затв.		Кудерметов Р.К.									

Вибір протоколу передачі за характеристиками бездротових технологій



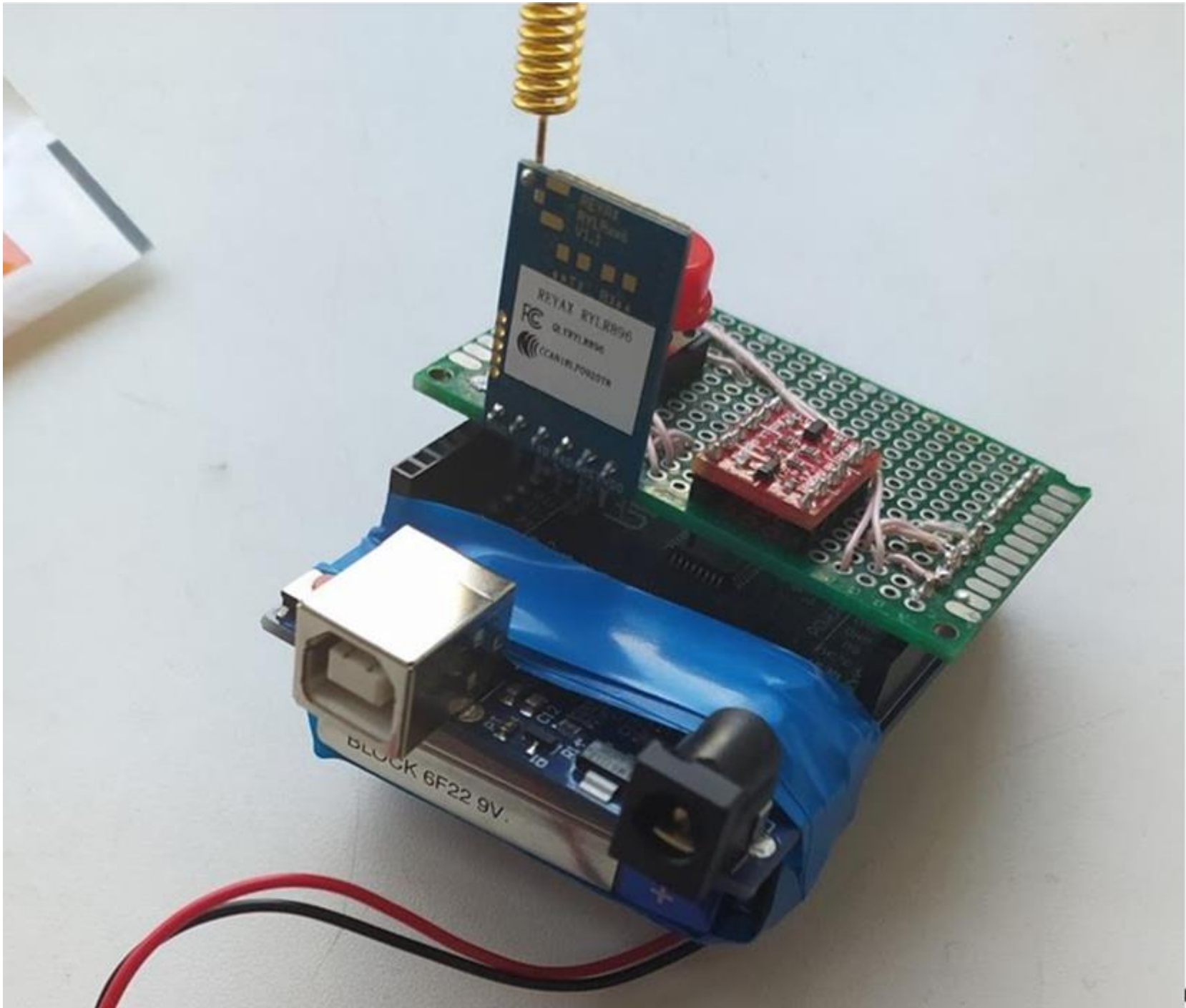
Порівняння протоколів



Принцип роботи мережі LoRa

					13.02070849.51044 ПЛ2			
					Вибір протоколу передачі за характеристиками бездротових технологій Розробка системи оповіщення та контролю за безпекою автотранспортного засобу	Лит.	Маса	Масит.
Зам.	Лист.	№ докум.	Підп.	Дата				
Розроб.		Шевченко Д.О.						
Перев.		Ілляшенко М.Б.						
Т.контр.						Лист 1	Листів 1	
Н.контр.		Щербак Н.В.				НУ «Запорізька політехніка»		
Затв.		Кудерметов Р.К.				КНТ-510сп		

Складання сигналізації, що встановлюється в автотранспорт



					13.02070849.51044 ПЛЗ						
					Складання сигналізації, що встановлюється в автотранспорт Розробка системи оповіщення та контролю за безпекою автотранспортного засобу	Лит.		Маса		Масит.	
Зам.	Лист.	№ докум.	Підп.	Дата							
Розроб.		Шевченко Д.О.									
Перев.		Ільяхенко М.Б.									
Т.контр.											
						Лист 1		Листів 1			
Н.контр.		Щербак Н.В.				НУ «Запорізька політехніка»					
Затв.		Кудерметов Р.К.				КНТ-510сп					

Складання сигналізації, що виступає як сервер



					13.02070849.51044 ПЛ4							
					Складання сигналізації, що виступає як сервер Розробка системи оповіщення та контролю за безпекою автотранспортного засобу	Лит.			Маса		Масит.	
Зам.	Лист.	№ докум.	Підп.	Дата								
Розроб.		Шевченко Д.О.										
Перев.		Ільяхенко М.Б.										
Т.контр.												
Н.контр.		Щербак Н.В.										
Затв.		Кудерметов Р.К.										