МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ПРИКАРПАТСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
ІМЕНІ ВАСИЛЯ СТЕФАНИКА
Кафедра комп’ютерних наук та інформаційних систем

Дипломна робота
на здобуття першого (бакалаврського) рівня вищої освіти на тему
«Розробка віддалено керованої інтелектуальної системи управління із застосуванням мобільної мережі для моніторингу та контролю параметрів пристроїв”

Виконав: студент IV курсу,
групи ІСТ-41

126-інформаційно системні технології
Вислюк Богдан Романович.

Керівник Стрілецький Ю.Й
Рецензент _____
(прізвище та ініціали)

Івано-Франківськ – 2025 р.

Зміст стор.

ВСТУП

Сучасний світ характеризується стрімким розвитком технологій, що все більше інтегруються в повсякденне життя, зокрема в системи управління житловими та промисловими об'єктами. Ефективне керування живленням будинку є ключовим фактором для забезпечення комфорту, енергоефективності та безпеки. В умовах зростання цін на енергоносії та потреби в раціональному споживанні ресурсів, розробка зручних та адаптованих систем дистанційного контролю та управління електроживленням стає надзвичайно актуальною. Відсутність гнучких та надійних інструментів для дистанційного управління живленням часто призводить до надмірних витрат, нераціонального використання енергії та обмеженого контролю за станом електромережі. У цьому контексті **розробка GSM-системи управління живленням** стає актуальною науково-практичною проблемою, що потребує комплексного підходу до вирішення.

Актуальність проєкту зумовлена необхідністю підвищення ефективності та безпеки управління електроживленням шляхом створення спеціалізованого програмно-апаратного комплексу, який би відповідав потребам сучасних користувачів. Критичний аналіз існуючих рішень у галузі домашньої автоматизації та систем керування живленням свідчить про їхні переваги, зокрема широкий функціонал та можливість інтеграції з іншими системами. Однак, виявляються й недоліки: обмежена адаптивність до специфічних потреб дистанційного управління через мобільну мережу, складність налаштування для пересічного користувача, а також висока вартість пропрієтарних рішень. Розробка власної GSM-системи дозволяє врахувати ці питання, забезпечивши гнучкість, доступність, надійність та безпеку.

Науково-практична значущість теми полягає в можливості створення інструменту, який сприятиме оптимізації споживання електроенергії, підвищенню безпеки експлуатації електроприладів та

забезпеченню дистанційного контролю за станом електромережі. Суспільна цінність — у підвищенні комфорту та зниженні експлуатаційних витрат для власників будинків та об'єктів. Автор роботи брав активну участь у всіх етапах дослідження – від аналізу потреб користувачів та вибору компонентів до реалізації та тестування системи, що забезпечило практичну спрямованість результатів.

Об'єктом дослідження є процес управління живленням будинку як комплекс взаємопов'язаних дій, спрямованих на контроль, комутацію та моніторинг електропостачання. **Предметом дослідження** виступає GSM-система, розроблена для автоматизації цього процесу, що є частиною об'єкта та дозволяє конкретизувати напрямок аналізу й розробки. Саме предмет дослідження визначає фокус роботи – створення програмно-апаратного рішення, яке оптимізує дистанційну взаємодію з системою живлення.

Метою роботи є розробка GSM-системи для управління живленням будинку, яка забезпечить дистанційну комутацію, моніторинг наявності електроживлення та вимірювання споживаної потужності. Для досягнення цієї мети визначено такі завдання:

- Вивчити існуючі системи керування живленням будинку та проаналізувати їхні особливості.
- Дослідити бездротові платформи та технології зв'язку, що застосовуються в домашній автоматизації.
- Обґрунтувати вибір оптимального рішення для дистанційної комутації живлення та моніторингу.
- Розробити систему команд для управління GSM-системою через дозвін та SMS.
- Реалізувати програмне забезпечення для мікроконтролера, що забезпечує взаємодію з GSM-модемом та обробку команд.
- Провести тестування системи для перевірки її працездатності та надійності.

Ці завдання структурують зміст роботи та визначають логіку викладу матеріалу в її розділах.

Стан наукової розробки: проблематика керування живленням та автоматизації житлових приміщень активно досліджується в сучасній літературі. Праці, присвячені системам домашньої автоматизації (наприклад, з використанням протоколів Bluetooth Low Energy, Insteon), підкреслюють важливість бездротових технологій. Дослідження в галузі мікроконтролерних систем та GSM-зв'язку акцентують увагу на необхідності надійних протоколів передачі даних та ефективного управління живленням. Водночас, аналіз джерел, присвячених промисловим ПЛК для систем автоматизації, свідчить про їхню надійність, але часто складність інтеграції в домашні рішення. Проте, недостатня кількість досліджень, присвячених інтеграції GSM-технологій для **автономного та безпечного управління живленням будинку з фокусом на бістабільних реле та моніторингу споживання**, обґрунтовує новизну цієї роботи.

У роботі використано комплекс методів: **аналіз літератури та аналогів** – для оцінки потреб та стану розробки систем керування живленням; **системний підхід** – для проєктування архітектури апаратної та програмної частин; **методи програмування мікроконтролерів** – для реалізації модулів; **тестування** – для перевірки працездатності та надійності системи. Методологічною основою слугували принципи об'єктно-орієнтованого програмування та інженерії мікроконтролерних систем, що забезпечили достовірність результатів.

Практичне значення одержаних результатів: розроблена GSM-система має прикладне значення, оскільки може бути використана для дистанційного управління живленням у житлових будинках, дачах, гаражах та інших об'єктах. Рекомендації щодо її інтеграції та експлуатації дозволяють адаптувати систему до потреб конкретних користувачів, а відкритий підхід до вибору компонентів сприяє подальшому вдосконаленню.

Апробація результатів: результати дослідження були представлені на групових зборах та засіданнях кафедри.

Структура роботи: робота складається зі вступу, трьох розділів, висновків, списку використаних джерел, 3 таблиць, 13 рисунків і 3 додатків, що містять текст програми. У першому розділі проаналізовано існуючі системи керування живленням та обґрунтовано вибір технологій. Другий розділ присвячено проєктуванню апаратної частини системи, а третій – розробці та тестуванню програмного забезпечення. У висновках узагальнено результати дослідження.

1 АНАЛІЗ СИСТЕМ КЕРУВАННЯ ЖИВЛЕННЯМ БУДИНКУ

Система дистанційного управління живленням квартири через GSM модем є ключовим компонентом сучасної концепції "розумного дому". Даний пристрій представляє собою апаратно-програмний комплекс, що забезпечує віддалений контроль та керування електроживленням житлових приміщень з використанням технології мобільного зв'язку GSM.

Технічна реалізація системи базується на інтеграції GSM модуля з мікроконтролерним блоком управління та силовими комутаційними елементами. GSM модем виконує функцію приймально-передавального пристрою, що забезпечує двосторонній зв'язок між користувачем та системою керування через мережу мобільного оператора. Модуль працює у стандартних діапазонах GSM 850/900/1800/1900 МГц, що гарантує сумісність з мережами більшості операторів зв'язку.

Мікроконтролерний блок обробляє вхідні команди, отримані через GSM канал, та генерує керуючі сигнали для силових реле. Система підтримує різні протоколи передачі даних, включаючи SMS, GPRS та LTE, залежно від конфігурації модема. Кодування команд здійснюється за допомогою захищених алгоритмів шифрування для запобігання несанкціонованому доступу до системи керування.

Силова частина пристрою включає набір електромагнітних або напівпровідникових реле, розрахованих на комутацію електричних кіл з напругою 220В та струмом до 16А на канал. Залежно від конфігурації, система може керувати від 2 до 16 незалежними каналами живлення, що дозволяє диференційовано контролювати різні групи електроприладів у помешканні.

Програмна частина комплексу забезпечує можливість налаштування сценаріїв автоматичної роботи обладнання відповідно до часових інтервалів, показників датчиків або зовнішніх подій. Інтерфейс керування реалізується через спеціалізований мобільний додаток з інтуїтивно зрозумілим

інтерфейсом, який дозволяє здійснювати моніторинг стану системи та керування в реальному часі.

Пристрій оснащується автономним джерелом живлення, що забезпечує безперебійну роботу системи керування протягом обмеженого періоду в разі відключення основного електроживлення. Для збільшення надійності може застосовуватися технологія дублювання каналів зв'язку через використання двох SIM-карт різних операторів.

Енергоефективність системи досягається за рахунок оптимізації роботи електроприладів відповідно до потреб користувача та зовнішніх умов. Вбудований модуль аналітики дозволяє відстежувати спожиту електроенергію та формувати статистику використання електроприладів для подальшої оптимізації.

У контексті інтеграції з іншими системами "розумного дому", пристрій підтримує відкриті API та стандартні протоколи обміну даними, що дозволяє об'єднувати його з існуючими системами безпеки, кліматичного контролю та управління освітленням. Модульна архітектура системи забезпечує можливість масштабування та модернізації без необхідності заміни базових компонентів.

1.1 Історичні аспекти

Розвиток систем домашньої автоматизації охоплює кілька ключових етапів, кожен з яких характеризується появою нових технологій та зміною підходів до організації "розумних" просторів.

Перші спроби автоматизації побуту датуються серединою XX століття, коли розробки створювалися переважно для окремих преміальних проектів. Ситуація змінилася у 1980-х роках із виникненням спеціалізованих компаній, які зосередилися на серійному виробництві обладнання та розгортанні партнерських мереж для його інсталяції. Довгий час основним стандартом у галузі залишався протокол X10, проте його обмежені можливості змусили виробників розробляти власні закриті рішення. Це

призвело до фрагментації ринку, де до початку 2010-х домінували пропрієтарні платформи.

Паралельно із комерційними рішеннями розвивався сегмент DIY (зроби сам), особливо після появи доступних мікроконтролерів, таких як Arduino та Raspberry Pi. Ентузіасти почали використовувати промислові програмовані логічні контролери (ПЛК) та відкриті технології для створення власних систем автоматизації.

Значний прорив стався у 2000-х роках із розвитком бездротових технологій (WiFi, Z-Wave, ZigBee, Bluetooth) та появою концепції Інтернету речей (IoT). Великі IT-компанії, такі як Apple, Google та Samsung, запропонували споживачам прості та інтуїтивні рішення, що сформували новий тренд — побудову "розумного будинку" на основі взаємопов'язаних пристроїв з обмеженим, але зрозумілим функціоналом.

Окремо варто відзначити стандарт KNX, розроблений європейськими виробниками електротехнічного обладнання. Ця відкрита платформа, що поєднує проводні та бездротові інтерфейси, стала популярною у преміальному сегменті та продемонструвала можливість створення масштабних, але гнучких систем автоматизації.

Сучасний ринок домашньої автоматизації представлений двома основними напрямками: масовими бездротовими екосистемами, орієнтованими на зручність та доступність, та професійними рішеннями на основі KNX та інших спеціалізованих платформ, що забезпечують високу функціональність за відповідної вартості. Ця динаміка свідчить про постійний розвиток галузі та адаптацію технологій до різних потреб користувачів.

У системах розумного будинку існують дві основні архітектурні концепції. Централізована архітектура передбачає використання головного контролера, який здійснює керування всіма підсистемами будинку через єдиний інтерфейс. Така організація забезпечує узгоджену роботу всіх компонентів системи, але створює залежність від надійності центрального

вузла. Централізований підхід особливо ефективний для складних інтегрованих рішень, де потрібна точна координація роботи різних підсистем.

Розподілена архітектура ґрунтується на принципі автономності окремих пристроїв, які взаємодіють між собою за попередньо визначеними сценаріями. Кожен смарт-пристрій у такій системі має власні обчислювальні ресурси та здатний приймати рішення локально. Цей підхід набув особливого поширення з появою бездротових технологій, таких як Z-Wave та ZigBee. Однак розподілена архітектура має суттєвий недолік – складність вирішення конфліктів між підсистемами, коли різні пристрої можуть видавати суперечливі команди.

Сучасні системи часто поєднують обидва підходи, використовуючи як централізоване керування, так і локальну автономію пристроїв. Наприклад, шина KNX дозволяє реалізувати гібридну архітектуру, де окремі пристрої можуть працювати як автономно, так і під керівництвом центрального контролера. Визначення "розумного будинку" передбачає наявність складних алгоритмів, які забезпечують узгоджену роботу всіх підсистем, а не просто сукупність окремих автоматизованих пристроїв. Такі алгоритми можуть включати автоматичне перемикання на резервне живлення або координацію роботи систем вентиляції та безпеки у надзвичайних ситуаціях.

1.2 Бездротові платформи домашньої автоматизації

Сучасний розвиток систем домашньої автоматизації характеризується поступовим переходом від провідних рішень до цифрових радіоінтерфейсів. Цей тренд обумовлений активним розвитком технологій бездротового зв'язку, таких як IEEE 802.15.4, Z-Wave та Bluetooth LE, які дозволили значно знизити енергоспоживання пристроїв та забезпечити їх тривалу автономну роботу. Використання mesh-топології та нових методів підтримки зв'язку підвищило надійність комунікації між вузлами мережі.

Провідні виробники швидко адаптувалися до нових можливостей, інтегруючи бездротові технології у свої системи. Це призвело до появи повноцінних бездротових платформ, які значно спростили розгортання систем домашньої автоматизації, особливо для прихильників концепції DIY. Відсутність необхідності прокладання додаткової кабельної інфраструктури зробила такі рішення особливо привабливими.

Історично бездротові технології в системах автоматизації використовуються з 1970-х років, коли з'явився протокол X10. Однак справжній прорив стався у 2000-х роках із розробкою спеціалізованих протоколів, таких як Z-Wave (2001) та ZigBee (2004). Ці технології були оптимізовані саме для потреб домашньої автоматизації, на відміну від більш енерговитратних рішень на основі WiFi.

Технологія Z-Wave, розроблена компанією Zensys, стала піонером ринку завдяки низькому енергоспоживанню та високій надійності. Вона забезпечує сумісність пристроїв різних виробників, хоча існують регіональні відмінності у використовуваних частотах. У 2016 році було представлено вдосконалену систему безпеки Z-Wave S2 Security на основі алгоритму AES-128.

Сімейство протоколів ZigBee, засноване на стандарті IEEE 802.15.4, відрізняється відкритістю та гнучкістю. Незважаючи на початкові проблеми з сумісністю, останні версії протоколу (ZigBee 3.0) забезпечують стабільну взаємодію пристроїв. Важливою перевагою ZigBee є підтримка роботи в глобальному діапазоні 2.4 ГГц, що усуває проблеми з регіональними частотами.

Сучасні бездротові платформи часто поєднують кілька технологій (Z-Wave, ZigBee, WiFi) у одному хабу, що дозволяє створювати комплексні системи з різними типами пристроїв. Особливу увагу приділяється питанням безпеки, оскільки зростання кількості підключених пристроїв збільшує потенційні ризики кібератак.

Розвиток бездротових технологій продовжується, з'являються нові протоколи (Thread, LoRaWAN), які розширюють можливості систем домашньої автоматизації. Важливим напрямком є інтеграція з IPv6 (ZigBee IP, Z-Wave Over IP), що відкриває нові перспективи для розвитку Інтернету речей у сфері розумного дому.

1.3 Бездротові технології звязку в системах керування

Бездротові технології, як Bluetooth Low Energy (BLE) та Insteon, стали важливою частиною систем домашньої автоматизації. BLE, який появився у 2006 році, забезпечує низькоенергетичну передачу даних на частоті 2,4 ГГц. Ця технологія була інтегрована у стандарт Bluetooth 4.0 та пізніше оновлена до Bluetooth 5.0, який підвищив швидкість передачі даних та знизив енергоспоживання.

Insteon, розроблений у 2005 році, kombinuje передачу сигналів по лініях електропроводки та радіоканалу на частоті 915 МГц. Ця технологія має високу швидкість передачі команд та може передавати їх багатьом пристроїм одночасно. Проте, в силу відмінностей в радіочастотних стандартах та стандартів електрозабезпечення, Insteon не отримав широкого поширення поза США.

З появою доступних бездротових технологій, на ринку з'явилося багато стартапів, які пропонували різноманітні смарт-пристрої. Це призвело до вимоги створення платформ, які могли б інтегрувати ці пристрої в одну систему. В результаті, у 2010-х роках відбулося розмежування підходів до побудови систем розумного будинку. З одного боку, були пропрієтарні платформи, які спеціалізувалися на домашній автоматизації та використовували власні закриті технології. З іншого боку, великі компанії, такі як Apple, Google та Samsung, розробляли власні екосистеми, де керуючі функції міграційно переносились в хмарні сервіси.

Сьогодні, на ринку домінують такі бездротові платформи, як Apple HomeKit, Google Home, Samsung SmartThings, Wink, Xiaomi Mi Home та

Insteon. Ці екосистеми включають хмарні сервіси, мобільні додатки, апаратні шлюзи та набір сумісних смарт-пристроїв. Проте, бездротові платформи все ще мають недоліки, такі як залежність від хмарних сервісів, що може знижувати надійність системи та збільшувати ризики злому. Пропрієтарні платформи, які використовують провідні технології та не делегують критичні функції у хмарні сервіси, часто виявляються більш надійними.

1.3.1 Розробка Bluetooth Low Energy та його вплив на ринок

У початку 2000-х років багато компаній розробляли власні пропрієтарні технології для економічного бездротового зв'язку. Одна з найбільших компаній, Nokia, представила у 2006 році технологію Wibree, яка була спроектована як доповнення до відкритого стандарту Bluetooth. Ця технологія забезпечувала передачу даних на частоті 2,4 ГГц з фізичною швидкістю 1 Мбіт/с. У 2010 році Wibree увійшла до складу 4-ї версії стандарту Bluetooth під назвою Bluetooth Low Energy (BLE). Першим смартфоном, який повністю підтримував стандарт Bluetooth 4.0, став iPhone 4S, що вийшов у жовтні 2011 року. Згодом багато інших пристроїв також почали підтримувати цей стандарт. У 2016 році стандарт був оновлений та опублікований у версії Bluetooth 5.0, де швидкість передачі даних зросла до 2 Мбіт/с, а енергоспоживання додатково знизилось.

Apple HomeKit, представлена у 2014 році, став власною платформою Apple для створення та управління смарт-девайсами та системами домашньої автоматизації. BLE став основним протоколом бездротового зв'язку для цієї платформи. У 2017 році Apple представила першу розумну колонку HomePod, яка була оснащена Bluetooth 5.0.

1.3.2 Insteon та його альтернативи

Insteon, розроблений компанією SmartLabs, є іншим прикладом бездротової технології для домашньої автоматизації. Ця технологія kombinuje можливості передачі сигналів по лініях електропроводки та

радіоканалу на частоті 915 МГц. Insteon успадкував від свого предтечі X10 можливість передачі сигналів управління та даних. Мережі Insteon підтримують обмежену зворотню сумісність з мережами X10. Insteon має високу швидкість передачі команд та може передавати їх багатьом пристроїм одночасно за допомогою технології Simulcast. Мережі Insteon не мають практично жодних обмежень у розмірі, а технологія Dual mesh дозволяє гнучко вибирати маршрути передачі даних.

Проте, в силу відмінностей в радіочастотних стандартах та стандартів електрозабезпечення, Insteon не отримав широкого поширення поза США. Наприклад, у Росії частоти, на яких працює Insteon, не дозволені для вільного використання, що обмежує поширення цих пристроїв. Проте, коли потрібно забезпечити зв'язок по лініях електропроводки, Insteon стає майже безальтернативним рішенням.

1.3.3 Популярні бренди та екосистеми

З появою доступних бездротових технологій, на ринку з'явилося багато стартапів, які пропонували різноманітні смарт-пристрої. Це призвело до вимоги створення платформ, які могли б інтегрувати ці пристрої в одну систему. В результаті, у 2010-х роках відбулося розмежування підходів до побудови систем розумного будинку. З одного боку, були пропрієтарні платформи, які спеціалізувалися на домашній автоматизації та використовували власні закриті технології. З іншого боку, великі компанії, такі як Apple, Google та Samsung, розробляли власні екосистеми, де керуючі функції міграційно переносились в хмарні сервіси.

Сьогодні, на ринку домінують такі бездротові платформи, як Apple HomeKit, Google Home, Samsung SmartThings, Wink, Xiaomi Mi Home та Insteon. Ці екосистеми включають хмарні сервіси, мобільні додатки, апаратні шлюзи та набір сумісних смарт-пристроїв. Проте, бездротові платформи все ще мають недоліки, такі як залежність від хмарних сервісів, що може знижувати надійність системи та збільшувати ризики злому. Пропрієтарні платформи, які використовують провідні технології та не

делегують критичні функції у хмарні сервіси, часто виявляються більш надійними.

1.4 Пропріетарні системи

Пропрієтарні платформи домашньої автоматизації традиційно domineauть на ринку, оскільки великі гравці, які розробили передові технології, забезпечили їх бурхливий розвиток. Проте, залежність від бездротових технологій призвела до того, що власники таких систем часто упускають необхідність комплексного розвитку всієї системи розумного будинку, де важливу роль відіграють також високошвидкісні шини передачі даних, які розраховані на традиційну дротову інфраструктуру. Ці шини та пов'язане з ними програмне забезпечення стали відігравати визначальну роль в проектуванні та забезпеченні необхідних показників систем домашньої автоматизації.

Перші пропрієтарні платформи та системи домашньої автоматизації були розроблені у 1980-х роках, коли інтерес до розумних будинків був на піку. Типові проекти того часу використовували відкритий протокол X10, розроблений американською компанією Pico Electronics у 1975 році, а також поширених комп'ютерних інтерфейсів, таких як RS-232 або Ethernet. Проте, до середини 1980-х років стало зрозуміло, що для передачі даних і команд управління між пристроями в системах розумного будинку потрібні нові цифрові інтерфейси та протоколи, які забезпечать високу швидкість передачі без надмірних вимог до проводки.

У період з 1985 по 1995 роки кілька компаній розробили різноманітні технології, які стали основою мейнстріму індустрії. Наприклад, американська компанія Vantage Controls, заснована у 1986 році, використовувала спеціальну шину Vantage Station Bus. Бельгійська компанія TELETASK, заснована у 1984 році, розробила шину Autobus. Австралійська компанія Clipsal, разом з партнерами, почала просувати шину C-Bus з 1994

року, а італійська BTicino вивела на ринок пристрої з шиною SCS у 2001 році.

Навіть участь у консорціумах, таких як EIB (European Installation Bus), яка пізніше стала основою стандарту KNX, не змусила провідних гравців швидко реалізовувати підтримку власних пропрієтарних технологій. Кожен розробник пропонував повний спектр інфраструктурних рішень, включаючи центральні контролери, спеціалізоване програмне забезпечення та виконавчі пристрої, що приводило до ситуації, коли споживач, обравши власницьке рішення, став залежним від одного постачальника при розширенні або модернізації системи.

На початку 2000-х років ринок домашньої автоматизації змінився, оскільки звичайні споживачі середнього достатку почали проявляти більший інтерес до таких систем. Пропрієтарні платформи спробували стати ближче до масового споживача, але все ще орієнтувалися на високий ціновий сегмент. Найбільші гравці, такі як Crestron, Vantage, Savant, BTicino, TELETASK та Clipsal, домінують на своїх локальних ринках, пропонуючи різноманітні рішення з мультимедіа, дизайном та керуванням.

Пропрієтарні платформи домашньої автоматизації традиційно домінують на ринку, оскільки великі гравці, які розробили передові технології, забезпечили їх бурхливий розвиток. Проте, залежність від бездротових технологій призвела до того, що власники таких систем часто упускають необхідність комплексного розвитку всієї системи розумного будинку, де важливу роль відіграють також високошвидкісні шини передачі даних, які розраховані на традиційну дротову інфраструктуру. Ці шини та пов'язане з ними програмне забезпечення стали відігравати визначальну роль в проектуванні та забезпеченні необхідних показників систем домашньої автоматизації.

Перші пропрієтарні платформи та системи домашньої автоматизації були розроблені у 1980-х роках, коли інтерес до розумних будинків був на піку. Типові проекти того часу використовували відкритий протокол X10,

розроблений американською компанією Pico Electronics у 1975 році, а також поширених комп'ютерних інтерфейсів, таких як RS-232 або Ethernet. Проте, до середини 1980-х років стало зрозуміло, що для передачі даних і команд управління між пристроями в системах розумного будинку потрібні нові цифрові інтерфейси та протоколи, які забезпечать високу швидкість передачі без надмірних вимог до проводки.

У період з 1985 по 1995 роки кілька компаній розробили різноманітні технології, які стали основою мейнстріму індустрії. Наприклад, американська компанія Vantage Controls, заснована у 1986 році, використовувала спеціальну шину Vantage Station Bus. Бельгійська компанія TELETASK, заснована у 1984 році, розробила шину Autobus. Австралійська компанія Clipsal, разом з партнерами, почала просувати шину C-Bus з 1994 року, а італійська BTicino вивела на ринок пристрої з шиною SCS у 2001 році.

Навіть участь у консорціумах, таких як EIB (European Installation Bus), яка пізніше стала основою стандарту KNX, не змусила провідних гравців швидко реалізовувати підтримку власних пропрієтарних технологій. Кожен розробник пропонував повний спектр інфраструктурних рішень, включаючи центральні контролери, спеціалізоване програмне забезпечення та виконавчі пристрої, що приводило до ситуації, коли споживач, обравши власницьке рішення, став залежним від одного постачальника при розширенні або модернізації системи.

На початку 2000-х років ринок домашньої автоматизації змінився, оскільки звичайні споживачі середнього достатку почали проявляти більший інтерес до таких систем. Пропрієтарні платформи спробували стати ближче до масового споживача, але все ще орієнтувалися на високий ціновий сегмент. Найбільші гравці, такі як Crestron, Vantage, Savant, BTicino, TELETASK та Clipsal, домінують на своїх локальних ринках, пропонуючи різноманітні рішення з мультимедіа, дизайном та керуванням.

Пропрієтарні платформи домашньої автоматизації традиційно домінують на ринку, оскільки великі гравці, які розробили передові технології, забезпечили їх бурхливий розвиток. Проте, залежність від бездротових технологій призвела до того, що власники таких систем часто упускають необхідність комплексного розвитку всієї системи розумного будинку, де важливу роль відіграють також високошвидкісні шини передачі даних, які розраховані на традиційну дротову інфраструктуру. Ці шини та пов'язане з ними програмне забезпечення стали відігравати визначальну роль в проектуванні та забезпеченні необхідних показників систем домашньої автоматизації.

Перші пропрієтарні платформи та системи домашньої автоматизації були розроблені у 1980-х роках, коли інтерес до розумних будинків був на піку. Типові проекти того часу використовували відкритий протокол X10, розроблений американською компанією Pico Electronics у 1975 році, а також поширених комп'ютерних інтерфейсів, таких як RS-232 або Ethernet. Проте, до середини 1980-х років стало зрозуміло, що для передачі даних і команд управління між пристроями в системах розумного будинку потрібні нові цифрові інтерфейси та протоколи, які забезпечать високу швидкість передачі без надмірних вимог до проводки.

У період з 1985 по 1995 роки кілька компаній розробили різноманітні технології, які стали основою мейнстріму індустрії. Наприклад, американська компанія Vantage Controls, заснована у 1986 році, використовувала спеціальну шину Vantage Station Bus. Бельгійська компанія TELETASK, заснована у 1984 році, розробила шину Autobus. Австралійська компанія Clipsal, разом з партнерами, почала просувати шину C-Bus з 1994 року, а італійська BTicino вивела на ринок пристрої з шиною SCS у 2001 році.

Навіть участь у консорціумах, таких як EIB (European Installation Bus), яка пізніше стала основою стандарту KNX, не змусила провідних гравців швидко реалізовувати підтримку власних пропрієтарних технологій. Кожен

розробник пропонував повний спектр інфраструктурних рішень, включаючи центральні контролери, спеціалізоване програмне забезпечення та виконавчі пристрої, що приводило до ситуації, коли споживач, обравши власницьке рішення, став залежним від одного постачальника при розширенні або модернізації системи.

На початку 2000-х років ринок домашньої автоматизації змінився, оскільки звичайні споживачі середнього достатку почали проявляти більший інтерес до таких систем. Пропрієтарні платформи спробували стати ближче до масового споживача, але все ще орієнтувалися на високий ціновий сегмент. Найбільші гравці, такі як Crestron, Vantage, Savant, BTicino, TELETASK та Clipsal, domineauть на своїх локальних ринках, пропонуючи різноманітні рішення з мультимедіа, дизайном та керуванням.

Пропрієтарні платформи домашньої автоматизації традиційно domineauть на ринку, оскільки великі гравці, які розробили передові технології, забезпечили їх бурхливий розвиток. Проте, залежність від бездротових технологій призвела до того, що власники таких систем часто упускають необхідність комплексного розвитку всієї системи розумного будинку, де важливу роль відіграють також високошвидкісні шини передачі даних, які розраховані на традиційну дротову інфраструктуру. Ці шини та пов'язане з ними програмне забезпечення стали відігравати визначальну роль в проектуванні та забезпеченні необхідних показників систем домашньої автоматизації.

Перші пропрієтарні платформи та системи домашньої автоматизації були розроблені у 1980-х роках, коли інтерес до розумних будинків був на піку. Типові проекти того часу використовували відкритий протокол X10, розроблений американською компанією Pico Electronics у 1975 році, а також поширених комп'ютерних інтерфейсів, таких як RS-232 або Ethernet. Проте, до середини 1980-х років стало зрозуміло, що для передачі даних і команд управління між пристроями в системах розумного будинку потрібні нові

цифрові інтерфейси та протоколи, які забезпечать високу швидкість передачі без надмірних вимог до проводки.

У період з 1985 по 1995 роки кілька компаній розробили різноманітні технології, які стали основою мейнстріму індустрії. Наприклад, американська компанія Vantage Controls, заснована у 1986 році, використовувала спеціальну шину Vantage Station Bus. Бельгійська компанія TELETASK, заснована у 1984 році, розробила шину Autobus. Австралійська компанія Clipsal, разом з партнерами, почала просувати шину C-Bus з 1994 року, а італійська BTicino вивела на ринок пристрої з шиною SCS у 2001 році.

Навіть участь у консорціумах, таких як EIB (European Installation Bus), яка пізніше стала основою стандарту KNX, не змусила провідних гравців швидко реалізовувати підтримку власних пропрієтарних технологій. Кожен розробник пропонував повний спектр інфраструктурних рішень, включаючи центральні контролери, спеціалізоване програмне забезпечення та виконавчі пристрої, що приводило до ситуації, коли споживач, обравши власницьке рішення, став залежним від одного постачальника при розширенні або модернізації системи.

На початку 2000-х років ринок домашньої автоматизації змінився, оскільки звичайні споживачі середнього достатку почали проявляти більший інтерес до таких систем. Пропрієтарні платформи спробували стати ближче до масового споживача, але все ще орієнтувалися на високий ціновий сегмент. Найбільші гравці, такі як Crestron, Vantage, Savant, BTicino, TELETASK та Clipsal, домінеають на своїх локальних ринках, пропонуючи різноманітні рішення з мультимедіа, дизайном та керуванням.

Пропрієтарні платформи домашньої автоматизації традиційно домінеають на ринку, оскільки великі гравці, які розробили передові технології, забезпечили їх бурхливий розвиток. Проте, залежність від бездротових технологій призвела до того, що власники таких систем часто упускають необхідність комплексного розвитку всієї системи розумного

будинку, де важливу роль відіграють також високошвидкісні шини передачі даних, які розраховані на традиційну дротову інфраструктуру. Ці шини та пов'язане з ними програмне забезпечення стали відігравати визначальну роль в проектуванні та забезпеченні необхідних показників систем домашньої автоматизації.

Перші пропрієтарні платформи та системи домашньої автоматизації були розроблені у 1980-х роках, коли інтерес до розумних будинків був на піку. Типові проекти того часу використовували відкритий протокол X10, розроблений американською компанією Pico Electronics у 1975 році, а також поширених комп'ютерних інтерфейсів, таких як RS-232 або Ethernet. Проте, до середини 1980-х років стало зрозуміло, що для передачі даних і команд управління між пристроями в системах розумного будинку потрібні нові цифрові інтерфейси та протоколи, які забезпечать високу швидкість передачі без надмірних вимог до проводки.

У період з 1985 по 1995 роки кілька компаній розробили різноманітні технології, які стали основою мейнстріму індустрії. Наприклад, американська компанія Vantage Controls, заснована у 1986 році, використовувала спеціальну шину Vantage Station Bus. Бельгійська компанія TELETASK, заснована у 1984 році, розробила шину Autobus. Австралійська компанія Clipsal, разом з партнерами, почала просувати шину C-Bus з 1994 року, а італійська BTicino вивела на ринок пристрої з шиною SCS у 2001 році.

Навіть участь у консорціумах, таких як EIB (European Installation Bus), яка пізніше стала основою стандарту KNX, не змусила провідних гравців швидко реалізовувати підтримку власних пропрієтарних технологій. Кожен розробник пропонував повний спектр інфраструктурних рішень, включаючи центральні контролери, спеціалізоване програмне забезпечення та виконавчі пристрої, що приводило до ситуації, коли споживач, обравши власницьке рішення, став залежним від одного постачальника при розширенні або модернізації системи.

Пропрієтарні системи домашньої автоматизації та платформи, такі як BTicino My HOME, Clipsal, Control4, Crestron Home, Savant та Vantage InFusion, є відомими на ринку розумних будинків. BTicino My HOME, розроблена у 2001 році, використовує шину SCS bus для взаємодії пристроїв. Clipsal, австралійський бренд, що входить до концерну Schneider Electric, пропонує різноманітне обладнання для розумних будинків, включаючи лінійку C-Bus та серію NERO. Control4, виведена на ринок у 2004 році, пропонує операційну систему для розумного будинку з вбудованими інтерфейсами Ethernet, RS232, WiFi та ZigBee. Crestron Home, заснована у 1971 році, пропонує повний спектр обладнання для автоматизації будь-якого розміру. Savant, створена у 2005 році, є лідером в преміальному сегменті американського ринку. Vantage InFusion використовує централізовану схему з двопроводною шиною WireLink та можливістю радіоінтерфейсу RadioLink.

Крім того, існують платформи на основі шини KNX, такі як iRidium, Theben LUXORliving, GIRA KNX system та Schneider Electric U.MOTION KNX Server. Ці платформи використовують відкритий стандарт KNX для інтеграції різних пристроїв у систему розумного будинку.

У останні роки набуло популярності рух DIY (Do It Yourself) у сфері домашньої автоматизації. Цей тренд сприяв розвитку відкритих платформ, таких як Arduino та Raspberry Pi, які дозволяють створювати недорогі та функціональні системи розумного будинку. Відкриті програмні платформи, такі як OpenHAB, MajorDoMo, ioBroker, IoT Manager, Domoticz, Home Assistant, HomeGenie, FHEM, Misterhouse та Homebridge, стають все популярнішими серед ентузіастів DIY. Ці платформи пропонують різноманітні можливості для керування розумними пристроями, включаючи голосове керування, мобільні додатки, інтеграцію з Apple HomeKit та інші функції.

Важливо зазначити, що всі вищевказані системи та платформи є лише програмними оболонками, які призначені для управління кінцевим

обладнанням, з якого складаються підсистеми розумного будинку. Концепція DIY передбачає використання різноманітних пристроїв, включаючи розумні гаджети, noname-обладнання та самостійно зібрані підсистеми. Ці програмні оболонки можуть бути встановлені на різних комп'ютерах, включаючи мікрокомп'ютери типу Raspberry Pi, які використовуються як центральний процесор розумного будинку.

BTicino My HOME, розроблена у 2001 році італійською компанією BTicino, є системою домашньої автоматизації, яка використовує шину SCS bus для взаємодії пристроїв. Ця система пропонує широкий спектр обладнання для різних завдань домашньої автоматизації, включаючи освітлення, опалення, кондиціонування та інші підсистеми. BTicino My HOME є відомою своїм італійським дизайном та високою надійністю.

Clipsal, австралійський підрозділ концерну Schneider Electric, займається домашньою автоматизацією з 1994 року. Clipsal пропонує кілька лінійок різноманітного обладнання та рішень для розумних будинків, включаючи лінійку обладнання на основі фірмової шини C-Bus, програми та рішення для управління Wiser, а також хаби та виконавчі пристрої серії NERO. Clipsal відомий своєю надійністю та високою якістю обладнання.

Control4, виведена на ринок у 2004 році, є відносно новою, але вдалий платформою для домашньої автоматизації. Control4 позиціонується як операційна система для розумного будинку, а її пристрої, в основному, є центральними процесорами, які працюють під управлінням цієї операційної системи. Для взаємодії з обладнанням розумного будинку контролери Control4 мають вбудовані інтерфейси Ethernet, RS232, WiFi та ZigBee. Control4 також може використовувати зовнішні шлюзи з відповідними драйверами для підтримки інших інтерфейсів, таких як KNX, Modbus, 1-Wire, Z-Wave та інших. У 2012 році Control4 опублікувала свій протокол SDDP (Simple Device Discovery Protocol), який дозволяє швидко виявляти та розпізнавати нові пристрої в домашній мережі.

Crestron Home, розроблена компанією Crestron Electronics, є одним з найстаріших та найбільш відомих рішень на ринку домашньої автоматизації. Компанія була заснована у 1971 році та пропонує повний спектр обладнання та рішень для автоматизації будь-якого розміру. Crestron Home відомий своїми передовими технологіями в сфері мультимедіа та високої роздільної здатності для багатозонних систем. Crestron пропонує широкий спектр обладнання для розумних будинків, включаючи контролери, панелі управління та виконавчі пристрої

1.5 Використання промислових ПЛК для систем автоматизації

Промислові ПЛК з відкритим вихідним кодом на основі Arduino або Raspberry Pi набувають популярності в системах розумного будинку. Це пов'язано не лише з низькою вартістю та доступністю відкритого коду, а й з тим, що такі ПЛК призначені для автономного використання в неблагоприятних умовах без постійного обслуговування. Багато поширених промислових ПЛК відповідають вимогам систем управління розумним будинком. Вибір устаткування для DIY-систем сильно залежить від ціни, тому Arduino є популярним вибором. Однак останнім часом з'явилося багато інших недорогих ПЛК, які легко освоїти та застосувати, а також просто інтегруються з програмними платформами для домашньої автоматизації.

Apple HomeKit - це програмний фреймворк, інтегрований в операційні системи iOS, macOS, watchOS, tvOS та audioOS, який дозволяє керувати сумісними пристроями за допомогою пристроїв Apple та голосового помічника Siri. У системі розумного будинку роль центрального процесора виконує один з пристроїв Apple, такий як iPad, HomePod, Apple TV або Macbook, який виступає головним хабом (HomeKit Hub). Взаємодія з пристроями здійснюється за допомогою пропрієтарного протоколу HomeKit Accessory Protocol (HAP), який використовує IP або Bluetooth LE як транспорт. Управління можливе з будь-якого пристрою Apple через додаток Apple Home App.

Apple HomeKit була представлена разом з iOS 8 у 2014 році. Більшість пристроїв Apple і партнерів спочатку орієнтувалися на бездротові інтерфейси, тому HomeKit став бездротовою платформою, яка працює поверх WiFi або Bluetooth LE. З появою Apple TV 4-го покоління у 2015 році, користувачам та розробникам став доступним безпосередній доступ до традиційних Ethernet-комунікацій. Однак, багато виробників продовжують орієнтуватися на прості засоби зв'язку на основі Bluetooth LE, що передбачає безпосередню взаємодію по радіоканалу між HomeKit Hub і смарт-девайсами. Це накладає обмеження на умови і місце розміщення хаба та можливості системи розумного будинку в цілому.

У 2018 році Apple випустила свою розумну колонку HomePod, яка функціонує на базі Siri і може виступати хабом HomeKit Hub. З цього моменту HomePod став основним елементом розумного будинку від Apple, а багато хто повністю ототожнює розумну колонку з голосовим інтерфейсом та розумний будинок як такий. У кінці 2018 року macOS Mojave було оновлено, і Macbook також отримав можливість виступати хабом для HomeKit.

Успіх Apple HomeKit залежить від того, скільки виробників різноманітних смарт-пристроїв і систем розумного будинку будуть вбудовувати підтримку цієї платформи в свої продукти. Компанія Apple має програму ліцензування MFi Program, де учасники отримують права на використання протоколу HAP і можуть вбудовувати його в своє обладнання. Спрощена версія цього протоколу з'явилася в 2017 році і може використовуватися для кастомізованої розробки без права вбудовування в тиражні рішення.

Apple HomeKit має спрощений підхід до алгоритмів автоматичного управління, який базується на сценаріях (scenes), програмування яких включено у функціонал додатка Apple Home App. Хоча сценарії можуть бути складними, але для реалізації адаптивних алгоритмів, наприклад, для системи опалення, сценарне управління не є найбільш зручним

інструментом. Проте, архітектура Apple HomeKit має інший розвинений інструментарій та дозволяє написання власних iOS-додатків для реалізації складних проектів.

Напівпромислові платформи на основі ПЛК відрізняються можливістю гнучкого програмування за допомогою стандартних інструментів і мов програмування, які поширені в сфері промислових ПЛК. Такі платформи розвивають невеликі компанії та ентузіасти, які пишуть програмну начинку, але не можуть створюти повноцінні екосистеми. Однак, деякі такі рішення є популярними завдяки своїй гнучкості, коли розробник може дописати відсутній драйвер або просте логічне правило. Наприклад, деякі компанії збирають власні контролери на основі доступних вбудованих плат мікроконтролерів та ПЛК. Такі контролери розумного будинку дуже близькі до промислових ПЛК за своїми характеристика

2 РОЗРОБКА АПАРАТНОЇ ЧАСТИНИ СИСТЕМИ

2.1 Розробка вимог до GSM-системи управління живленням

Система має забезпечити надійне управління реле живлення через GSM-зв’язок. Для цього вона виконує такі функції:

Забезпечення стабільного підключення модему до мережі. Це критично для гарантування керованості пристрою через SMS і дзвінки.

Формування списку авторизованих користувачів. Це дає змогу відрізнити довірених абонентів від сторонніх і дозволяти виконання команд лише перевіреним номерам.

Обробка SMS-повідомлень і вхідних дзвінків. За їхнім змістом система виконує відповідні дії: змінює стан реле, додає номери, надсилає USSD-запити.

Обробка аварійних подій. Наприклад, перегрів — реле вимикається, і надсилається SMS.

2.2 Загальна архітектура системи

Система віддаленого керування електроживленням квартири побудована на основі мікроконтролера, що взаємодіє з GSM модемом через інтерфейс UART. Центральним елементом системи є бістабільне реле, яке забезпечує комутацію мережі живлення та зберігає свій стан після відключення керуючої напруги. Завдяки використанню бістабільного реле, система стабільно функціонує навіть при тимчасових збоях електроживлення, зберігаючи останній встановлений стан.

2.2.1 Вибір оптимального рішення для дистанційної комутації живлення

При проектуванні систем дистанційного управління електроживленням ключовим елементом є вибір належного комутаційного пристрою, що дозволяє малопотужним керуючим сигналам ефективно комутувати потужні навантаження. Такий підхід забезпечує енергетичну

ефективність, безпеку експлуатації та надійність системи управління в цілому.

Електромеханічні реле залишаються найпоширенішим рішенням завдяки оптимальному співвідношенню вартості та функціональності. Принцип їх дії ґрунтується на електромагнітній індукції: струм, що проходить через котушку керування (зазвичай 5-24В, 20-100мА), створює магнітне поле, яке приводить у рух якір з контактною групою. Це дозволяє комутувати навантаження з параметрами до 380В і струмами до десятків амперів при потужності керуючого сигналу менше 1Вт. Повна гальванічна розв'язка між керуючим колом і силовою частиною забезпечує високий рівень безпеки системи.

Магнітні пускачі являють собою більш масштабний різновид електромеханічних реле (рис.2.1), спеціально адаптований для комутації потужних промислових навантажень. Вони мають посилену контактну систему, розраховану на струми до сотень амперів, часто оснащуються додатковими системами дугогасіння та тепловими реле захисту. Використання малопотужного сигналу для керування магнітним пускачем через проміжне реле дозволяє істотно збільшити відстань між пунктом контролю та об'єктом керування, використовуючи для передачі керуючих сигналів низьковольтні, малопоперечні кабелі.

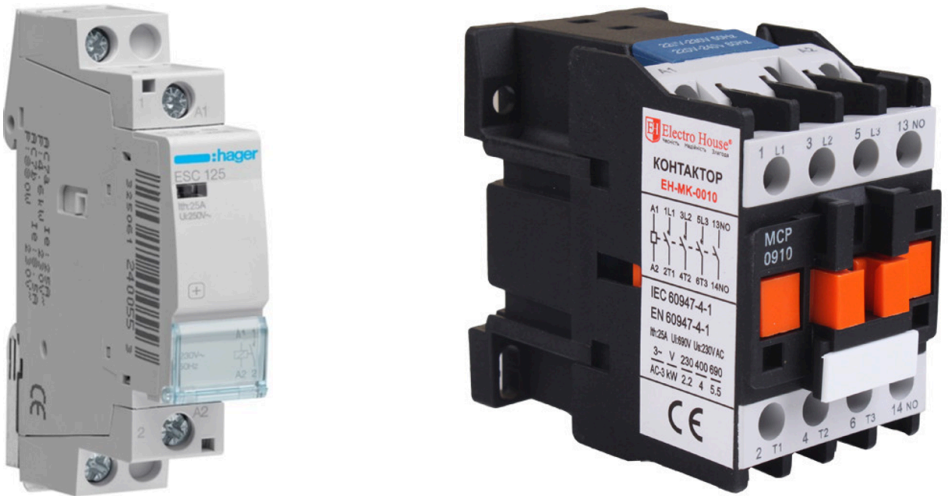

Рисунок 2.1– Види магнітних пускачів на один і чотири полюси

Твердотільні реле (SSR - Solid State Relay) представляють сучасну альтернативу електромеханічним пристроям. Їх принцип дії базується на напівпровідникових ключах (тиристорах, симісторах або IGBT-транзисторах), які керуються через оптичну розв'язку. Керуючий сигнал, часто не перевищуючий 10-30мА при 3-24В, активує світлодіод оптопари, фотоелемент якої відкриває силовий напівпровідниковий елемент. Безінерційність твердотільних реле дозволяє здійснювати комутацію з високою частотою, що неможливо для електромеханічних аналогів, а відсутність механічних частин забезпечує практично необмежений ресурс по кількості спрацьовувань.

Рисунок 2.2– Вигляд твердотільного реле

Для систем з особливими вимогами до надійності та енергоефективності оптимальним є використання бістабільних реле, що потребують імпульсного керуючого впливу лише в момент зміни стану та не споживають енергію для підтримки комутаційного положення.

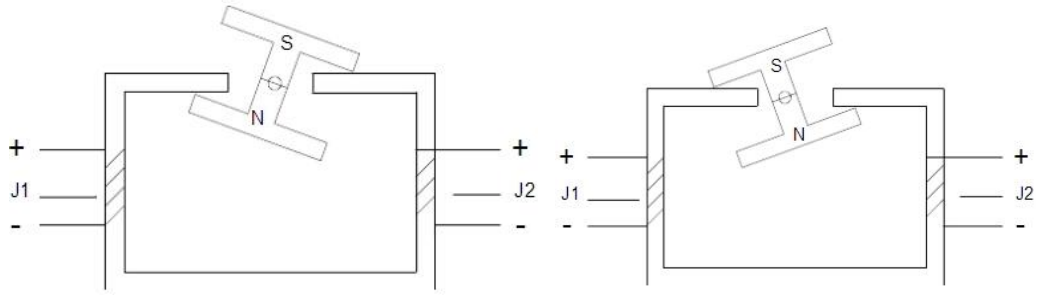

Рисунок 2.3– Принцип роботи бістабільного реле

Вибір конкретного типу комутаційного пристрою визначається комплексом параметрів, включаючи характер навантаження (активне, індуктивне, ємнісне), вимоги до швидкодії, частоти комутацій, рівня електромагнітних завад, умов експлуатації та економічних обмежень. Для систем дистанційного управління критичним є також забезпечення відмовостійкості та прогнозованої поведінки при збоях керуючих сигналів або живлення.

Інтеграція описаних комутаційних пристроїв з сучасними телекомунікаційними системами, такими як GSM-модеми, забезпечує можливість глобального контролю електроспоживання з будь-якої точки світу при мінімальних вимогах до інфраструктури зв'язку.

Бістабільний перемикач, наприклад BIS-411-24V з діапазоном напруги керування 9-30В AC/DC і комутаційною здатністю 16А, має низку суттєвих технічних переваг порівняно з традиційними електромагнітними реле та тиристорними пускачами.

Ключовою особливістю бістабільного перемикача є наявність двох стійких станів, які зберігаються без постійного споживання енергії. Перемикання відбувається лише при подачі короткочасного імпульсу, після чого пристрій залишається в новому стані без додаткових енергетичних витрат. Це докорінно відрізняє його від звичайного електромагнітного реле, де котушка повинна бути постійно під напругою для утримання якоря в активному положенні. При номінальному струмі котушки стандартного реле 25-40 мА економія електроенергії при тривалій експлуатації стає істотною.

В контексті GSM-систем управління навантаженням, що часто живляться від резервних джерел при відключенні основного живлення, енергоефективність бістабільного рішення критично важлива для забезпечення тривалого часу автономної роботи. Розрахунки показують, що при використанні акумулятора ємністю 2000 мАг, звичайне реле скоротить час автономної роботи системи в 5-10 разів.

На відміну від тиристорних пускачів, бістабільні перемикачі мають повну гальванічну розв'язку силових кіл, що запобігає проникненню перешкод у вимірювальні та керуючі кола системи. Тиристорні пускачі генерують високочастотні завади при комутації, вимагають радіаторів для відведення тепла, і в напівпровідниковому ключі завжди присутнє падіння напруги (0,7-1,5В), що призводить до додаткових втрат потужності до 24Вт при струмі 16А.

Додатковою перевагою бістабільного перемикача є збереження останнього стану після відключення живлення системи управління. Це забезпечує детерміновану поведінку підключеного навантаження при збоях в електромережі. Дана властивість особливо цінна в системах з дистанційним управлінням, де оператор може не мати оперативної інформації про стан об'єкта.

Імпульсний режим керування бістабільного перемикача (50-100 мс) дозволяє мультиплексувати операції перемикання в багатоканальних системах, використовуючи один драйвер для послідовного керування декількома навантаженнями, що спрощує схемотехніку і знижує вартість системи управління.

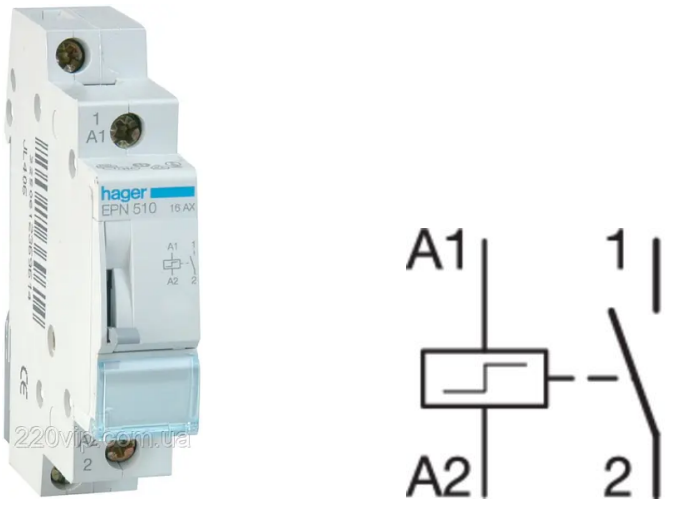

Рисунок 2.4– Зовнішній вгляд і схемне позначення імпульсного реле

Конструктивно, імпульсний перемикач має підвищену стійкість до механічних вібрацій у порівнянні зі звичайними реле, оскільки в стані спокою забезпечується надійна фіксація контактної групи.

2.2.2 Управління бістабільним перемикачем для включення/виключення живлення

Система дистанційного управління навантаженням повинна забезпечувати управління бістабільним електромеханічним реле, що виконує функцію комутації електричного кола навантаження. Бістабільне реле характеризується наявністю двох стійких станів, що зберігаються без подальшого споживання енергії після перемикання. Комутація реле здійснюється шляхом подачі короткочасного імпульсу струму на відповідну обмотку керування. Контролер GSM-модуля повинен генерувати керуючі сигнали тривалістю 50-100 мс для надійного спрацьовування реле, з обмеженням струму до значень, що відповідають технічним характеристикам конкретного типу реле (зазвичай 5-12 В, 30-50 мА).

Управління реле має здійснюватися за допомогою SMS-команд або DTMF-сигналів під час голосового виклику з авторизованих номерів телефонів, що зберігаються в енергонезалежній пам'яті системи. Система повинна підтримувати не менше 8 авторизованих номерів з можливістю їх віддаленого редагування адміністратором системи.

2.2.3 Моніторинг наявності електроживлення

GSM-система повинна постійно контролювати наявність основного електроживлення за допомогою спеціалізованого модуля детектування напруги з гальванічною розв'язкою. Виявлення зникнення напруги має відбуватися протягом не більше 100 мс після фактичного відключення, а визначення відновлення – протягом 1-2 секунд після появи стабільного електроживлення для уникнення хибних спрацьовувань при короткочасних перепадах напруги.

Модуль моніторингу повинен працювати з напругою мережі 220В ±10% при частоті 50Гц та забезпечувати електричну ізоляцію не менше 2500В між високовольтними колами та низьковольтними колами керування. По факту виявлення зміни стану електроживлення система має автоматично

надсилати повідомлення на попередньо визначені номери телефонів з часовою міткою події.

2.2.4 Вимірювання споживаної потужності

Система має забезпечувати вимірювання поточних параметрів електроспоживання навантаження: діючі значення напруги, струму та розрахунок активної потужності. Вимірювальний модуль повинен підтримувати діапазон струмів від 0,1 до 32А з похибкою не більше ±2%, напруги від 180 до 250В з похибкою не більше ±1%. Частота дискретизації сигналів має становити не менше 1кГц для коректного визначення форми сигналу та розрахунку потужності для нелінійних навантажень.

Дані про споживану потужність мають накопичуватися в енергонезалежній пам'яті з інтервалом усереднення, що конфігурується (від 1 хвилини до 1 години), з можливістю віддаленого зчитування через SMS-запити або передачі на сервер через GPRS-з'єднання за розкладом. Система повинна зберігати історію вимірювань не менше 30 днів при 15-хвилинному інтервалі усереднення.

GSM-система повинна надавати інформацію про поточний стан її компонентів та керованого обладнання. Зворотний зв'язок має включати наступні параметри: стан бістабільного реле (включено/виключено), наявність основного електроживлення, поточне значення споживаної потужності, рівень сигналу GSM-мережі (у dBm та якісній оцінці), залишок коштів на SIM-карті, температура всередині корпусу пристрою, заряд резервної батареї (у відсотках та орієнтовний час автономної роботи).

Система повинна відповідати на запити авторизованих користувачів шляхом надсилання SMS-повідомлень з відповідною інформацією за запитом, або через USSD-запити для отримання балансу SIM-карти. Також має бути передбачена можливість автоматичного оповіщення при досягненні граничних значень контрольованих параметрів (наприклад, при перевищенні максимально допустимої потужності або при зниженні заряду резервної батареї нижче 20%).

2.2.5 Робота в умовах періодичного відключення електроживлення

GSM-система має забезпечувати автономну роботу при відключенні основного електроживлення не менше 24 годин у режимі очікування та виконання не менше 20 операцій управління релейним виходом. Система повинна оснащуватися літій-іонним або літій-полімерним акумулятором ємністю не менше 2000 мАг з контролером заряду/розряду, що забезпечує захист від перезаряду та глибокого розряду.

При відновленні основного електроживлення система має автоматично повертатися до нормального режиму роботи та починати процес заряду акумулятора. Час повного заряду резервної батареї не повинен перевищувати 8 годин. Контролер системи повинен реалізовувати алгоритми енергозбереження при роботі від резервного джерела, включаючи переведення GSM-модуля в режим зниженого енергоспоживання, зменшення частоти опитування датчиків та збільшення інтервалів передачі даних.

Система має зберігати всі налаштування та зібрані дані в енергонезалежній пам'яті для запобігання їх втрати при повному розряді акумулятора.

2.3 Вибір GSM-модемему

Основою системи є модем GSM, який відповідає за зв'язок з базовою станцією та дозволяє системі виконувати операції, пов'язані з управлінням та моніторингом.

При проектуванні системи дистанційного керування бістабільним реле критичним компонентом є GSM-модуль, що забезпечує комунікацію через мобільну мережу. Сучасний ринок пропонує різноманітні варіанти модулів, які відрізняються функціональністю, надійністю та вартістю. Детальний аналіз доступних на ринку модулів дозволяє обрати оптимальне рішення для конкретної задачі.

Модуль M590 представляє собою бюджетне рішення для GSM-комунікації, що підтримує основні функції для організації голосового зв'язку та обміну SMS-повідомленнями. Цей модуль привабливий передусім завдяки низькій вартості та широкій доступності на ринку електронних компонентів. Однак M590 має суттєві обмеження у функціональності — підтримує лише обмежений набір AT-команд та працює виключно в діапазонах GSM 900/1800 МГц. Практичний досвід експлуатації виявив проблеми зі стабільністю роботи M590 при тривалій експлуатації: модуль схильний до зависань під час втрати зв'язку з мережею, що вимагає апаратного перезавантаження. Додатковим недоліком є відсутність вбудованої підтримки протоколу TCP/IP, що обмежує можливості розширення функціональності системи в майбутньому. M590 поставляється у компактному SMD-корпусі, що ускладнює його монтаж у аматорських умовах та вимагає використання спеціалізованого обладнання для пайки.

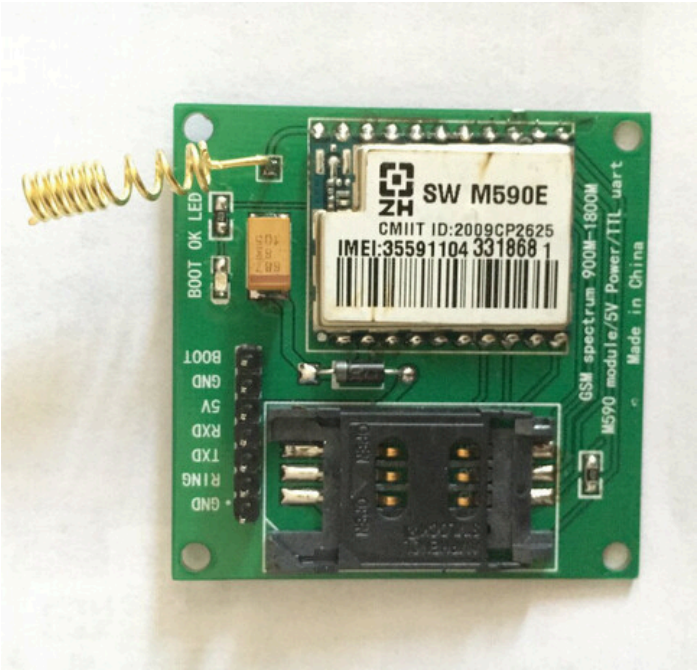

Рисунок 2.5 – плата модуля m590E із холдером SIM карти

Модуль M12 представляє сучасне рішення для вбудованих GSM-додатків з розширеними вимогами до надійності. Цей модуль відрізняється розширеним температурним діапазоном експлуатації (-40°C до +85°C) та посиленим захистом від електромагнітних завад, що робить його

придатним для використання в промислових умовах. M12 забезпечує підтримку розширеного набору AT-команд, включаючи команди для підключення до мереж третього покоління, що гарантує його функціональність на тривалу перспективу при модернізації мобільних мереж. Інтегрований аудіо-кодек високої якості дозволяє використовувати M12 для реалізації систем гучного зв'язку та голосового контролю. Особливістю модуля є наявність вбудованого інтерфейсу для підключення зовнішньої flash-пам'яті, що відкриває можливості для розширеного журналювання подій та збереження конфігураційних даних. M12 характеризується оптимізованим енергоспоживанням з автоматичним переключенням між режимами роботи залежно від поточної активності, що робить його ідеальним для систем з автономним живленням. Інтегровані схеми контролю та стабілізації живлення забезпечують захист від перепадів напруги та полярності, що підвищує загальну надійність системи. Однак M12 має вищу вартість порівняно з більшістю аналогів та обмежену доступність на локальних ринках, що може становити проблему при серійному виробництві пристроїв.

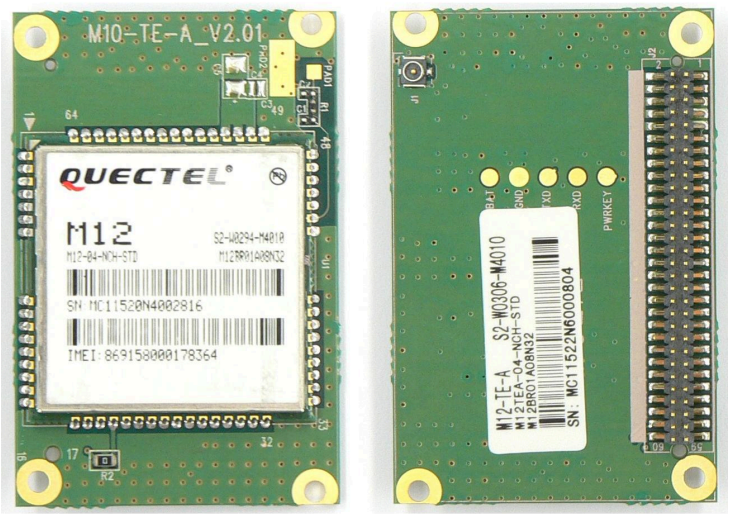

Рисунок 2.6– Плата із встановленим модулем M12

Модулі серії SIM800L характеризуються значно вищою надійністю та розширеною функціональністю порівняно з M590. SIM800L забезпечує роботу в чотирьох частотних діапазонах (GSM 850/900/1800/1900 МГц), що гарантує сумісність з мережами мобільних операторів у більшості країн світу. Ключовою перевагою є вбудована підтримка GPRS класу 12, що дозволяє реалізувати функції передачі даних через інтернет. Модуль підтримує розширений набір AT-команд, включаючи команди для керування GPRS-з'єднанням, що відкриває можливості для розробки більш функціональних систем з віддаленим моніторингом через веб-інтерфейс. SIM800L демонструє високу стабільність роботи завдяки вдосконаленому алгоритму автоматичного відновлення з'єднання з мережею після тимчасової втрати сигналу. Однак для стабільної роботи модуль вимагає якісного джерела живлення з низьким рівнем пульсацій та здатністю забезпечувати пікові струми до 2А при передачі даних. На ринку представлені різні варіанти модулів на базі SIM800L, включаючи готові рішення на платах з інтегрованими стабілізаторами напруги та роз'ємами для антени, що спрощує їх інтеграцію в аматорські проекти.

Рисунок 2.7– Модуль SIM800L встановлений на плату із холдером для міні SIM карти.

Модулі SIM7600 представляють собою високоінтегровані рішення з підтримкою мереж 4G LTE, що забезпечує значно вищу швидкість передачі даних порівняно з GSM/GPRS модулями. Ці модулі підтримують як традиційні GSM-діапазони, так і сучасні LTE-діапазони, що гарантує їх сумісність з мережами різних поколінь. SIM7600 інтегрує GPS/GLONASS приймач, що дозволяє реалізувати функції геолокації без додаткових

компонентів. Вбудований процесор з архітектурою ARM забезпечує можливість запуску користувацьких додатків безпосередньо на модулі, зменшуючи навантаження на основний мікроконтролер системи. Інтерфейси USB, SPI, I2C, UART забезпечують гнучкість при інтеграції в різні системи. Однак використання SIM7600 виправдано лише для складних проектів з потребою у високошвидкісній передачі даних, оскільки ці модулі мають найвищу вартість та енергоспоживання серед розглянутих альтернатив.

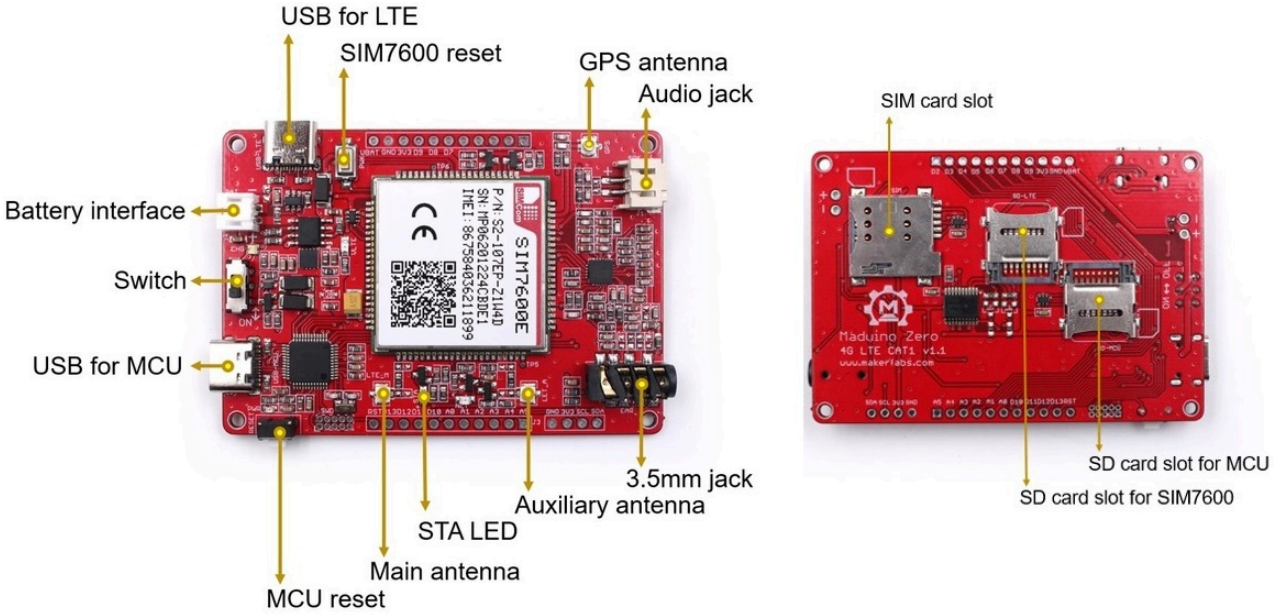

Рисунок 2.8– Модуль SIM7600 на платі Maduino Zero 4G LTE

Модулі EC20 виробництва Quectel орієнтовані на промислові IoT застосування з підвищеними вимогами до надійності та довговічності. Модуль забезпечує роботу в мережах 4G зі швидкістю передачі даних до 150 Мбіт/с. EC20 має вбудовані протоколи TCP/IP, PPP, HTTP, FTP, що спрощує розробку мережевих застосувань. Особливістю модуля є наявність інтерфейсу PCM для підключення зовнішніх аудіо-кодеків та підтримка протоколу DTMF для інтеграції з системами голосового управління. EC20 характеризується високою стійкістю до електромагнітних завад та стабільною роботою в умовах значних коливань температури. Однак висока вартість та складність інтеграції обмежують використання цього модуля у бюджетних проектах.

Модулі SIM900 належать до попереднього покоління, але все ще широко використовуються завдяки високій надійності та розширеній функціональності. Ці модулі підтримують повний спектр GSM-сервісів, включаючи голосові дзвінки, SMS, GPRS, та інтегрований TCP/IP стек. SIM900 характеризується підвищеною стійкістю до несприятливих умов експлуатації, включаючи коливання температури та напруги живлення. Важливою перевагою є наявність великої кількості документації та готових програмних бібліотек для роботи з цим модулем. Втім, SIM900 має більші габарити порівняно з аналогами, що може бути критичним при проектуванні компактних пристроїв. Також модуль характеризується дещо вищим енергоспоживанням, що може бути суттєвим фактором для систем з автономним живленням.

Модуль A6 виробництва компанії AI Thinker позиціонується як оптимальне рішення для IoT-додатків з обмеженим бюджетом. A6 забезпечує роботу в чотирьох діапазонах GSM та підтримує розширений набір функцій, включаючи голосові дзвінки, SMS, GPRS, та TCP/IP. Модуль відрізняється компактними розмірами (22.8 × 17.8 × 2.5 мм) та низьким енергоспоживанням у режимі очікування (близько 1 мА). A6 підтримує повний набір стандартних AT-команд, а також розширені команди для керування мережевими функціями. Висока інтеграція компонентів на кристалі дозволила виробнику досягти оптимального співвідношення ціна/функціональність. Модуль має вбудовані схеми захисту від перенапруги та стабілізації живлення, що підвищує його надійність у реальних умовах експлуатації.

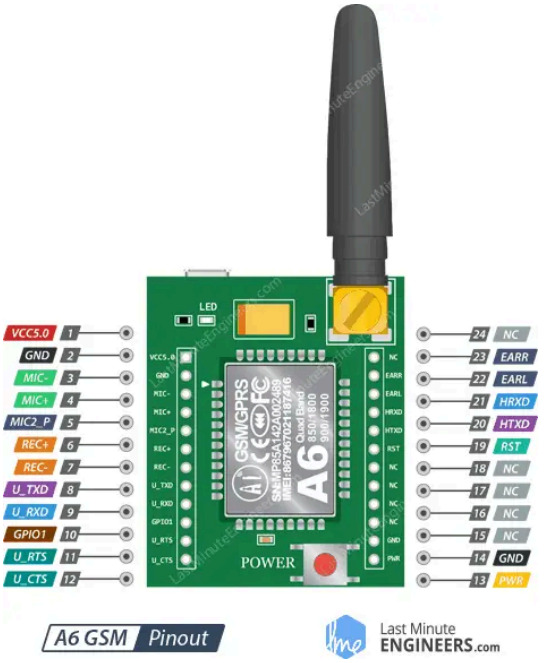

Рисунок 2.9– плата із встановленим модулем A6. Холдер міні SIM карти на іншій стороні плати

За сукупністю всіх параметрів для розробки системи керування бістабільним реле було обрано модуль A6. Ключовими характеристиками, що визначили цей вибір, є: підтримка чотирьох діапазонів GSM (850/900/1800/1900 МГц), вбудований TCP/IP стек, інтерфейс UART з контролем потоку, підтримка швидкості передачі даних до 115200 бод, напруга живлення 3.3-4.2В, пікове споживання струму до 1А при передачі, режим зниженого енергоспоживання з автоматичним пробудженням, вбудований годинник реального часу, розширений температурний діапазон від -30°C до +80°C, та компактні розміри, що дозволяють інтегрувати модуль у корпус невеликих розмірів. Модуль A6 демонструє високу стабільність роботи при тривалій експлуатації без необхідності зовнішнього перезавантаження, що критично важливо для систем автоматизації з мінімальним обслуговуванням. Додатковою перевагою є наявність на ринку готових плат розширення з модулем A6, які включають необхідні периферійні компоненти, що спрощує інтеграцію в електронні пристрої та знижує час розробки.

Для модему GSM A6, який використовується у проекті, необхідно виконати серію ініціалізаційних дій, які забезпечують правильне налаштування та стабільне функціонування. Ці дії включають відправлення АТ-команд, які дозволяють контролювати роботу модема, а також встановлювати та підтримувати зв'язок з мережею GSM.

Рисунок 2.10– Вид плати із встановленим модулем GSM A6

2.4 Вибір мікроконтролера для системи керування живленням

Вибір оптимального мікроконтролера є критичним етапом у проектуванні системи дистанційного керування бістабільним реле. Для забезпечення надійної роботи такої системи мікроконтролер повинен задовольняти специфічним вимогам: стабільна робота в умовах електромагнітних завад, низьке енергоспоживання, достатній набір периферії для забезпечення взаємодії з GSM-модулем, прийнятна вартість та доступність компонентів. Розглянемо основні сімейства мікроконтролерів, представлені на сучасному ринку.

Мікроконтролери сімейства AVR від компанії Microchip (раніше Atmel) широко використовуються в аматорських та комерційних розробках

завдяки розвиненій екосистемі та відносній простоті програмування. AVR характеризуються RISC-архітектурою з гарвардською організацією пам'яті, що забезпечує високу продуктивність при відносно низькій тактовій частоті. Для розробки програмного забезпечення доступні як безкоштовні, так і комерційні середовища розробки, включаючи Atmel Studio, Arduino IDE та інші. Програмування AVR можливе через низьковартісні програматори, такі як USBasp або Arduino в режимі програматора. Однак у контексті розробки системи керування бістабільним реле мікроконтролери AVR демонструють недостатню стійкість до електромагнітних завад, що особливо критично при розміщенні електроніки в безпосередній близькості до комутаційних елементів. Практичні випробування показали, що при комутації індуктивних навантажень без належного екранування AVR-мікроконтролери схильні до зависань та самовільних перезавантажень навіть при наявності базових схем захисту.

Мікроконтролери сімейства PIC від компанії Microchip представляють собою широкий спектр пристроїв різного рівня продуктивності та функціональності. PIC-мікроконтролери характеризуються високою енергоефективністю та наявністю розширених режимів енергозбереження, що робить їх привабливими для систем з автономним живленням. Середовища розробки MPLAB X та MPLAB Code Configurator (безкоштовні) спрощують процес розробки програмного забезпечення, а широкий вибір програматорів, включаючи PICkit та ICD, забезпечує гнучкість при програмуванні та відлагодженні. PIC-мікроконтролери демонструють вищу стійкість до електромагнітних завад порівняно з AVR, однак їх програмування вимагає більш глибокого розуміння архітектури та специфіки роботи периферійних модулів, що ускладнює швидку розробку прототипів.

Мікроконтролери сімейства MSP430 від компанії Texas Instruments позиціонуються як ультраенергоефективні рішення для вбудованих систем. MSP430 відрізняються надзвичайно низьким споживанням електроенергії в

активному режимі та режимі сну, що особливо важливо для систем з автономним живленням. Для розробки програмного забезпечення доступне безкоштовне середовище Code Composer Studio та енергоефективні програматори MSP-FET. MSP430 забезпечують достатню стійкість до електромагнітних завад, однак їх обмежена доступність на локальному ринку та відносно висока вартість ускладнюють використання в бюджетних проектах.

Сімейство мікроконтролерів STM8 від компанії STMicroelectronics представляє оптимальний баланс між вартістю, продуктивністю та надійністю. STM8, зокрема модель STM8S003, характеризується 8-бітною архітектурою з розширеним набором периферійних модулів, включаючи UART, SPI, I²C, що забезпечує гнучкість при взаємодії з зовнішніми компонентами системи, такими як GSM-модулі. Ці мікроконтролери мають вбудовані схеми захисту від перенапруги та демонструють високу стійкість до електромагнітних завад, що критично важливо при роботі в безпосередній близькості до силових комутаційних елементів. STM8S003 працює при напрузі живлення від 2.95В до 5.5В, що спрощує інтеграцію з різними типами живлення, та має низьке енергоспоживання: близько 8 мА в активному режимі при 16 МГц та менше 1 мкА в режимі сну. Для програмування STM8 доступні як комерційні середовища розробки, так і безкоштовні рішення, включаючи IDE IAR Embedded Workbench (з обмеженням розміру коду) та STVD з компілятором Cosmic C. Програмування мікроконтролерів STM8 можливе через недорогі програматори ST-Link v2, які широко доступні на ринку за прийнятною ціною.

За сукупністю всіх параметрів для розробки системи керування бістабільним реле оптимальним вибором є мікроконтролер STM8S003. Цей мікроконтролер забезпечує необхідну функціональність, високу стійкість до електромагнітних завад, низьке енергоспоживання та має прийнятну вартість. Широка доступність на ринку та наявність недорогих засобів

програмування спрощують розробку та масштабування проекту. STM8S003 містить 8 КБ флеш-пам'яті програм, 1 КБ оперативної пам'яті та 128 байт EEPROM, що цілком достатньо для реалізації функціоналу системи керування бістабільним реле з GSM-модулем, включаючи зберігання телефонних номерів авторизованих користувачів та журналювання подій.

3 РОЗРОБКА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ СИСТЕМИ

3.1 Розробка системи команд для управління GSM-системи управління живленням

3.1.1 Функції команд управління через дозвін (CLIP-режим)

Система дозволяє керувати електричним навантаженням за допомогою дозвонів у режимі CLIP (Calling Line Identification Presentation). Ці команди особливо корисні, оскільки вони не тарифікуються, що робить їх доцільним для частого використання. Однак, через їх примітивність, інформативність таких команд обмежена. Основні функції, які можуть бути виконані за допомогою дозвонів, включають ввімкнення, вимкнення або перевключення електричного навантаження.

Коли система отримує дозвін, вона перевіряє номер абонента, порівнюючи його з номерами, збереженими у внутрішньому записнику. Якщо номер відповідає одному з зареєстрованих, система виконує певні дії, які визначені режимом роботи. Наприклад, у випадку включення режиму "1", система вимикає електричне навантаження на певний період часу, а потім увімкює його знову. У режимі "2", система просто переключає стан навантаження. Цей механізм забезпечує базове керування без необхідності відправлення SMS, що економить кошти користувача.

Однак, такий метод керування має свої обмеження. Оскільки інформація передається лише через номер дзвінка, система не може отримати додаткових інструкцій або параметрів. Це означає, що команди, які можуть бути виконані за допомогою дозвону, обмежуються лише базовими операціями. Наприклад, користувач не може вказати конкретний час вимикання або включення навантаження, або відправити будь-які діагностичні запити.

Незважаючи на ці обмеження, команди управління через дозвін є ефективним інструментом для швидкого керування. Вони особливо корисні у ситуаціях, коли користувачу потрібно швидко вмикати або вимикати навантаження, не витрачаючи додаткових коштів на SMS.

3.1.2 Функції команд управління через SMS

SMS-команди надають більшу гнучкість та інформативність у порівнянні з командами через дозвін. Користувач може відправляти SMS з певними інструкціями для керування електричним навантаженням. Ці команди можуть включати в себе більше деталей, таких як конкретні параметри, час виконання або діагностичні запити.

Наприклад, користувач може відправити SMS з командою для включення навантаження, вимкнення або перемикання. Крім того, можуть бути відправлені запити на отримання стану системи, такі як стан електричного навантаження, температура або інші діагностичні параметри.

Система обробляє SMS-команди, аналізуючи текст повідомлення та визначаючи необхідні дії. Це дозволяє реалізувати більш складні сценарії керування, ніж ті, які можуть бути виконані за допомогою дозвону.

Однак, використання SMS-команд має свої недоліки. Оскільки відправлення SMS тарифікується, цей метод може бути менш економічним у порівнянні з дозвонами. Крім того, SMS можуть мати затримки у доставці, що може вплинути на швидкість виконання команд.

SMS-запити інформації дозволяють користувачеві отримувати діагностичні дані про стан системи. Ці запити можуть включати в себе запити на отримання стану електричного навантаження, температури, або інших параметрів системи.

Наприклад, користувач може відправити SMS з запитом на отримання стану електричного навантаження. Система, у свою чергу, відправляє відповідь у вигляді SMS, що містить поточний стан навантаження. Це дозволяє користувачеві отримати необхідну інформацію без необхідності фізично присутність біля системи.

SMS-запити інформації є корисним інструментом для моніторингу системи на відстані. Вони дозволяють користувачеві отримувати поточний стан системи, що може бути важливо для прийняття рішень про керування.

Однак, використання SMS-запитів також має свої недоліки. Оскільки відправлення SMS тарифікується, цей метод може бути менш економічним у порівнянні з іншими методами моніторингу. Крім того, SMS можуть мати затримки у доставці, що може вплинути на швидкість отримання інформації.

3.1.3 Забезпечення безпеки керування

Система має протокол безпеки, який забезпечує захист від несанкціонованого доступу. Основним елементом протоколу безпеки є перевірка номера дзвінка або SMS. Система порівнює номер дзвінка або номер відправника SMS з номерами, збереженими у внутрішньому записнику. Якщо номер не відповідає жодному з зареєстрованих, система не виконує жодних дій.

Крім того, система може бути налаштована так, щоб відправляти SMS-повідомлення про несанкціоновані спроби доступу. Це дозволяє адміністратору системи швидко виявити та відреагувати на можливі загрози.

У цілому, протокол безпеки забезпечує захист від несанкціонованого доступу та надійну роботу системи. Він дозволяє адміністратору системи контролювати доступ до системи та швидко виявити та відреагувати на можливі варти або технічні проблеми.

Система також має механізми для обробки помилок при виконанні команд. Наприклад, якщо система отримує команду, яка не може бути виконана через технічні обмеження або неправильний формат команди, вона відправляє повідомлення про помилку адміністратору системи. Це дозволяє швидко виявити та відреагувати на проблеми, пов'язані з виконанням команд.

Система також має захист від атак через SMS. Наприклад, якщо система виявить серію SMS-повідомлень з незареєстрованих номерів, вона може автоматично заблокувати доступ з цих номерів на певний період часу. Це забезпечує захист від спаму та атак через SMS, що можуть вплинути на нормальне функціонування системи.

3.1.4 Огляд і вибір команд ініціалізації GSM-модему

При старті системи необхідно виконати ініціалізацію модему GSM A6. Цей процес включає відправлення серії AT-команд, які налаштовують модем на роботу з мережею GSM та встановлюють необхідні параметри для подальшого керування. Основними командами ініціалізації є:

Таблиця 3.1– Команди модема A6

Команда	Опис
AT	Перевіряє, чи модем готовий до прийому команд. Відповідь модему "OK" свідчить про готовність.
ATE0	Вимикає ехо команд на термінал. Це корисно для уникнення дублювання команд у вихідному потоці.
AT+CLIP=1	Вмикає режим відображення номера дзвінка (Calling Line Identification Presentation). Це дозволяє модему повідомляти номер вхідного дзвінка.
AT+CSDH=1	Вмикає режим відображення статусу дзвінка. Це дозволяє модему повідомляти статус дзвінка (прийнятий, відхилений тощо).
AT+CMGF=1	Встановлює формат SMS-повідомлень у текстовому режимі. Це дозволяє працювати з SMS-повідомленнями як з звичайним текстом.
AT+CSCS=HEX	Встановлює кодову сторінку для SMS-повідомлень у шістнадцятковому форматі. Це корисно для роботи з символами, які не входять у стандартний ASCII.
AT+CPBS=SM	Встановлює поточний телефонний книжку на SIM-карту. Це дозволяє зберігати та отримувати номери з SIM-карти.

Ці команди відправляються до модему у визначеному порядку, що забезпечує правильне налаштування та готовність модему до подальшої роботи. У разі відсутності позитивного результату на будь-якому етапі, система повинна виявити помилку та виконати дії для її виправлення. Доступним є формування сигналу Reset для повторної ініціалізації модуля модема.

3.1.5 Огляд і вибір команд для роботи з GSM модему з SMS

Після ініціалізації модему, система може приступати до роботи з SMS-повідомленнями. Для цього використовуються команди AT+CMGS та AT+CMGR, які дозволяють відправляти та отримувати SMS-повідомлення.

Таблиця 3.2– Команди для роботи із SMS повідомленнями

Команда	Опис
AT+CMGS	Відправляє SMS-повідомлення на вказаний номер.
AT+CMGR	Отримує SMS-повідомлення з вказаного індексу.

Команда AT+CMGS вимагає вказати номер отримувача та текст повідомлення. Команда AT+CMGR вимагає вказати індекс повідомлення у телефонній книжці. Ці команди забезпечують повний цикл роботи з SMS-повідомленнями, від відправлення до отримання.

3.1.6 Огляд і вибір команд для обробки вхідних викликів

Для обробки вхідних викликів використовується команда AT+CLIP. Ця команда дозволяє модему повідомляти номер вхідного дзвінка. Основною функцією цієї команди є відправлення номера вхідного дзвінка до системи, що дозволяє системі виконувати необхідні дії на основі отриманого номера.

Таблиця 3.3- Команда для обробки вхідних викликів

Команда	Опис
AT+CLIP	Вмикає режим відображення номера дзвінка.

Команда AT+CLIP вимагає встановлення параметру на 1, що вмикає режим відображення номера дзвінка. Ця команда забезпечує можливість отримувати номер вхідного дзвінка, що дозволяє системі виконувати необхідні дії на основі отриманого номера.

3.1.7 Огляд і вибір команд для перевірки стану GSM модему та GSM мережі

Для перевірки стану модему та мережі використовуються наступні команди:

Таблиця 3.4– Команди для перевірку сттупну модему

Команда	Опис
AT+CIPSTATUS	Перевіряє стан з'єднання з мережею.
AT+COPS?	Отримує інформацію про поточного оператора.
AT+CREG?	Отримує інформацію про реєстрацію у мережі.

Команда AT+CIPSTATUS дозволяє перевірити стан з'єднання з мережею, що дозволяє системі виявити та відреагувати на можливі технічні проблеми. Команда AT+COPS? дозволяє отримати інформацію про поточного оператора, що дозволяє системі виявити та відреагувати на можливі зміни оператора. Команда AT+CREG? дозволяє отримати інформацію про реєстрацію у мережі, що дозволяє системі виявити та відреагувати на можливі проблеми з реєстрацією.

3.2 Реалізація команд управління живленням системою

Загалом робота системи тісно повязана із роботою GSM модема, який тримає зв'язок із базовою станцією стільникового звязку. Модем повинен зєднатися із базовою станцією, забезпечивши зв'язок із іншими абонентами системи. пілся цього необхідно налаштувати зв'язок модема вже із контролером який буде опрацьовувати команди абонентів. Тільки після узгодження ланцюжка передачі команд контролер отримує повідомлення від модема у виді дозвонів абонентів чи SMS повідомлень і виконує ці команди.

3.2.1 Реалізація команд управління живленням через дозвін

Програма підтримує кілька типів команд управління через дозвін. Ці команди є поточними, оскільки не тарифікуються, але мають обмежену

інформативність. Основні команди управління через дозвін містять тільки три команди в залежності від режиму роботи. Вони зведні в табл.

Таблиця 3.5 – Команди по дозвону

Команда	Опис
Включення навантаження	Система вмкає електричне навантаження.
Вимкнення навантаження	Система вимикає електричне навантаження.
Перевищення навантаження	Система переключає стан навантаження (вмикання/вимикання).

Ці команди реалізуються після перевірки номера абонента який дозвонився на номер модема. Якщо номер дзвінка збігається з номером, збереженим у записнику, система виконує відповідну дію.

3.2.2 Реалізація команд управління станом системи через SMS

SMS-команди надають більшу гнучкість та інформативність у порівнянні з командами через дозвін. Система підтримує наступні SMS-команди:

Таблиця 3.6– Команди надіслані в SMS для зміни стану системи

Команда	Опис
`mode.reset`	Скидає режим роботи системи до стандартного.
`mode.switch`	Змінює режим роботи системи на альтернативний.
`msocket.on`	Увімкює електричне навантаження.
`msocket.off`	Вимикає електричне навантаження.
`msocket.status`	Відправляє SMS з поточним станом навантаження.
`msocket.accaunt`	Відправляє SMS з інформацією про стан рахунку.
`add`	Додає новий номер до записника.
`number.clear`	Видаляє всі номери з записника.

Ці команди реалізуються шляхом аналізу тексту SMS. Система визначає необхідні дії на основі отриманого тексту та виконує відповідні

операції. Наприклад, якщо користувач відправить SMS з текстом `msocket.on`, система увімкне електричне навантаження.

3.2.3 Реалізація команд запиту інформації у системи за допомою SMS команд

SMS-запити інформації дозволяють користувачеві отримувати діагностичні дані про стан системи. Система підтримує наступні SMS-запити:

Таблиця 3.7–Команди надіслані в SMS для запиту стану

Запит	Опис
`msocket.status`	Відправляє SMS з поточним станом навантаження.
`msocket.accaunt`	Відправляє SMS з інформацією про стан рахунку.
`get_temper`	Відправляє SMS з поточною температурою.

Ці запити реалізуються шляхом аналізу тексту SMS. Система визначає необхідні дії на основі отриманого тексту та відправляє відповідь у вигляді SMS. Наприклад, якщо користувач відправить SMS з текстом `msocket.status`, система відправить SMS з поточним станом навантаження.

3.3 Реалізація протоколу роботи мікроконтролера із GSM модемом і GSM мережею

3.3.1 Очікування підключення до GSM мережі

При подачі живлення на модем A6 відбувається його автоматичний запуск. Модем починає виконувати внутрішню ініціалізацію та надсилає через UART послідовність системних повідомлень. Контролер має постійно зчитувати дані з порту UART та аналізувати отримані повідомлення.

Спочатку модем надсилає службові повідомлення з префіксом ^CINIT:, що сигналізують про ініціалізацію різних підсистем:

```
^CINIT: 2, 32, 41891
^CINIT: 1, 0, 0
^STN: 38
^CINIT: 4, 8192, 38
^CINIT: 8, 2048, 1
```

```
^CINIT: 16, 0, 1638410
^CINIT: 32, 0, 0
```

Після внутрішньої ініціалізації модем розпочинає процес підключення до стільникової мережі. Спершу з'являється повідомлення +CREG: 0, яке означає, що модем не зареєстрований у мережі й починає пошук базової станції. Після знаходження доступної мережі з'являються повідомлення:

```
+CIEV: service, 1
+CIEV: roam, 0
```

Повідомлення +CIEV: service, 1 вказує на наявність сигналу мережі, а +CIEV: roam, 0 свідчить про те, що модем знаходиться в домашній мережі (не в роумінгу). Далі модем автоматично реєструється в мережі та надсилає повідомлення +CREG: 1, що підтверджує успішну реєстрацію.

Отримавши повідомлення про успішну реєстрацію, необхідно зачекати приблизно 10 секунд для стабілізації підключення. Після цього слід перевірити статус реєстрації, надіславши команду:

```
AT+CREG?
```

Модем має відповісти:

```
+CREG: 1,1
```

де перша цифра "1" означає, що сповіщення про статус мережі увімкнено, а друга "1" свідчить про успішну реєстрацію в домашній мережі. Обов'язково перевіряємо, щоб останній байт у відповіді на цю команду був рівний 1. Якщо отримано інший статус, це свідчить про проблеми підключення до мережі, тому необхідно сигналізувати про помилку та виконати скидання модуля.

3.3.2 Налаштування GSM модему для роботи із мікроконтролером

Після успішного підключення до мережі необхідно сконфігурувати модем відповідно до вимог додатку. Налаштування виконуються шляхом надсилання відповідних AT-команд з очікуванням успішної відповіді після кожної.

Спочатку встановлюємо базові параметри функціонування модему:

```
ATS0=0
```

Ця команда вимикає автоматичний підйом трубки при вхідному дзвінку. Очікуємо відповідь OK з переведенням рядка ($0D,$0A).

```
ATV0
```

Команда змінює режим відповідей з текстового на числовий. Замість "OK" модем відповідатиме "0". Після цієї команди очікуємо відповідь 0, а не OK.

Наступним кроком налаштовуємо обробку дзвінків та SMS:

```
AT+CLIP=1
```

Ця команда вмикає визначення номера вхідного дзвінка. Очікуємо відповідь 0 завдяки попередній команді ATV0.

```
AT+CMGF=1
```

Встановлюємо текстовий режим для SMS повідомлень. Режим "1" означає текстовий формат, що спрощує обробку повідомлень. Очікуємо відповідь 0.

```
AT+CSDH=1
```

Вмикаємо відображення детальної інформації про SMS повідомлення. Очікуємо відповідь 0.

```
AT+CSCS="HEX"
```

Встановлюємо кодування символів у форматі HEX для SMS. Очікуємо відповідь 0.

```
AT+CPBS="SM"
```

Вибираємо пам'ять SIM-карти для зберігання повідомлень. Очікуємо відповідь 0.

3.3.3 Вибір команд очищення пам'яті SIM карти та підготовка GSM модема до роботи в складі системи керування живленням

Для запобігання переповненню пам'яті SIM-карти необхідно видалити всі старі SMS повідомлення. Це робиться шляхом послідовного видалення повідомлень з перших чотирьох слотів пам'яті:

```
AT+CMGD=1
AT+CMGD=2
AT+CMGD=3
AT+CMGD=4
```

Після кожної команди очікуємо відповідь 0, що підтверджує успішне видалення повідомлення або відсутність повідомлення у вказаному слоті.

Після завершення всіх налаштувань модем готовий до основної роботи. Переходимо в режим моніторингу, при якому контролер безперервно аналізує потік даних з UART інтерфейсу модему, шукаючи ключові слова, що свідчать про різні події:

+CREG: - зміна статусу реєстрації в мережі. При надходженні цього префіксу необхідно перевірити наступний байт після двокрапки та пробілу. Цей байт засовуємо в буфер стану мережі, проштовхуючи всі попередні значення вправо.

+CLIP: - сигналізує про вхідний дзвінок. При надходженні цього префіксу потрібно знайти перший символ лапок ($22), і починаючи з наступного байта до наступного символу лапок виокремити номер телефону, видаляючи всі нецифрові символи. Номер зберігається у вихідний параметр. Далі слід очікувати, поки не надійде повідомлення ERROR з переведенням рядка ($0D,$0A), і встановити прапорець вказівки на вхідний дзвінок.

+CMT: - сигналізує про надходження нового SMS повідомлення. При надходженні цього префіксу також шукаємо символ лапок ($22), і з наступного байта до наступного символу лапок виокремлюємо номер відправника. Після цього пропускаємо всі символи до переведення рядка ($0D,$0A), а наступні символи до нового переведення рядка записуємо як текст SMS повідомлення. Встановлюємо прапорець отримання нового SMS.

По завершенні всіх налаштувань модем перебуває в стані постійного моніторингу та готовий обробляти вхідні виклики та SMS повідомлення. Контролер повинен безперервно зчитувати дані з UART інтерфейсу та аналізувати їх на наявність зазначених вище ключових слів, відповідно реагуючи на різні події.

У разі втрати зв'язку з мережею (надходження повідомлення +CREG: 0) необхідно визначити подальшу стратегію: очікувати автоматичного

відновлення зв'язку або виконати примусове перезавантаження модему шляхом надсилання команди AT+CFUN=1,1.

Таким чином, модем проходить ініціалізацію, налаштування та переходить у стан прослуховування, готовий до обробки вхідних викликів та SMS повідомлень.

3.4 Розробка порядку роботи мікроконтролера в режимі очікування команд від GSM модема

3.4.1 Очікування та виявлення подій

Після завершення ініціалізації та конфігурації модем переходить у режим очікування. В цьому стані мікроконтролер безперервно зчитує дані з інтерфейсу UART модему та аналізує інформаційний потік на наявність службових повідомлень. Основними подіями, які потребують обробки, є вхідні дзвінки, надходження SMS повідомлень, а також зміни в стані підключення до мережі.

3.4.2 Виявлення вхідного дзвінка та ідентифікація абонента

При надходженні вхідного дзвінка модем автоматично надсилає повідомлення з префіксом "RING", після чого, якщо активована функція визначення номера командою AT+CLIP=1, надсилає рядок з інформацією про абонента. Ця послідовність виглядає наступним чином:

```
RING
+CLIP: "+380XXXXXXXXX",145,"",,"",0
```

Повідомлення "RING" генерується модемом повторно з кожним гудком, тому його надходження служить індикатором продовження виклику. Обробник подій має регулярно перевіряти наявність цього повідомлення в буфері, очищаючи його після обробки.

Для виокремлення номера телефону абонента необхідно проаналізувати рядок, що починається з "+CLIP:". Номер знаходиться між першою парою лапок. Алгоритм виокремлення номера передбачає пошук першого символу лапок ($22) в повідомленні, запам'ятовування всіх подальших символів до наступного символу лапок, та відкидання

нецифрових символів. Така фільтрація дозволяє отримати чистий номер без префіксу "+", що спрощує подальше порівняння з записами в записнику авторизованих номерів.

При виявленні вхідного дзвінка система має виконати порівняння отриманого номера з базою авторизованих номерів для прийняття рішення про подальші дії: автовідповідь, відхилення виклику або запис у журнал подій.

3.4.3 Виявлення SMS повідомлення та його обробка

Виявлення нового SMS повідомлення відбувається при надходженні від модему повідомлення з префіксом "+CMT:". При налаштуванні модему в текстовому режимі SMS командою AT+CMGF=1 та увімкненні функції негайного сповіщення про нові повідомлення, модем автоматично надсилає інформацію про нове SMS у форматі:

```
+CMT: "+380XXXXXXXXX",,"YY/MM/DD,HH:MM:SS+ZZ"
Текст повідомлення
```

Для отримання номера відправника необхідно виконати аналогічну до обробки вхідного дзвінка процедуру: знайти символ лапок ($22), скопіювати символи до наступних лапок, відфільтрувати нецифрові символи.

Для отримання тексту повідомлення необхідно пропустити всі символи до першого символу переведення рядка ($0D,$0A), який відокремлює заголовок від тексту повідомлення. Весь подальший текст до наступного символу переведення рядка є власне тілом SMS повідомлення. Цей текст слід скопіювати у відведений для нього буфер для подальшої обробки.

Обробка тексту SMS може включати аналіз командних слів, якщо система підтримує віддалене керування через SMS, або просто зберігання повідомлення у випадку, якщо система виконує функції віддаленого моніторингу.

3.4.4 Робота з телефонним записником авторизованих абонентів

Модем A6 підтримує доступ до телефонної книги, розташованої на SIM-карті. Цей функціонал можна використовувати для зберігання та зчитування списку авторизованих абонентів, які мають право на віддалене керування системою.

3.4.5 Зчитування записів з телефонної книги

Для доступу до телефонної книги SIM-карти необхідно спочатку вибрати її як джерело даних командою:

```
AT+CPBS="SM"
```

Після отримання підтвердження "OK" можна приступати до зчитування записів. Для визначення кількості записів у телефонній книзі використовується команда:

```
AT+CPBS?
```

Модем відповідає рядком такого формату:

```
+CPBS: "SM",N,M
```

де N - кількість використаних записів, а M - загальна ємність телефонної книги SIM-карти.

Для зчитування конкретного запису з телефонної книги використовується команда:

```
AT+CPBR=X
```

де X - індекс запису (від 1 до N). У відповідь модем надсилає інформацію про запис у форматі:

```
+CPBR: X,"НОМЕР",ТИЙП,"ІМ'Я"
```

Для зчитування всіх записів можна використовувати діапазон індексів:

```
AT+CPBR=1,N
```

У цьому випадку модем надішле послідовність відповідей для всіх існуючих записів.

3.4.6 Додавання нових авторизованих абонентів

Для додавання нового запису в телефонну книгу SIM-карти використовується команда:

```
AT+CPBW=X,"НОМЕР",ТИЙП,"ІМ'Я"
```

де:

X - індекс запису (1-N, або пропущений для автоматичного вибору вільної позиції)

"НОМЕР" - номер телефону в міжнародному форматі

ТИЙП - тип номера (зазвичай 145 для міжнародних номерів з префіксом "+" або 129 для локальних номерів)

"ІМ'Я" - ім'я абонента або тег авторизації

Приклад команди для додавання нового авторизованого абонента:

```
AT+CPBW=,"380XXXXXXXXX",145,"USER1"
```

При успішному виконанні команди модем відповідає "OK", що підтверджує збереження запису в телефонній книзі.

3.4.7 Механізм авторизації нових абонентів

Для реалізації механізму віддаленої авторизації нових абонентів можна використовувати SMS-повідомлення з спеціальним форматом. Припустимо, система має визначеного головного адміністратора, номер якого попередньо записаний у телефонну книгу з спеціальним тегом "ADMIN".

При отриманні SMS з командою авторизації система перевіряє, чи походить повідомлення від абонента з тегом "ADMIN". Якщо так, вона аналізує структуру повідомлення, яке може мати, наприклад, формат:

```
ADD +380YYYYYYY USER2
```

де "ADD" - команда додавання, "+380YYYYYYY" - номер нового авторизованого абонента, а "USER2" - його тег.

Після розбору команди система формує та відправляє модему AT-команду для додавання нового запису в телефонну книгу:

```
AT+CPBW=,"+380YYYYYYY",145,"USER2"
```

Після успішного додавання система може відправити підтверджувальне SMS як адміністратору, так і новому авторизованому абоненту.

3.4.8 Обробка стану підключення до мережі

В режимі очікування важливо також контролювати стан підключення до мережі, оскільки можливі тимчасові втрати зв'язку через погані умови прийому або проблеми на стороні оператора.

При надходженні повідомлення з префіксом "+CREG:" система повинна аналізувати наступний за ним байт. Значення "1" вказує на успішну реєстрацію в домашній мережі, значення "2" свідчить про пошук мережі, значення "0" сигналізує про відсутність реєстрації, а значення "5" вказує на реєстрацію в роумінгу.

У випадку втрати зв'язку (отримання значення "0" або "2") система має увімкнути таймер очікування відновлення зв'язку. Якщо протягом визначеного часу (наприклад, 2-3 хвилин) зв'язок не відновлюється, доцільно виконати перезавантаження модему командою:

```
AT+CFUN=1,1
```

Така стратегія забезпечує максимальну надійність роботи системи навіть в умовах нестабільного стільникового зв'язку.

3.4.9 Перевірка наявності непрочитаних SMS

Якщо система налаштована на збереження вхідних SMS в пам'яті SIM-карти, а не на негайну обробку, важливо періодично перевіряти наявність непрочитаних повідомлень. Для цього використовується команда:

```
AT+CMGL="ALL"
```

У відповідь модем надсилає список всіх збережених повідомлень з їх індексами, статусами, номерами відправників, датою та часом отримання, а також текстом повідомлень. Система аналізує цей список, обробляє повідомлення та видаляє їх з пам'яті командою:

```
AT+CMGD=X
```

де X - індекс обробленого повідомлення.

3.4.10 Перевірка рівня сигналу мережі

В режимі очікування корисно періодично перевіряти рівень та якість сигналу стільникової мережі. Для цього використовується команда:

```
AT+CSQ
```

Модем відповідає у форматі:

```
+CSQ: XX,YY
```

де XX - рівень сигналу (0-31, де 31 - максимальний рівень), а YY - якість сигналу.

На основі цієї інформації система може оцінювати надійність поточного підключення та, за необхідності, сигналізувати про проблеми зі зв'язком.

3.5 Реалізація програми для автономної GSM-системи управління

3.5.1 Ініціалізація системи при включенні

Текст програми для управління живленням за допмогою GSM модема наведено в додатках. При включенні системи, першим кроком є ініціалізація усіх необхідних компонентів для нормальної роботи. Цей процес включає в себе налаштування модему GSM, ініціалізацію периферійних пристроїв, налаштування таймерів та інших важливих системних параметрів.

У наданій програмі, ініціалізація системи розпочинається з виклику функції `perif_init()`, яка відповідає за ініціалізацію периферійних пристроїв, таких як GPIO, UART, таймери та аналого-цифровий конвертер (ADC). Ця функція включає в себе налаштування швидкості UART, ініціалізацію таймерів для виміру часу та інші важливі налаштування.

Після ініціалізації периферійних пристроїв, програма викликає функцію `modem_status()`, яка відповідає за ініціалізацію модему GSM. Ця функція відправляє серію AT-команд для перевірки готовності модему, налаштування параметрів роботи та перевірки з'єднання з мережею GSM. Якщо модем не готовий або не може з'єднатися з мережею, програма виконує процедуру відновлення, яка включає в себе перезавантаження модему та повторну спробу ініціалізації.

3.5.2 Реалізація обробки вхідних дзвінків (без відповіді)

Після ініціалізації, система перебуває у режимі очікування вхідних дзвінків та SMS-повідомлень. Для обробки вхідних дзвінків, програма використовує функцію `getstr()`, яка аналізує вхідний потік даних від модему та визначає тип отриманого повідомлення.

У випадку вхідного дзвінку, модем відправляє повідомлення у форматі "+CLIP:", яке містить номер вхідного дзвінка. Програма аналізує це повідомлення та визначає номер дзвінка. Якщо номер відповідає одному з зареєстрованих у системі, програма виконує відповідні дії, такі як включення або вимкнення електричного навантаження.

Оскільки система не призначена для ведення розмов, вона не відповідає на вхідні дзвінки. Замість цього, вона просто аналізує номер дзвінка та виконує необхідні команди керування.

3.5.3 Реалізація прийому та обробки SMS-команд

Окрім вхідних дзвінків, система також може приймати та обробляти SMS-команди. Для цього програма використовує функцію `getstr()`, яка аналізує вхідний потік даних від модему та визначає тип отриманого повідомлення.

У випадку отримання SMS-повідомлення, модем відправляє повідомлення у форматі "+CMT:", яке містить текст повідомлення та номер відправника. Програма аналізує текст повідомлення та визначає команду, яку необхідно виконати.

Наприклад, якщо отримане SMS-повідомлення містить команду "msocket.on", програма виконує команду керування для увімкнення електричного навантаження. Якщо отримане SMS-повідомлення містить команду "msocket.status", програма відправляє SMS-повідомлення з поточним станом навантаження.

3.5.4 Реалізація формування та відправки звітів про стан системи

Окрім виконання команд керування, система також може відправляти звіти про стан. Ці звіти можуть бути відправлені за запитом користувача або автоматично за певним графіком.

Для формування та відправки звітів, програма використовує функцію `sendSMS()`, яка формує текст звіту та відправляє його на вказаний номер. Наприклад, якщо користувач відправить SMS-команду "msocket.status", програма відправить SMS-повідомлення з поточним станом навантаження.

Крім того, програма може бути налаштована на відправлення автоматичних звітів про стан за певним графіком. Це може бути корисно для моніторингу системи на відстані.

3.5.5 Реалізація обробки станів при відключенні електроживлення

У випадку відключення електроживлення, система повинна коректно обробити цей стан та виконати необхідні дії для збереження стану та даних. Це включає в себе збереження стану навантажень, номерів та інших важливих параметрів.

У наданій програмі, обробка відключення електроживлення реалізована за допомогою функції `overtemp()`, яка викликається у випадку виявлення надмірної температури. Ця функція вимикає електричне навантаження та зберігає стан у пам’яті.

3.5.6 Реалізація процедури відновлення роботи після повернення електроживлення

Після повернення електроживлення, система повинна відновити свою роботу та відновити попередній стан. Це включає в себе ініціалізацію модему, периферійних пристроїв та відновлення стану навантажень.

У наданій програмі, процедура відновлення роботи реалізована у функції `modem_status()`, яка перевіряє стан модему та мережі GSM. Якщо модем не готовий або не може з'єднатися з мережею, програма виконує

процедуру відновлення, яка включає в себе перезавантаження модему та повторну спробу ініціалізації.

Для забезпечення автономної роботи системи, необхідно реалізувати алгоритм енергозбереження. Цей алгоритм повинен мінімізувати витрати енергії та продовжувати роботу системи при відсутності електроживлення.

У наданій програмі, алгоритм енергозбереження реалізований за допомогою управління станом модему та периферійних пристроїв. Програма може вимикати модем та периферійні пристрої, коли вони не потрібні, та увімкювати їх тільки при необхідності. Це мінімізує витрати енергії та забезпечує автономну роботу системи.

3.6 Реалізація початкової ініціалізації GSM модема та конфігурація системи

При включенні пристрою мікроконтролер виконує серію діагностичних операцій, перевіряючи працездатність основних компонентів системи. Спочатку зчитується стан фізичної кнопки на корпусі пристрою. Якщо кнопка знаходиться в натиснутому стані, система переходить у режим первинної конфігурації, який дозволяє зареєструвати номер супервізора.

У режимі первинної конфігурації система включає таймер на 5 хвилин і переходить у стан очікування вхідного дзвінка. Перший абонент, що здійснить дзвінок протягом цього часу, автоматично реєструється як супервізор системи. Його номер записується в енергонезалежну пам'ять як перший запис у телефонній книзі пристрою. Цей механізм забезпечує просту, але надійну процедуру первинної авторизації без необхідності використання додаткових інтерфейсів програмування.

Незалежно від стану кнопки, мікроконтролер виконує процедуру ініціалізації GSM модема F6. Процес ініціалізації включає наступні етапи:

Подача живлення на модем та очікування його готовності до роботи (приблизно 10-15 секунд)

Відправка AT-команди "AT" для перевірки зв'язку з модемом

Налаштування параметрів текстового режиму для SMS: AT+CMGF=1

Конфігурація формату номера абонента: AT+CSDH=1

Вибір кодування повідомлень: AT+CSCS="GSM"

Налаштування автоматичного повідомлення про вхідні SMS: AT+CNMI=2,2,0,0,0

Перевірка реєстрації в мережі: AT+CREG?

Визначення рівня сигналу: AT+CSQ

Лише після успішного завершення ініціалізації модема система переходить до основного режиму роботи. Якщо при ініціалізації виникають помилки (відсутність відповіді на AT-команди, неможливість зареєструватися в мережі), система виконує перезавантаження модема та повторює процедуру. Після декількох невдалих спроб система переходить у режим очікування з періодичними спробами відновлення працездатності.

3.7 Реалізація управління авторизованими користувачами

Система підтримує ієрархічну структуру користувачів з розподілом прав доступу. На вершині ієрархії знаходиться супервізор — користувач з максимальними правами, що має можливість керувати списком авторизованих користувачів системи.

При отриманні вхідного дзвінка або SMS система виконує аутентифікацію абонента за його номером. Процес аутентифікації включає наступні кроки:

Визначення номера абонента з вхідного дзвінка або SMS

Приведення номера до стандартного формату (міжнародний формат з кодом країни)

Пошук номера в базі авторизованих користувачів

Визначення рівня доступу користувача

Прийняття рішення про обробку команди відповідно до рівня доступу

3.8 Режими управління бістабільним реле

Система підтримує декілька режимів управління бістабільним реле, що забезпечує гнучкість у використанні та адаптацію до різних сценаріїв експлуатації:

Режим прямого управління — при вхідному дзвінку від авторизованого користувача система змінює стан реле на протилежний (вмикає, якщо було вимкнено, і навпаки). Цей режим є найпростішим і не вимагає додаткових налаштувань.

Режим командного управління — керування здійснюється через SMS-команди:

"ON" або "ВКЛ" — включення живлення

"OFF" або "ВИКЛ" — відключення живлення

"RESET" або "РЕСЕТ" — короткочасне відключення з подальшим включенням (перезавантаження керованого обладнання)

Режим температурного контролю (опціонально, при наявності датчика температури) — автоматичне керування реле залежно від температури. порогові значення температури в градусах Цельсія задаються в константах програми.

Для забезпечення надійності системи реалізовано захисні механізми від частого перемикання реле. Система не дозволяє виконувати повторне перемикання реле раніше, ніж через налаштований інтервал часу (за замовчуванням — 5 секунд). Це запобігає можливому пошкодженню реле при масовому надходженні команд.

3.9 Реалізація моніторингу стану системи та керованого об'єкта

Система забезпечує комплексний моніторинг як власного стану, так і стану керованого об'єкта. Авторизовані користувачі можуть отримати інформацію через SMS-запити:

Запит поточного стану реле — SMS-команда "STATUS" або "СТАН". У відповідь система надсилає SMS з інформацією про поточний стан реле (включено/виключено) та час перебування в цьому стані.

Запит споживаної потужності — SMS-команда "POWER" або "ПОТУЖНІСТЬ". Система аналізує дані з вбудованого датчика струму та розраховує миттєву споживану потужність у ватах, яку надсилає у відповідному SMS.

Запит статистики енергоспоживання — SMS-команда "ENERGY" або "ЕНЕРГІЯ". У відповідь система надає інформацію про споживання електроенергії за останню добу та поточний місяць.

Запит стану рахунку SIM-карти — SMS-команда "BALANCE" або "БАЛАНС". Система формує USSD-запит до оператора зв'язку (наприклад, *100# для більшості операторів) і пересилає отриману відповідь у вигляді SMS.

Для забезпечення проактивного моніторингу система налаштована на автоматичну відправку SMS-повідомлень при настанні критичних подій:

Зміна стану реле (включення/відключення)

Відновлення живлення після аварійного відключення

Критично низький баланс на SIM-карті (менше заданого порогу)

Аномально високе споживання потужності (вище заданого порогу)

Пропадання GSM-сигналу та подальше відновлення зв'язку

3.10 Реалізація обробки вхідних повідомлень та команд

Для ефективної обробки вхідних даних система використовує подієвий механізм з буферизацією команд. При надходженні даних від GSM модема через UART інтерфейс відбувається їх накопичення в кільцевому буфері до отримання повного повідомлення (завершення рядка або завершення блоку SMS-повідомлення).

Алгоритм обробки вхідних даних включає наступні етапи:

Розпізнавання типу події (вхідний дзвінок, SMS, відповідь на AT-команду, USSD-відповідь)

Парсинг вмісту повідомлення для визначення номера абонента та команди

Перевірка авторизації абонента

Інтерпретація команди та формування відповідної дії

Виконання дії та підготовка відповіді

Відправка відповіді абоненту (за необхідності)

Для підвищення стійкості системи реалізовано механізм обробки помилок, що дозволяє відновлювати роботу після нештатних ситуацій. При виникненні помилки зв'язку з модемом система виконує його перезавантаження та повторну ініціалізацію. У випадку критичних помилок дані про них зберігаються у внутрішньому журналі, доступному для аналізу через спеціальні команди.

3.10.1 Послідовність виконання команд

Програма виконується циклічно пеервіряючи стан модему і можливу вхідну команду. Загалом діаграма роботи основного циклу наведена рис. 3.1.

Рисунок 3.1– Діаграма роботи основного циклу програми

Після запуску відбувається ініціалізація периферії: тактова система, GPIO, UART, ADC, таймери. Потім подається живлення на модем, але

тільки після короткої діагностики. Система переходить до фази встановлення з’єднання з мережею — виконується функція modem_status().

Ця функція:

керує подачею живлення на модем (через сигнал MODEMV),

очікує появи повідомлення +CREG: x, де x — код стану мережі (наприклад, +CREG: 1 — зареєстровано в домашній мережі),

якщо при встановленні звязку базовою станцією сталася неусувна помилка, то запускається візуальна індикація цього стану за допомогою світлодіодів на передній панелі пристрою

після вдалої реєстрації модема в мережі надсилає базові AT-команди: відключення exo (ATE0), включення визначення номера (+CLIP=1), перехід у текстовий режим для SMS (+CMGF=1).

Перевірка через +CREG не дає повної гарантії, що модем може надсилати/приймати дані. Тому додатково надсилається USSD-запит (AT+CUSD=1,"..."), який дозволяє переконатися, що модем не лише зареєстрований, але й може працювати з мережею (тобто, SIM активна, баланс ненульовий, USSD-служба працює). Це єдиний надійний спосіб визначити, що модем справді "живий" у GSM-сенсі.

Зберігання поточних параметрів системи здійснюється телефонній книжці сімкарти. Телефонна книжка SIM-карти використовується як постійне сховище конфігураційних параметрів: номерів абонентів, USSD-команд, режимів роботи та інших даних. Модем A6 працює з телефонною книжкою через стандартні AT-команди AT+CPBR (читання) і AT+CPBW (запис), а сам доступ налаштовано на використання пам’яті SIM-карти (AT+CPBS=SM).

Кожен запис у телефонній книжці має фіксовану структуру і зберігає кілька полів: номер телефону, тип номера (зазвичай 129 — національний формат), і короткий ідентифікатор — так званий text tag (у жаному випадку використовується "my" як маркер). Параметри, які зберігаються у вигляді

рядків, записуються у поле номера, навіть якщо сам запис логічно не є номером телефону (наприклад, "22" для режиму або "*100#" для USSD).

Щоб прочитати значення з комірки, надсилається команда AT+CPBR=n, де n — номер запису. У відповідь модем повертає рядок +CPBR: n,"string",type,"tag", з якого програма витягує вміст поля "string". Цей рядок інтерпретується згідно з призначенням: як номер телефону, або як код USSD, або як код режиму. Ідентифікація виконується за фіксованим розміщенням у комірках — наприклад, запис №1 зарезервовано для супервізора, запис з номером RECUSSD — для USSD-команди, а RECMODE — для режиму.

Щоб записати новий рядок у певну комірку, використовується команда AT+CPBW=n,"string",129,"my": параметр n задає позицію в записній книжці, string — це власне текстовий вміст (номер або конфігурація), 129 — код типу номера, my — необов’язкове текстове поле, яке у цьому випадку не використовується для логіки, але може слугувати як маркер. Для оновлення режиму або USSD достатньо перезаписати відповідну комірку цими ж засобами.

Таким чином, структура комірки є однорідною незалежно від призначення: інтерпретація значення відбувається на рівні програми. Це дозволяє зберігати не лише телефонні номери, а й будь-які короткі конфігураційні параметри, з гарантією збереження навіть після вимкнення живлення.

Телефонна книга SIM-карти у цій системі використовується не лише для збереження параметрів (таких як USSD-команда чи режим реле), а й для зберігання авторизованих номерів абонентів, включно із супервізором. Розподіл комірок жорстко зафіксовано у коді, що дає змогу програмі визначати, яку роль виконує кожен запис.

Комірка з номером 1 зарезервована для супервізора — основного авторизованого користувача, який має розширені права. Його номер зберігається у форматі текстового рядка (без символу "+", який у процесі

вилучається функцією dplus()), і саме цей запис використовується для перевірки легітимності абонента при обробці викликів і SMS. Усі перевірки відбуваються шляхом послідовного читання комірок через AT+CPBR=n і порівняння прочитаного номера з тим, що надійшов.

Для зберігання USSD-команди, яка використовується для перевірки стану мережі або рахунку, виділено окрему комірку. Її номер визначено через ідентифікатор RECUSSD, який у коді передається у функцію chtostr() при формуванні команди AT+CPBR=RECUSSD. У відповідь зчитується текстовий рядок — наприклад, *100# або інший операторський USSD-запит, який потім використовується як аргумент у команді AT+CUSD=1,"…".

Аналогічно, для зберігання режиму роботи реле використовується комірка з номером, заданим через RECMODE. Зміст цієї комірки — рядок із двома символами, наприклад "11" або "22", який зчитується в str_mode[]. Перший символ визначає режим роботи під час дзвінка (наприклад, "1" — короткочасне вимкнення з наступним увімкненням, "2" — перемикання стану реле). Значення "11" або "22" також зберігаються у форматі телефонного номера, тобто як текст у полі number при записі AT+CPBW.

Для зберігання додаткових авторизованих номерів абонентів, які можуть надсилати команди, використовуються комірки з номерами від 2 і до MAXNBOOK включно (де MAXNBOOK — максимально допустима кількість записів, що читається модулем). Це дозволяє зберігати до MAXNBOOK - 1 додаткових номерів, які мають обмежений або повний доступ, залежно від контексту обробки SMS. Додавання виконується за допомогою SMS-команди від супервізора, яка має формат addN+380..., де N — номер комірки (використовується як індекс у команді AT+CPBW=N+1).

Таким чином, адресний простір записної книжки SIM поділено логічно:

1 — супервізор.

2 … MAXNBOOK — інші авторизовані абоненти.

RECUSSD — USSD-команда для перевірки зв’язку.

RECMODE — режим керування реле.

Усі ці записи зберігаються у форматі, придатному для AT-команд, без використання спеціалізованих структур. Перевага такого підходу в тому, що для налаштування системи достатньо SIM-карти, яка вставляється в модем: користувач може завчасно записати потрібні параметри навіть без програмного втручання, просто використовуючи інший пристрій або термінал. Зчитування і запис забезпечуються командами AT+CPBR=n і AT+CPBW=n,"стрічка",129,"my", де "стрічка" інтерпретується як номер або як параметр відповідно до номера комірки.

2. Налаштування системи — фаза встановлення супервізора

У разі якщо після вмикання пристрою натиснута кнопка, система переходить у режим очікування дзвінка від майбутнього "супервізора" — основного користувача. Це реалізовано через функцію supervisor_wait_key(). Поки активний цей режим:

при вхідному дзвінку номер абонента зчитується,

номер записується у першу комірку телефонної книги модему (AT+CPBW=1,...),

дзвінок скидається (AT+CHUP),

система виходить з режиму навчання.

Цей механізм дозволяє прив’язати певного користувача без використання комп’ютера або додаткових інтерфейсів — достатньо натиснути кнопку й зателефонувати.

3. Основний цикл: моніторинг подій

Після початкової конфігурації система працює в нескінченному циклі. Основні події, які вона обробляє:

Події з кнопки:

- натискання у будь-який момент скасовує режим очікування супервізора;

- довге натискання перезавантажує модем (скидає прапорець `cmd_modemOk`, що викликає `modem_status()`).

Температурний контроль:

- якщо `get_temper() > 100`, реле вимикається, у телефонній книзі оновлюється режим, і надсилається SMS супервізору. Це запобігає пошкодженню навантаження або самого пристрою.

Отримання SMS чи дзвінка:

- обробка SMS відбувається у два етапи: спочатку зчитується заголовок із номером, потім — тіло повідомлення.

- при дзвінку — система зчитує номер абонента, скидає дзвінок і запускає перегляд телефонної книги (`AT+CPBR=n`) у пошуках збігу номера.

Уся робота з прийомом відповідей від модема в цій програмі виконується не в перериванні, а в основному циклі while (1). Конкретно, відповідь від модема (через UART) аналізується за допомогою функції:

```
getstr();
```

Це основна команда, яка перевіряє, чи є нові символи у вхідному буфері UART, і якщо знайдено завершений рядок (0x0A), запускає аналіз змісту через strExec().

Тобто механізм такий:

UART-переривання приймає байти і кладе їх у вхідний буфер (переривання UART працює в фоновому режимі, але там лише прийом).

У while (1) викликається getstr(), яка читає з буфера символи і, якщо сформувався повний рядок, розпізнає тип відповіді (SMS, дзвінок, USSD, +CREG тощо).

Результат getstr() — це код події: 1, 2, 3, 4, 10 і т.д., згідно з розпізнаним заголовком у strExec().

3.10.2 Реалізація аутентифікації абонентів

Модем A6 не має вбудованої аутентифікації, тому програмно виконується звірка номера. Це дозволяє визначити, чи є абонент легітимним для даної команди.

Після знайдення збігу:

встановлюється прапорець авторизації (cmd_enable),

виконується команда, залежно від типу події:

якщо SMS — розбір через execSMS(),

якщо дзвінок — дія в execCall() згідно з поточним режимом.

4. Виконання команд користувача

Формат команд у SMS простий і текстовий:

mode.r / mode.s — змінити режим роботи (скинути або перемикати реле).

msocket.on/off/t/c — керувати реле (вмикання, вимикання, стан, USSD-запит рахунку).

addN+380... — додати новий номер у телефонну книгу (тільки для супервізора).

number.clear — стерти всі записи (крім супервізора).

У разі команд типу msocket.status, msocket.accaunt — система надсилає SMS із відповідним рядком (наприклад, стан реле чи відповідь на USSD-запит).

Всі SMS передаються за допомогою команди AT+CMGS, форматуються відповідно до змісту, що зберігається в str.

3.10.35. Реалізація контролю наявносіт зв’язку в мережі GSM через USSD команди

Періодично система виконує перевірку доступності мережі, надсилаючи USSD-команду (наприклад, *100# для рахунку). Відповідь відслідковується за +CUSD:. Якщо відповідь не надходить кілька разів

поспіль (ussdrep > USSDCREP), вважається, що модем втратив підключення, і викликається повторна ініціалізація через modem_status().

Це дозволяє виявити ситуацію, коли модем ніби зареєстрований, але не має доступу до сервісів GSM (наприклад, SIM заблокована, рахунок вичерпано тощо).

У програмі контроль наявності підключення модема до мережі реалізовано через періодичні USSD-запити з використанням двох незалежних лічильників часу та лічильника кількості спроб. Така організація дозволяє не блокувати виконання основного циклу та забезпечити відновлення зв’язку в разі його втрати.

Після початкової ініціалізації модему (через modem_status()), програма переходить у вічний цикл. У циклі постійно перевіряється, чи активний прапорець cmd_modemOk, який сигналізує про те, що модем успішно пройшов ініціалізацію і приєднався до мережі. Якщо прапорець встановлений, додатково виконується блок перевірки USSD.

Цей блок базується на двох лічильниках:

ldelay1s — основний лічильник, який визначає інтервал між USSD-запитами. Після кожного успішного запиту він виставляється знову (наприклад, у DELAYREP — це може бути 60 секунд).

ldelay1d — другий допоміжний лічильник, також встановлюється після успішного запиту. Його мета — затримати наступний USSD-запит у разі відсутності відповіді, щоб уникнути занадто частих запитів.

Якщо один із цих лічильників зменшився до нуля, це означає, що наблизився час перевірки, і USSD-запит формується і надсилається:

```
strcpy(str,"AT+CUSD=1,\"");
strcat(str,str_ussd);
strcat(str,"\",15\r\n");
commExec(str,0);
```

Команда commExec(...,0) надсилає рядок модему без очікування на OK, тобто не блокує виконання циклу, а getstr() у фоні згодом має перехопити відповідь (команда +CUSD:).

У цей момент обидва лічильники перезапускаються: ldelay1s = DELAYREP, ldelay1d = DELAYREP. Якщо відповідь на +CUSD: приходить (і розпізнається у strExec() як команда 3), в case 3 у main() відбувається підтвердження зв'язку: знову встановлюються обидва лічильники, скидається лічильник помилок ussdrep, і, якщо був запит балансу, надсилається SMS.

Якщо ж відповідь не надходить, то через певну кількість таких безуспішних ітерацій (коли ldelay1d спливає, а +CUSD: не приходить), збільшується змінна ussdrep — це лічильник кількості невдалих USSD-запитів. Якщо ussdrep перевищить межу USSDCREP, система вважає, що модем втратив зв’язок.

У такому разі виводиться повідомлення через UART:

```
sendUART2("USSD ans error\r\n\0");
```

і викликається функція modem_status() — вона повністю перезапускає модем: вимикає живлення на кілька секунд (MODEMV=1, потім MODEMV=0), чекає на +CREG (реєстрацію в мережі), і заново надсилає стартові AT-команди. Якщо +CREG не дає потрібного значення (creg != 3), програма виконує аналіз коду помилки (CREG: 0, CREG: 3, CREG: 2, тощо) і викликає showError(...).

Функція showError() через світлодіоди сигналізує про тип помилки:

err_SIM — якщо SIM не читається,

err_PIN — якщо запитано PIN-код (а його не оброблено),

err_unknow — інша проблема з мережею,

err_AT — модем не відповів на AT-команди.

Світлова індикація реалізована через блимання LED2 з відповідною кількістю циклів. Після цього система знову спробує ініціалізувати модем, і цей процес буде повторюватися, доки не вдасться налагодити з’єднання або доки користувач не втрутиться.

Таким чином, схема контролю GSM мережі.

Працює циклічно, з періодичністю, визначеною таймерами.

Не блокує виконання інших дій у головному циклі.

Реагує на відсутність відповіді перезапуском модему.

Має світлову діагностику для користувача.

Забезпечує стійке функціонування навіть при часткових втратах зв’язку.

Це дозволяє забезпечити самовідновлення системи без необхідності ручного втручання.

ВИСНОВКИ

У ході бакалаврської роботи було проведено дослідження систем з дистанційним керуванням. Проаналізовано різноманітні типи дистанційного керування, розроблені з кінця 10-го століття. На основі зібраної інформації визначено найбільш ефективний метод дистанційного керування навантаженням мережі живлення, що базується на радіохвилях та технології GSM.

Спроектовано структурну схему пристрою дистанційного керування, до складу якої входять: мікроконтролер, операторська кнопка керування, два точкові індикатори, GSM модуль типу A6, силовий ключ у формі симістора, датчик температури радіатора силового ключа та додаткова постійна пам'ять.

Розроблено принципові схеми мікропроцесорного блоку з мікроконтролером STM8S003F3, комутаційними ключами живлення GSM модуля, кнопкою, точковими індикаторами та мікросхемою додаткової постійної пам'яті. Також створено схему блоку силового ключа з опторозв'язаним керуванням симістора типу MOC3063 та безпосередньо симістором BTA12. Модуль додатково оснащено напівпровідниковим температурним сенсором з від'ємною температурною характеристикою.

Створено програмне забезпечення для функціонування пристрою, яке при першому запуску дозволяє зареєструвати телефонний номер адміністратора, а надалі отримувати від нього команди для запису номерів операторів. Пристрій керує навантаженням у двох режимах (переключення та перевключення) через команди, що надходять у вигляді SMS або вхідних дзвінків. Зміна стану пристрою, його відображення, а також моніторинг балансу SIM-картки в GSM модулі здійснюються за допомогою спеціально розроблених SMS команд.

ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Кан О. А., Жаркимбекова А. Т., Кадирова Ж. Б., Жаксыбаева С. Р., Жолмагамбетова Б. Р. Спосіб передачі дискретної інформації // Сучасні наукомісткі технології. – 2015. – № 4. – С. 44–46. Режим доступу: http://www.top-technologies.ru/ru/article/view?id=35012 (дата звернення: 13.05.2025).
2. Rappaport T. S. Wireless Communications: Principles and Practice. – 2nd ed. – Upper Saddle River, NJ : Prentice Hall, 2002. – 707 p.
3. Haykin S., Moher M. Modern Wireless Communications. – Upper Saddle River, NJ : Pearson Education, 2005. – 568 p.
4. Carlson B., Crilly P., Rutledge J. Communication Systems: An Introduction to Signals and Noise in Electrical Communication. – 5th ed. – New York : McGraw-Hill, 2010. – 640 p.
5. Russell T. Wireless Security. – New York : McGraw-Hill, 2008. – 512 p.
6. Malvino A. P., Bates D. Electronic Principles. – 8th ed. – New York : McGraw-Hill Education, 2015. – 1248 p.
7. Boylestad R. L. Electronic Devices and Circuit Theory. – 11th ed. – Boston : Pearson, 2013. – 912 p.
8. Franco S. Design with Operational Amplifiers and Analog Integrated Circuits. – 4th ed. – New York : McGraw-Hill, 2014. – 704 p.
9. Ibrahim D. Microcontroller-Based Process Control. – Oxford : Newnes, 2005. – 304 p.
10. Mazidi M. A., McKinlay R. D., Mazidi J. G. The 8051 Microcontroller and Embedded Systems: Using Assembly and C. – 2nd ed. – Upper Saddle River, NJ : Pearson, 2005. – 640 p.
11. Ai Thinker Co., Ltd. A6 GSM/GPRS Module AT Command Set. – Revision 1.03. – 2016. – 82 p. Режим доступу: https://wiki.ai-thinker.com/_media/gprs/a6/a6_at_command_set_v1.03.pdf (дата звернення: 13.05.2025).
12. Ibrahim D. Designing Embedded Systems with PIC Microcontrollers: Principles and Applications. – Oxford : Newnes, 2014. – 660 p.
13. Axelson J. Serial Port Complete: COM Ports, USB Virtual COM Ports, and Ports for Embedded Systems. – 2nd ed. – Madison, WI : Lakeview Research, 2007. – 362 p.
14. Kochan S. G. Programming in C. – 4th ed. – Boston : Pearson Education, 2014. – 552 p.

ДОДАТКИ

Додаток А

Програмний модуль main.c

```
#include "main.h"
char buf[130],cbuf=0,reqOk,aUSSD[100];
char str[MAXSTRLEN];//в цю змінну влізе цифри із командою CPBR
char strparam[MAXPARAMCH],cparam=0;
char str_ussd[USSDSTRLEN];      //рядок для дозвону USSD команди
char str_mode[MODESTRLEN];      //рядок для читання режиму RS|SW


unsigned int mode_cmd_command; //команда яку треба виконати
//b7 - модем функціонує
//b6 -
char  mode_cmd_rep;         //проміжна змінна (номер перегляду записника|номер для запису
ADDxx|номер відовіді від CREG)
char mode_cmd_cont;    //признак CMC
char mode_cmd_tel[MAXPARAMCH];//символи номеру телефону
char mode_cmd_ctel;     //кількість символів у номері телефону
char mode_cmd_sms[MAXPARAMSMSCH];
char mode_cmd_csms=0;     //кількість символів у команді

char keyNew,keyOld;
int keyDelay;

#include "iostm8s003f3.h"
#include "intrinsics.h"
#include "math.h"
#include "string.h"
#include "uart.c"
#include "routins.c"



char delay1s,delay1s2=60,ussdrep,ledstatus=0;    //змінна зменшується кожу сек на 1, затримка
виключення режиму очікування супервізора
int ldelay1s=2,ldelay1d=2;
unsigned int leddelay;  //
unsigned int delay0_1s;  //зменшується кожну 0.1с на 1

void main( void )
{

  char mcmd=0,sr[4];
  perif_init();
  mode_cmd_command=0;
  sendUART2("start Ok\r\n\0");
/*
  while(1)
    {
      int2str(buf,get_temper());
      sendUART2(buf);
      putchar2(0x0d);
      del0_1s(200);
    }
*/

   modem_status();           //connect modem, send init string, get USSD answer for check
connection, get param string
  supervisor_wait_key();    //wait key for supervisor incall
  while (1){
    mcmd=getKey();
    // ? reset supervisor incall, if any push key
        if  (mcmd  &&  (mode_cmd_command  &  cmd_incallsupervisorwait))  {mode_cmd_command  &=
~cmd_incallsupervisorwait;LED2OFF;}
    //reset supervisor if time up
     if ((delay1s2==1) && (mode_cmd_command & cmd_incallsupervisorwait)) {mode_cmd_command &=
~cmd_incallsupervisorwait;LED2OFF;}
    //manual reset modem if long push
    if (mcmd==3) mode_cmd_command &= ~cmd_modemOk;

     if((mode_cmd_command & cmd_modemOk)==0) modem_status();//якщо модем не готовий, то його
запустити
    //if (get_temper()>100 && (OUTMOC==1)) {overtemp();sendUART2("OVERTEMP\r\n\0");}
    if ((get_temper()>100) && (mode_cmd_command & cmd_overtemp)==0)
```

```
      {
        sendUART2("OVERTEMP\r\n\0");
        LED1ON;
        mode_cmd_command|=cmd_overtemp;
        overtemp();
      }
    if(mode_cmd_command & cmd_modemOk)
    {
      switch (getstr())
      {
      case 0: if (leddelay==0 && (mode_cmd_command & cmd_incallsupervisorwait)==0) {
                    switch (ledstatus){
                    case 0: LED2ON;leddelay=100;ledstatus=1;break;    //засвітити перший раз
                       case 1: LED2OFF;leddelay=1000;ledstatus=2;break;     //загасити перший
раз
                    case 2: LED2ON;leddelay=100;ledstatus=3;break;    //засвітити перший раз
                       case 3: LED2OFF;leddelay=60000;ledstatus=0;break;    //загасити перший
раз
                    }
              }
       break;
       case 1:     //отримати SMS
         sendUART2("got SMS\r\n\0");
         mode_cmd_command|=cmd_SMS;    //записати команду
          cparam=dplus(cparam);//викинути плюс з початку номера
           copy(mode_cmd_tel,strparam,cparam); mode_cmd_ctel=cparam;//записати номер телефону
із параметрів у виконання
          //нічого більше не робимо, бо наступний рядок має бути текст СМС
          break;
       case 2:    //отримано вхідний дзвінок
          sendUART2("in call\r\n\0");
             if (mode_cmd_command & cmd_incallsupervisorwait) {      //якщо це запис номера
супервізора при кнопці
            cparam=dplus(cparam);//викинути плюс з початку номера
               copy(mode_cmd_tel,strparam,cparam);mode_cmd_ctel=cparam;      //записати номер
телефону із параметрів у виконання

              if (OUTMOC) del1s(15); else del1s(2);                              //дозволити
"попікати" у трубці

            commExec("AT+CHUP\r\n\0",1);        //скинути дозвон
            writenbook(1,mode_cmd_tel);
            mode_cmd_command&=cmd_clear; //скинути команду
          }
          else
          { mode_cmd_command|=cmd_call;    //записати команду
            cparam=dplus(cparam);//викинути плюс з початку номера
               copy(mode_cmd_tel,strparam,cparam);mode_cmd_ctel=cparam;      //записати номер
телефону із параметрів у виконання
            del1s(2);                        //дозволити "попікати" у трубці
            commExec("AT+CHUP\r\n\0",1);        //скинути дозвон
            commExec("AT+CPBR=1\r\n\0",0);      //запросити записник
            mode_cmd_rep=1;           //номер запису записника із якого почалося опитування
          }
          break;
       case 3://+CUSD
          sendUART2("USSD request Ok\r\n\0");
              //прийшла відповідь від мережі- значить все Ок і ставимо лічильники на одне
значення
          ldelay1s=DELAYUSSD;ldelay1d=ldelay1s;
          //if CUSD is requested - execute it
          if (mode_cmd_command & cmd_accaunt) {sendSMS();mode_cmd_command &= ~cmd_accaunt;}
          break;
       case 4://команда внутрішня(+CPBR) - читаємо номер телефона із записника
          chtostr(buf,mode_cmd_rep); sendUART2(buf); sendUART2(" ph read\r\n\0");
             if (mode_cmd_command & (cmd_call | cmd_SMS)) //якщо попередня команда була на
викоання(може бути 1 чи 2)
          {
                  if (strncmp(mode_cmd_tel,strparam,mode_cmd_ctel)==0)   //телефон записника
співпав із командою
              {
                  mode_cmd_command|= cmd_enable; //виставити признак легітимності
                   if (mode_cmd_rep==1) {mode_cmd_command|=cmd_supervisor;} //номер телефону
суперівізора
                  if (mode_cmd_command & cmd_SMS) execSMS();
                  if (mode_cmd_command & cmd_call) execCall();
                  mode_cmd_command&=cmd_clear;
                  mode_cmd_rep=0;
                  mode_cmd_ctel=0;
                  mode_cmd_csms=0;//поочистити завдання
```

```
                  break;
               }
              if (mode_cmd_rep<MAXNBOOK) {//поки не пройшли весь записник- посилаємо черговий
запит
                strcpy(str,"AT+CPBR=\0");
                chtostr(sr,++mode_cmd_rep);
                strcat(str,sr);
                strcat(str,"\r\n");
                commExec(str,0);}
             else {//пройшли весь записник-очищаємо завдання

mode_cmd_command&=cmd_clear;mode_cmd_rep=0;mode_cmd_ctel=0;mode_cmd_csms=0;//поочистити
завдання
                }    //ups.........
          }
          if (mode_cmd_command & cmd_overtemp)
          {
            //supervisor's phonenumber in strparam
            copy(mode_cmd_tel,"+",1);
            //mode_cmd_tel="+";
            strcat(mode_cmd_tel,strparam);
            sendSMS();
            mode_cmd_command &= ~cmd_overtemp;
            LED1OFF;
          }
          break;
      case 11:    //отримано наступний рядок CMC
          copy(mode_cmd_sms,strparam,cparam); //зберігаємо текст SMS
          mode_cmd_csms=cparam;    //пишемо і кількість символів у SMS
          del1s(1);
          commExec("AT+CPBR=1\r\n\0",0);    //запросити записник
          mode_cmd_rep=1;           //номер запису записника
          break;
      }//switch


//-------------------------------------------------------------------------------------
      //намагаємося визначити чи ми в мережі посилаємо запит у мережу
      if (ldelay1s<2)
       {//один із лічильників  на грані
        if (ldelay1d==0)
            {// якщо інший вже закінчився, то значит ми вже пробували запустити команду і
виставили тимчасову затримку тільи на один лічильник
          ussdrep++;      //збільшуємо кількість спроб із тимчасовим лічильником
          if (ussdrep>USSDCREP)
           {//якщо ця кількість зросла до критичної то треба якось щось робити, аби вернутися
до мережі
            sendUART2("USSD ans error\r\n\0");
            modem_status();
          }
        }//а інший закінчився-
            //запускаємо перевірку мережі і виставляємо один із лічильників. Коли прийде
легітимна відповідь від мережі, то запустяться два лічильники одночасно
        strcpy(str,"AT+CUSD=1,\"");
        strcat(str,str_ussd);strcat(str,"\",15\r\n");
        commExec(str,0);
        ldelay1s=DELAYREP;
        sendUART2("get USSD\r\n\0");
        //поблимуємо при відсиланні запиту мережі
            if  ((mode_cmd_command  &  cmd_incallsupervisorwait)==0){  LED2ON;  del0_1s(1000);
LED2OFF; }
       }
    }
    else
    {   //проблеми з модемом
        modem_status();
    }
  };

  }
```

Додаток Б

Програмний модуль routins.c

```
//керування розеткою

#include "main.h"
char buf[130],cbuf=0,reqOk,aUSSD[100];
char str[MAXSTRLEN];//в цю змінну влізе цифри із командою CPBR
char strparam[MAXPARAMCH],cparam=0;
char str_ussd[USSDSTRLEN];      //рядок для дозвону USSD команди
char str_mode[MODESTRLEN];      //рядок для читання режиму RS|SW


unsigned int mode_cmd_command; //команда яку треба виконати
//b7 - модем функціонає
//b6 -
char  mode_cmd_rep;         //проміжна змінна (номер перегляду записника|номер для запису
ADDxx|номер відовіді від CREG)
char mode_cmd_cont;    //признак CMC
char mode_cmd_tel[MAXPARAMCH];//символи номеру телефону
char mode_cmd_ctel;     //кількість символів у номері телефону
char mode_cmd_sms[MAXPARAMSMSCH];
char mode_cmd_csms=0;     //кількість символів у команді

char keyNew,keyOld;
int keyDelay;

#include "iostm8s003f3.h"
#include "intrinsics.h"
#include "math.h"
#include "string.h"
#include "uart.c"
#include "routins.c"



char delay1s,delay1s2=60,ussdrep,ledstatus=0;   //змінна зменшується кожу сек на 1, затримка
виключення режиму очікування супервізора
int ldelay1s=2,ldelay1d=2;
unsigned int leddelay;  //
unsigned int delay0_1s;  //зменшується кожну 0.1с на 1

void main( void )
{

  char mcmd=0,sr[4];
  perif_init();
  mode_cmd_command=0;
  sendUART2("start Ok\r\n\0");
/*
  while(1)
    {
      int2str(buf,get_temper());
      sendUART2(buf);
      putchar2(0x0d);
      del0_1s(200);
    }
*/

   modem_status();           //connect modem, send init string, get USSD answer for check
connection, get param string
  supervisor_wait_key();    //wait key for supervisor incall
  while (1){
    mcmd=getKey();
    // ? reset supervisor incall, if any push key
       if  (mcmd  &&  (mode_cmd_command  &  cmd_incallsupervisorwait))  {mode_cmd_command  &=
~cmd_incallsupervisorwait;LED2OFF;}
    //reset supervisor if time up
     if ((delay1s2==1) && (mode_cmd_command & cmd_incallsupervisorwait)) {mode_cmd_command &=
~cmd_incallsupervisorwait;LED2OFF;}
    //manual reset modem if long push
    if (mcmd==3) mode_cmd_command &= ~cmd_modemOk;

     if((mode_cmd_command & cmd_modemOk)==0) modem_status();//якщо модем не готовий, то його
запустити
    //if (get_temper()>100 && (OUTMOC==1)) {overtemp();sendUART2("OVERTEMP\r\n\0");}
    if ((get_temper()>100) && (mode_cmd_command & cmd_overtemp)==0)
      {
        sendUART2("OVERTEMP\r\n\0");
        LED1ON;
```

```
        mode_cmd_command|=cmd_overtemp;
        overtemp();
      }
    if(mode_cmd_command & cmd_modemOk)
    {
      switch (getstr())
      {
      case 0: if (leddelay==0 && (mode_cmd_command & cmd_incallsupervisorwait)==0) {
                    switch (ledstatus){
                    case 0: LED2ON;leddelay=100;ledstatus=1;break;    //засвітити перший раз
                       case 1: LED2OFF;leddelay=1000;ledstatus=2;break;     //загасити перший
раз
                    case 2: LED2ON;leddelay=100;ledstatus=3;break;    //засвітити перший раз
                       case 3: LED2OFF;leddelay=60000;ledstatus=0;break;    //загасити перший
раз
                    }
              }
        break;
       case 1:     //отримати SMS
         sendUART2("got SMS\r\n\0");
         mode_cmd_command|=cmd_SMS;    //записати команду
          cparam=dplus(cparam);//викинути плюс з початку номера
           copy(mode_cmd_tel,strparam,cparam); mode_cmd_ctel=cparam;//записати номер телефону
із параметрів у виконання
          //нічого більше не робимо, бо наступний рядок має бути текст СМС
          break;
       case 2:    //отримано вхідний дзвінок
          sendUART2("in call\r\n\0");
             if (mode_cmd_command & cmd_incallsupervisorwait) {      //якщо це запис номера
супервізора при кнопці
            cparam=dplus(cparam);//викинути плюс з початку номера
               copy(mode_cmd_tel,strparam,cparam);mode_cmd_ctel=cparam;      //записати номер
телефону із параметрів у виконання

                if (OUTMOC) del1s(15); else del1s(2);                          //дозволити
"попікати" у трубці

            commExec("AT+CHUP\r\n\0",1);        //скинути дозвон
            writenbook(1,mode_cmd_tel);
            mode_cmd_command&=cmd_clear; //скинути команду
          }
          else
          { mode_cmd_command|=cmd_call;    //записати команду
            cparam=dplus(cparam);//викинути плюс з початку номера
               copy(mode_cmd_tel,strparam,cparam);mode_cmd_ctel=cparam;      //записати номер
телефону із параметрів у виконання
            del1s(2);                      //дозволити "попікати" у трубці
            commExec("AT+CHUP\r\n\0",1);        //скинути дозвон
            commExec("AT+CPBR=1\r\n\0",0);      //запросити записник
            mode_cmd_rep=1;           //номер запису записника із якого почалося опитування
          }
          break;
       case 3://+CUSD
          sendUART2("USSD request Ok\r\n\0");
              //прийшла відповідь від мережі- значить все Ок і ставимо лічильники на одне
значення
          ldelay1s=DELAYUSSD;ldelay1d=ldelay1s;
          //if CUSD is requested - execute it
          if (mode_cmd_command & cmd_accaunt) {sendSMS();mode_cmd_command &= ~cmd_accaunt;}
          break;
       case 4://команда внутрішня(+CPBR) - читаємо номер телефона із записника
          chtostr(buf,mode_cmd_rep); sendUART2(buf); sendUART2(" ph read\r\n\0");
             if (mode_cmd_command & (cmd_call | cmd_SMS)) //якщо попередня команда була на
викоання(може бути 1 чи 2)
          {
                   if (strncmp(mode_cmd_tel,strparam,mode_cmd_ctel)==0)   //телефон записника
співпав із командою
               {
                    mode_cmd_command|= cmd_enable; //виставити признак легітимності
                     if (mode_cmd_rep==1) {mode_cmd_command|=cmd_supervisor;} //номер телефону
суперівізора
                    if (mode_cmd_command & cmd_SMS) execSMS();
                    if (mode_cmd_command & cmd_call) execCall();
                    mode_cmd_command&=cmd_clear;
                    mode_cmd_rep=0;
                    mode_cmd_ctel=0;
                    mode_cmd_csms=0;//поочистити завдання
                    break;
               }
```

```
              if (mode_cmd_rep<MAXNBOOK) {//поки не пройшли весь записник- посилаємо черговий
запит
                strcpy(str,"AT+CPBR=\0");
                chtostr(sr,++mode_cmd_rep);
                strcat(str,sr);
                strcat(str,"\r\n");
                commExec(str,0);}
              else {//пройшли весь записник-очищаємо завдання

mode_cmd_command&=cmd_clear;mode_cmd_rep=0;mode_cmd_ctel=0;mode_cmd_csms=0;//поочистити
завдання
                }    //ups.........
          }
          if (mode_cmd_command & cmd_overtemp)
          {
            //supervisor's phonenumber in strparam
            copy(mode_cmd_tel,"+",1);
            //mode_cmd_tel="+";
            strcat(mode_cmd_tel,strparam);
            sendSMS();
            mode_cmd_command &= ~cmd_overtemp;
            LED1OFF;
          }
          break;
      case 11:    //отримано наступний рядок CMC
          copy(mode_cmd_sms,strparam,cparam); //зберігаємо текст SMS
          mode_cmd_csms=cparam;    //пишемо і кількість символів у SMS
          del1s(1);
          commExec("AT+CPBR=1\r\n\0",0);    //запросити записник
          mode_cmd_rep=1;           //номер запису записника
          break;
      }//switch


//-----------------------------------------------------------------------------------------------
----------
      //намагаємося визначити чи ми в мережі посилаємо запит у мережу
      if (ldelay1s<2)
       {//один із лічильників  на грані
        if (ldelay1d==0)
            {// якщо інший вже закінчився, то значит ми вже пробували запустити команду і
виставили тимчасову затримку тільи на один лічильник
          ussdrep++;      //збільшуємо кількість спроб із тимчасовим лічильником
          if (ussdrep>USSDCREP)
           {//якщо ця кількість зросла до критичної то треба якось щось робити, аби вернутися
до мережі
            sendUART2("USSD ans error\r\n\0");
            modem_status();
          }
        }//а інший закінчився-
            //запускаємо перевірку мережі і виставляємо один із лічильників. Коли прийде
легітимна відповідь від мережі, то запустяться два лічильники одночасно
        strcpy(str,"AT+CUSD=1,\"");
        strcat(str,str_ussd);strcat(str,"\",15\r\n");
        commExec(str,0);
        ldelay1s=DELAYREP;
        sendUART2("get USSD\r\n\0");
        //поблимуємо при відсиланні запиту мережі
            if  ((mode_cmd_command  &  cmd_incallsupervisorwait)==0){  LED2ON;  del0_1s(1000);
LED2OFF; }
       }
    }
    else
    {   //проблеми з модемом
        modem_status();
    }
  };

  }
```

Додаток В

Програмний модуль main.h

```
//керування розеткою
//port A1 вихід керування симістором (для вкл=1)
//port A2 вхід конденсатор для визначення збою
//port B4 вихід світлодіода LED1        (для вкл=0)
//port B5 вихід управління RESET модема (для RESET=0)
//port C3 вихід світлодіода LED2        (для вкл=0)
//port C4 вихід управління живленням модема (для вкл=1)
//port D4 вхід кнопки (при натиск=0)
//записник
//номер 1- номер запиту рахунку
//номер 2- номер супервізора

//#define LED1ON PB_ODR_bit.ODR4=0
//#define LED1OFF PB_ODR_bit.ODR4=1
//#define LED1 PB_ODR_bit.ODR4
//#define LED2ON PC_ODR_bit.ODR3=0
//#define LED2OFF PC_ODR_bit.ODR3=1
#define LED1ON PC_ODR_bit.ODR3=0
#define LED1OFF PC_ODR_bit.ODR3=1
#define LED1 PC_ODR_bit.ODR3
#define LED2ON PC_ODR_bit.ODR5=0
#define LED2OFF PC_ODR_bit.ODR5=1


#define OUTMOC PA_ODR_bit.ODR1  //вихідний сигнал на MOC
#define MODEMV PC_ODR_bit.ODR4  //управління живленням модема
#define MODEMRes PB_ODR_bit.ODR5
#define KEY1 PD_IDR_bit.IDR4     //кнопка
#define CAP PA_IDR_bit.IDR2     //вхід сигналу конденсатора

#define MAXSTRLEN 50    //кількість символів у проміжній рядковій змінній
#define MAXPARAMCH 50   //кількість символів параметру команди
#define MAXPARAMSMSCH 30   //кількість символів параметру команди
#define DELAYUSSD 6000  //затримка між запитами активності мережі за допомогою USSD команд
#define DELAYREP 10     //затримка повторного запиту USSD пісял помилки
#define USSDCREP 5      //кількість невдалих запитів до визнання проблем із звязком
#define OUTOFFDELAY 3 //затримка при виключенні розетки

#define RECUSSD 25      //номер запису в якому буде знаходитися номер USSD
#define RECMODE 24      //номер запису в якому буде знаходитися стан
#define MAXNBOOK 9     //максимальна кількість номерів у зап книжці

#define USSDSTRLEN 6      //кількість символів у рядку USSD (при включенні вичитується із
simcard)
#define MODESTRLEN 3     //кількість символів у рядку режимів (при включенні вичитується із
simcard)

#define cmd_call 0x0002   //прийшов виклик по дозвону
#define cmd_SMS  0x0001   //прийшло СМС
#define cmd_USSDSMS 0x0004        //запит на рах унок
#define cmd_enable 0x0010         //дозвіл на роботу по команді
#define cmd_supervisor 0x0020     //признак супервізора
#define cmd_incallsupervisorwait 0x0040           //признак очікування вхідного дзвінка для
запису його в супервізори
#define cmd_modemOk 0x0800          //признак очікування вхідного дзвінка для запису його в
супервізори
#define cmd_clear 0xf800        //маска зі всіма командами окрім modemOk
#define cmd_accaunt 0x0100       //потреба в відсиланні рахунку
#define cmd_clincall 0x0200      //потреба в скиданні запису телефону
#define cmd_overtemp 0x0400     //overtemp symistor



#define err_PIN 2
#define err_SIM 3
#define err_AT 1
#define err_unknow 4

#define TX2bit PA_ODR_bit.ODR3
```

Додаток Г

програмний модуль UART.c

```
#define UPE 0
#define DOR 3
#define FE 1
#define UDRE 7
#define RXC 5
#define FRAMING_ERROR (1<<FE)
#define PARITY_ERROR (1<<UPE)
#define DATA_OVERRUN (1<<DOR)
#define DATA_REGISTER_EMPTY (1<<UDRE)
#define RX_COMPLETE (1<<RXC)

#define RX_BUFFER_SIZE 64
char rx_buffer[RX_BUFFER_SIZE];
unsigned char rx_wr_index,rx_rd_index,rx_counter;

// This flag is set on USART Receiver buffer overflow
char rx_buffer_overflow;

// -------------------------USART Receiver interrupt service routine
#pragma vector=UART1_R_RXNE_vector //18       //номер вектора прийнятого символа із опису на
контролер
__interrupt void USART1_RXE(void)
 {
    char status,data;
    status=UART1_SR;
    data=UART1_DR;
    if ((status & (FRAMING_ERROR | PARITY_ERROR | DATA_OVERRUN))==0)
    {
    rx_buffer[rx_wr_index]=data;
    if (++rx_wr_index == RX_BUFFER_SIZE) rx_wr_index=0;
    if (++rx_counter == RX_BUFFER_SIZE)
          {
          rx_counter=0;
          rx_buffer_overflow=1;
          };
    };
    }
char getchar(void)
    {
    char data;
    while (rx_counter==0);
    data=rx_buffer[rx_rd_index];
    if (++rx_rd_index == RX_BUFFER_SIZE) rx_rd_index=0;
    __disable_interrupt();
    --rx_counter;
    __enable_interrupt();
    return data;
    }

// -------------------------------USART Transmitter buffer
#define TX_BUFFER_SIZE 8
char tx_buffer[TX_BUFFER_SIZE];
unsigned char tx_wr_index,tx_rd_index,tx_counter;

// USART Transmitter interrupt service routine
#pragma vector=UART1_T_TXE_vector // Прерывание по освобождению буфера.
__interrupt void UART1_TXE(void)
   {
    UART1_SR_TC=0;
     if (tx_counter)
    {
      --tx_counter;
      UART1_DR=tx_buffer[tx_rd_index];
      if (++tx_rd_index == TX_BUFFER_SIZE) tx_rd_index=0;
    };
   }

void putchar(char c)
    {
    while (tx_counter == TX_BUFFER_SIZE);
    __disable_interrupt();
    if (tx_counter || ((UART1_SR & DATA_REGISTER_EMPTY)==0))
    {
      tx_buffer[tx_wr_index]=c;
      if (++tx_wr_index == TX_BUFFER_SIZE) tx_wr_index=0;
      ++tx_counter;
    }
```

```
else
  UART1_DR=c;
__enable_interrupt();
}
```