Форма № 24

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»

Інститут інформатики та радіоелектроніки,
Факультет радіоелектроніки та телекомунікацій
(повне найменування інституту, факультету)

Кафедра інформаційних технологій електронних засобів
(повне найменування кафедри)

Пояснювальна записка

до дипломного проєкту (роботи)
бакалавр
(ступінь вищої освіти)

на тему Модуль контролю затоплення підвальних приміщень з автоматичним перекриттям електроклапанів подачі води

Виконав: студент(ка) 4 курсу, групи РТ-518сп

Спеціальності 172 «Телекомунікації та радіотехніка»
(код і найменування спеціальності)

Освітня програма (спеціалізація)
«Інтелектуальні технології мікросистемної радіоелектронної техніки»

Костін Д.М.
(прізвище та ініціали)

Керівник Малий О.Ю.
(прізвище та ініціали)

Рецензент Мороз Г. В.
(прізвище та ініціали)

2021

Форма № 25

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»
(повне найменування закладу вищої освіти)

Інститут, факультет Інститут інформатики та радіоелектроніки, Факультет радіоелектроніки та телекомунікацій
Кафедра Інформаційних технологій електронних засобів
Ступінь вищої освіти бакалавр
Спеціальність 172 «Телекомунікації та радіотехніка»
(код і найменування)
Освітня програма (спеціалізація) Інтелектуальні технології мікросистемної радіоелектронної техніки
(назва освітньої програми (спеціалізації)

ЗАТВЕРДЖУЮ
Завідувач кафедри УТЕЗ
к.т.н. Огренич С.В.
«27» 05 2021 року

З А В Д А Н Н Я
НА ДИПЛОМНИЙ ПРОЄКТ (РОБОТУ) СТУДЕНТА(КИ)

Костін Данило Максимович
(прізвище, ім'я, по батькові)

1. Тема проєкту (роботи) Модуль контролю затоплення підвальних приміщень з автоматичним перекриттям електроклапанів подачі води
керівник проєкту (роботи) Малий Олександр Юрійович, к.т.н.,
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)
затверджені наказом закладу вищої освіти від «26» квітня 2021 року №163
2. Строк подання студентом проєкту (роботи) 7 червня 2021 року
3. Вихідні дані до проєкту (роботи) системи диспетчирізації ЖКГ, датчики затоплення
4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити) Огляд області розробки, постановка задач кваліфікаційної роботи, розробка структури модулю, вибір елементів, розробка конструкції модулю, розробка програмного забезпечення
5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень)
16 рисунків, 10 таблиць, 14 слайдів

6. Консультанти розділів проекту (роботи)

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	прийняв виконане завдання
1 - 3	Малий О.Ю., доцент		
Нормоконтроль	Поспєєва І.Є., ст. викладач		

7. Дата видачі завдання «26» квітня 2021 року.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломного проекту (роботи)	Строк виконання етапів проекту (роботи)	Примітка
1	Огляд систем автоматизації та диспетчирізації в ЖКГ	26.04.21	
2	Постановка технічного завдання	28.04.21	
3	Розробка структурної схеми та вибір елементів	29.04.21	
4	Розробка конструкції	03.05.21	
5	Написання тексту програмного забезпечення	29.05.21	
6	Оформлення ПЗ та захист дипломного проекту	01.06.21	

Студент(ка) _____ Костін Д.М.. _____
(підпис) (прізвище та ініціали)

Керівник проєкту (роботи) _____ Малий О.Ю.. _____
(підпис) (прізвище та ініціали)

РЕФЕРАТ

ПЗ: 43 с., 10 табл., 16 рис., 13 джерел, 14слайдів.

Об'єкт розробки: системи контролю та обліку в сфері ЖКГ.

Предмет розробки: електронний модуль з програмним забезпеченням для контролю аварійних ситуацій при протіканні комунікацій по подаванню води у багатоповерхових житлових будинках з можливістю автоматичного перекривання клапанів подачі води.

Мета роботи: вибір компонентів, розробка структури та конструкції модуля контролю затоплення підвальних приміщень з програмним забезпеченням для віддаленої диспетчирізації.

Для здійснення поставленої мети необхідно було вирішити наступні задачі:

- ознайомитись з особливостями організації систем контролю доступу в ЖКГ;
- провести розробку структури модульної системи контролю затоплення підвальних приміщень;
- розробити конструкцію модулю контролю;
- розробити базу даних для зберігання даних, щодо розташування датчиків протікання та електроклапанів перекривання подачі води;
- розробити протокол обміну модуля контролю з віддаленим сервером;
- розробити програму керуючого мікроконтролеру модуля контролю.

МОДУЛЬ, ПЛАТА, ДАТЧИК ЗАТОПЛЕННЯ, ЕЛЕКТРОКЛАПАН КОНСТРУКЦІЯ, ПРОГРАМА, СЕРВЕР, БАЗА ДАНИХ, ОСББ, ПРОГРАМУВАННЯ, СИСТЕМА КОНТРОЛЮ ЗАТОПЛЕННЯ.

ЗМІСТ

ВСТУП

Одним з найбільших споживачів енергоресурсів є житлово-комунальне господарство міста. В даний час діяльність житлово-комунального господарства супроводжується вельми великими втратами енергоресурсів, як самими комунальними підприємствами, так і іншими споживачами. Важливим елементом житлово-комунальної реформи є енергозбереження, яке може реально зменшити асигнування бюджету міста в ЖКГ і знизити динаміку збільшення витрат населення на оплату споживання енергоресурсів, при одночасному поліпшенні якості комунального обслуговування.

Диспетчеризація та автоматизація в сфері ЖКГ збільшує ефективність використання енергоресурсів. Скорочує неконтрольований витрата води, газу, тепла та електроенергії. Значно знижує сукупні витрати на обслуговування інженерних систем і комунальної інфраструктури об'єктів.

Послуги населенню повинні надаватися найбільш ефективно, а собівартість при цьому мати низьку.

Адже вся неефективність роботи інженерних систем в кінцевому підсумку відображається на споживачах, які в цьому випадку оплачують завищені тарифи і перевитрати.

Диспетчеризація це інформаційна система, яка розширює функціонал засобів автоматизації. Але з іншого боку якщо автоматизувати цілу мережу об'єктів в рамках однієї системи, то єдиний диспетчерський пункт для збору інформації про підключені об'єктах, управління і контролю аварійних ситуацій, буде необхідний. І саме в цьому випадку система диспетчеризації не є доповненням до засобів автоматизації, а стає невід'ємною частиною всього комплексу автоматизації.

На теперішній час нажаль велика кількість багатоповерхових житлових будинків мають у своєму складі комунікації водопостачання та водовідведення, що потребують оновлення та ремонту. Тому часто

виникають аварійні ситуації при яких відбувається протікання труб. Коли протікання відбувається в межах квартир – жителі досить швидко виявляють цю проблему та приймають заходи щодо припинення аварійної ситуації шляхом ремонту труб або їх повної заміни на нові. Однак коли протікання відбувається в підвальних приміщеннях, за рахунок обмеженого доступу до цих приміщень, аварійна ситуація виявляється не одразу. Це призводить до затоплення підвальних приміщень, що сильно впливає на конструкцію будівлі багатоповерхового будинку призводячи до поступового руйнування фундаменту з просадкою будівлі, що в свою чергу може призвести до віднесення будинку до аварійного з виселенням жильців.

В кваліфікаційній роботі пропонується розробка модуля контролю затоплення підвальних приміщень з можливістю автоматично перекриття водопостачання у точках прориву. Додатково пропонується зв’язувати модуль з віддаленим сервером керуючої компанії для швидкого висвітлення аварійної ситуації з метою виїзду ремонтної бригади у будинок для відновлення водопостачання.

Об'єкт розробки: системи контролю та обліку в сфері ЖКГ.

Предмет розробки: електронний модуль з програмним забезпеченням для контролю аварійних ситуацій при протіканні комунікацій по подаванню води у багатоповерхових житлових будинках з можливістю автоматичного перекривання клапанів подачі води.

Мета роботи: вибір компонентів, розробка структури та конструкції модуля контролю затоплення підвальних приміщень з програмним забезпеченням для віддаленої диспетчирізації.

1 ОГЛЯД ОБЛАСТІ РОЗРОБКИ ТА ПОСТАНОВКА ЗАДАЧ

1.1 Автоматизація в ЖКГ

ЖКГ - житлово-комунальне господарство - широка сфера для впровадження систем автоматизації та диспетчеризації. Сьогодні існує гостра необхідність у суворій диспетчеризації вузлів обліку споживання ресурсів в багатоквартирних будинках і приватних домоволодіннях. Причому в обліку зацікавлені обидві сторони - як постачають організації, так і споживачі. Ресурсопостачальні організації хочуть отримувати оплату в повному обсязі за поставлені ресурси, а споживачі хочуть оплачувати тільки за те, що спожили за фактом і не платити за боржників. Часто практика «розкидання» боргів лягає важким тягарем на сумлінних платників і КК.

На допомогу в таких ситуаціях приходить прозора і чесна диспетчеризація систем водопостачання та електропостачання з доступом до неї двох сторін - постачає організації та споживачів. Це виключає ситуацію, коли по приладу обліку платника одні свідчення, а в рахунку від постачає організації зовсім інші цифри, що відрізняються в більшу сторону.

Основні напрямки розвитку енергозбереження в місті в області автоматизації.

Економія витрачання ресурсів і зниження тепловтрат:

- установка станцій групового управління насосами ЦТП з приладами обліку та регулювання теплової та електричної енергії, води. Створення системи управління та збору інформації;
- установка систем регулювання потужності електродвигунів вентиляторів котлів котелень. Оптимізація процесів горіння на котлах і впровадження оптимальних автоматизованих графіків регулювання;
- автоматизація процесів водопідготовки та транспорту води.

Облік і регулювання споживання енергоресурсів і води:

- установка групових приладів обліку енергоресурсів;

- створення автоматизованої системи регулювання та збору інформації;

- установка інтелектуальних багатофункціональних приладів обліку електроенергії.

Створення автоматизованої системи оперативно-диспетчерського контролю та обліку споживання енергоресурсів та води.

Сукупність усіх систем обліку споживання енергоресурсів, поєднана в єдиний обліковий комплекс комунальних платежів і становить Єдину автоматизовану систему обліку споживання енергоресурсів по ЖКГ міста. Переваги, які дає автоматизована система обліку споживання енергоресурсів:

- можливість миттєвого контролю і обліку за витратами енергоресурсів на вироблення теплової енергії, води;

- порівняння балансу виробленої і спожитої енергії, визначення та облік технологічних втрат;

- проведення автоматизованого розрахунку між енергопостачальними та житловими організаціями;

- подальший вихід на банківську систему розрахунку з побутовими споживачами;

- контроль роботи енергопостачальних підприємств з єдиного центру;

- оптимальний розрахунок вартості вироблюваних енергоресурсів;

- перехід на більш високо інтелектуальний рівень організації виробництва;

- об'єднання локальних облікових систем підприємств житлово-комунального комплексу в єдине ціле.

Стандартна система диспетчеризації ЖКГ складається з наступних елементів:

- оператор на диспетчерському пункті;

- об'єкти управління;

- комплекс засобів автоматизації;

- зв'язку і управління та їх об'єднує (сервер, комп'ютери, засоби зв'язку)

Принцип роботи системи диспетчеризації, можна розбити на наступні рівні. Нижній рівень. Це вимірювальна і контрольна апаратура, автоматичного управління. На цьому рівні відбувається вибір лічильників, ПЛК, визначається по яким протоколам, пристрої будуть передавати інформацію. Наступний рівень це комунікації. На цьому рівні відбувається вибір засобів і технологій з організації основних і дублюючих каналів зв'язку. Тут же визначаються пристрою узгодження (шлюзи) протоколів і інтерфейсів.

Рівень диспетчеризації. Це вибір програмного забезпечення, яке буде відповідати за збір і обробку даних. А так же рівень інтеграції, тобто забезпечення зв'язку з іншими підсистемами і системами верхнього рівня.

Під програмним забезпеченням комплексу маються на увазі SCADA-системи. Програмне забезпечення в системі диспетчеризації ЖКГ, має бути вільно програмованим. Так як це дає можливість автоматизувати об'єкти різних типів на базі обладнання одного виробника. Це дозволяє уникнути в подальшому проблем з підключенням різнобічного обладнання до єдиної системи диспетчеризації.

Програмне забезпечення обов'язково повинно здійснювати роботу в безперервному режимі, для того, щоб оператор зміг забезпечити:

- безперебійну цілодобову роботу системи сім днів на тиждень;
- швидке (не більше однієї години) відновлення працездатності системи і її частин в разі відмови їх роботи;
- постійне проведення моніторингу подій і поточного стану системи і її частин, що дозволяє безперервно відстежувати доступність програмно-апаратного комплексу системи і поточний стан використання обладнання;
- негайно формувати повідомлення оператору системи про відмову роботи системи і її частин;
- контроль і аналіз поточної продуктивності та інших параметрів роботи системи і її частин, своєчасне виявлення загроз.

Функціональні можливості автоматизованої системи диспетчерського управління. Функції контролю. Управління системою здійснюється за допомогою певної програми, встановленої на комп'ютері диспетчера. Вона із заданою періодичністю проводить опитування всіх контролерів, які встановлені на об'єктах житлово-комунального господарства. Ті, в свою чергу, опитують прилади обліку і датчики стану об'єкта, аналізують прихожу інформацію і перетворять отримані сигнали в фізичні величини (миттєві показники енергоспоживання, параметри стану об'єкта), контролюючи задані граничні значення параметрів. Оператор диспетчерського пункту має можливість вивести на екран монітора все характеристики контрольованого об'єкта у вигляді мнемосхем, таблиць, діаграм і графіків показників витрат енергоносіїв, з можливістю виведення інформації на друк.

Функції управління реалізуються на об'єктах по командам управління, що подається з комп'ютера диспетчера на виконавчі пристрої: насоси, вимикачі, регулятори і ін. Інформація, яка надходить на пульт в диспетчерську, повертається на об'єкт у вигляді керуючої команди:

- перевести об'єкт в той чи інший режим;
- змінити параметри;
- зупинити роботу (до приїзду аварійної служби).

Переваги впровадження системи диспетчеризації та автоматизації в сферу ЖКГ цілком очевидні.

По-перше, всі підключені інженерні системи контролюються централізовано.

По-друге, контроль аварійних ситуацій та оперативне реагування, а значить своєчасне усунення аварій.

По-третє, диспетчеризація, зменшує вплив людського фактора і знижує витрати на експлуатацію.

По-четверте, система автоматичного формування звітів, оптимізація документообігів.

Звичайно ж, найголовніше диспетчеризація і автоматизація послуг ЖКГ забезпечує економію енергоресурсів і надає максимальний комфорт для споживачів.

80% керівників компаній відчувають такі проблеми, як:

- високий коефіцієнт втрат по воді, його оплачують мешканці або керуюча компанія;
- немає можливості зібрати показання лічильників води – мешканці не здають, а в квартири не пускають;
- недобросовісні мешканці ставлять магніти на лічильник або занижують показання;
- споживання води для поливу, прибирання і в загальних місцях користування;
- бухгалтерія витрачає кілька днів на місяць на збір, обробку показань і виставлення рахунків;
- «Водоканал» завищує обсяг поставленої води, а аргументів, щоб оскаржити їх цифри недостатньо;
- абоненти не оплачують рахунки за спожиті ресурси або відмовляються зовсім;
- потрібні штатні обхідники для збору актуальних показників лічильників електроенергії.

Технічні рішення з використанням автоматизації дозволять забезпечити контроль і управління за наступним обладнанням, що застосовується в даній сфері:

- котельні установки і індивідуальні теплові пункти;
- устаткування каналізаційно-насосних станцій;
- системи водопостачання, в тому числі і оснащення протипожежного призначення;
- електричні мережі і системи енергопостачання будь-яких об'єктів;
- управління освітленням прибудинкових територій, вулиць;
- енергетичні вузли, включаючи трансформаторні підстанції;

- ліфтове господарство і системи вентиляції та димовидалення.

Завдяки рішенням в сфері диспетчеризації можна забезпечити автоматичний контроль показань будь-яких приладів обліку, що застосовуються в ЖКГ.

Слід відзначити той факт, що всі основні розробки спрямовані саме на спрощення процесу контролю. Результатом стали системи, управляти роботою яких може оператор без спеціальних навичок. Цілком достатньо вміти працювати за персональним комп'ютером зі стандартною операційною системою Windows.

Диспетчеризація об'єктів ЖКГ із застосуванням сучасних систем зв'язку, контролерів і відповідного програмного забезпечення відрізняється наступними перевагами:

- висока швидкість обробки даних;
- можливість моніторингу та управління необмеженим числом об'єктів різного призначення;
- висока точність контролю різних параметрів роботи обладнання і систем;
- можливість оперативного управління технологічними процесами і устаткуванням при виникненні нештатних або аварійних ситуацій;
- невисока вартість впровадження оснащення для диспетчеризації житлово-комунального господарства.

Поточна кваліфікаційна робота покликана вирішити питання з диспетчеризацією стону систем водопостачання та водовідведення, що спростить процес обслуговання житлових будинків.

1.2 Огляд аналогів

Для постановки задач кваліфікаційної роботи було зроблено аналіз систем керування в сфері ЖКГ та окремо пристроїв контролю затоплення.

Програмний пакет «Розумне ЖКГ».

Розумне ЖКГ - цифрова платформа для забудовників, керівників підприємств, жителів і постачальників послуг. Технологічна основа для створення єдиного стандарту сервісу, зростання продажів і рентабельності управління житловим фондом.

Програмний пакет HubEx FSM

Гнучкий CRM + Service desk для виїзних співробітників: поєднує облік заявок на виїзди, GPS-контроль переміщень, базу-даних клієнтів і звіти про виконану роботу. Єдиний інструмент для клієнтського сервісу в компаніях з виїзним персоналом.

Програмний пакет «Домовласник» Автоматизація роботи ЖКГ

Програмний пакет «Дом.Контроль».

Дом.Контроль - сервіс для управління багатоквартирними будинками і котеджними селищами. Модульна система дозволяє автоматизувати процеси експлуатації та сервісу, а також підвищити фінансову дисципліну абонентів.

Не дивлячись на досить широкий спектр пропозицій по диспетчеризації та автоматизації – представлені системи не приділяють увагу датчикам затоплення, а лише впроваджують віддалені системи обліку води. Затоплення підвалів у свою чергу призводить до просадки житлових будівель з подальшим ризиком виникнення аварійних ситуацій. Введення системи віддаленого контролю затоплення дозволить автоматизовано, а бо оперативно в ручному режимі виявляти та запобігати затопленню підвальних приміщень та запобігати руйнуванню житлових будівель.

Додатково на ринку існують окремі технічні рішення для контролю затоплення.

Найпростішим відносно автономним рішенням є бездротовий датчик затоплення фірми Сейба.

Рисунок 1.1 – Бездротовий датчик затоплення фірми Сейба (Око ТМ)

Wi-Fi датчик затоплення (потопу) - це бездротове портативний пристрій для моніторингу протікання води на віддалених об'єктах, таких як квартира, будинок, офіс, магазин, склад і т.п.

Основне завдання Wi-Fi датчика потопу полягає в тому, щоб він через Wi-Fi мережу і інтернет сповіщав господаря об'єкта про витік води в місцях з потенційним ризиком затоплення на смартфон за допомогою Push-повідомлень.

Принцип роботи Wi-Fi датчика протікання в тому, що при попаданні води на струмопровідні контакти датчика бездротовий сигнал через Wi-Fi мережу і інтернет надходить на смартфон господаря за допомогою Push-повідомлень. При цьому на телефоні господаря має бути встановлено мобільний додаток OKO-WiFi.

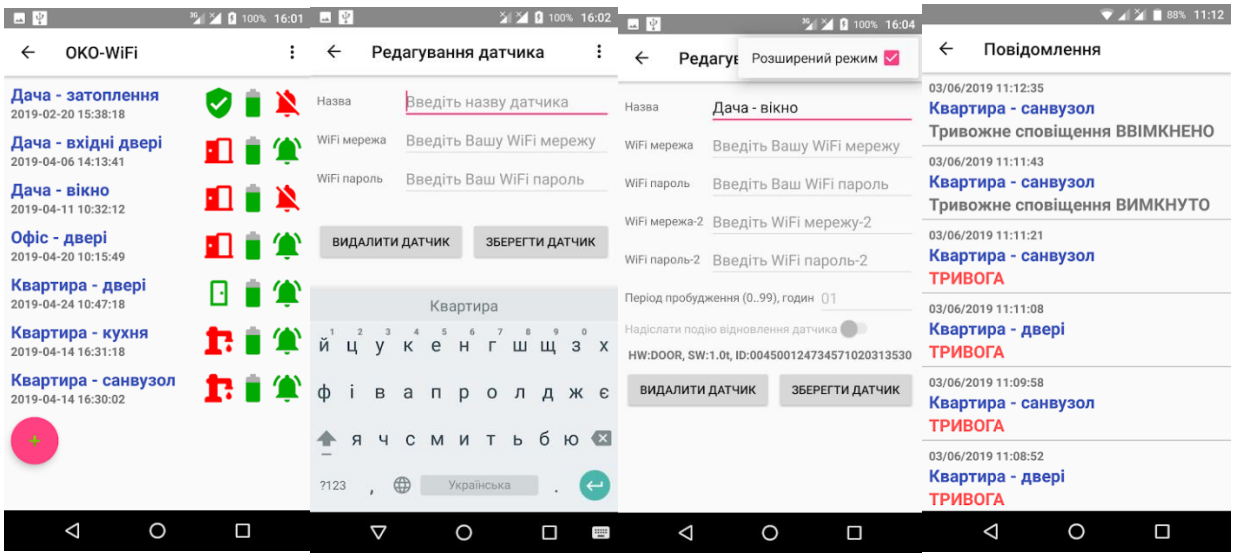

Рисунок 1.2 – Зовнішній вигляд додатку повідомлення про затоплення Oko-WiFi

Рисунок 1.3 – Комплект контролю затоплення для квартири або приватного будинку GIDROLOCK ULTIMATE BONOMI

Призначений для установки в квартирах з централізованим гарячим і холодним водопостачанням. Комплектується кульовими електроприводами GIDROLOCK UTIMATE BONOMI розмір 3/4 дюйма.

Система захисту від протікання води GIDROLOCK в автоматичному режимі (без участі людини) перекриває подачу води у разі витоку, а система безпеки AJAX сповістить користувача про проблему. Крім цього є автоматичне закриття / відкриття подачі води за сценарієм AJAX. Наприклад при постановці / знятті охорони, перехід в нічний режим або в ручному режимі по команді зі смартфона.

Відмінною особливістю системи є кульові крани з електроприводом з ресурсом 250 000 циклів "відкриття / закриття". Вбудований акумулятор 12V, 1,3 А / ч дозволяє системі тривалий час працювати без живлення 220 вольт. При включеній напруги живлення акумуляторна батарея знаходиться в режимі підзарядки.

У комплект входить базовий комплект бездротової системи безпеки Ajax з можливістю розширення захисних функцій за рахунок додавання додаткових пристроїв.

Основні відмінності комплекту:

- робота по Wi-Fi і Ethernet;

- підтримка двох SIM-карт.

Система управляється дистанційно за допомогою мобільних додатків на iOS і Android.

Характеристики:

- напруга живлення-220В ± 15%, 50Гц;

- резервне живлення-акумуляторна батарея 12В, 1.3 А * год;

- максимальна температура рідини-+120 градусів;

- діаметр підключення, "3/4;

Недоліками розглянутих аналогів програмного забезпечення є відсутність роботи з апаратним забезпеченням з постачання води, а недоліками розглянутих апаратних систем є їх націленість на окремі квартири або приватні будинки та на підходить для керуючих компаній та ОСББ.

В кваліфікаційній роботі пропонується об’єднати позитивні сторони перших та других систем для ЖКГ в систему, що дозволить проводити моніторинг стану систем водопостачання з можливістю автоматичного перекриття при аварійних ситуаціях до виїзду ремонтної бригади.

1.3 Постановка задач

Враховуючи актуальність теми та провівши аналіз аналогів систем затоплення в кваліфікаційній роботі було прийнято рішення розробити власний модуль контролю затоплення.

Модуль повинен мати наступні функції:

- мати можливість підключення великої кількості датчиків затоплення;
- мати можливість підключення великої кількості електроклапанів для керування подавання води у будинок;
- забезпечувати мінімальну кількість дротових з’єднань;
- мати можливість віддаленого моніторингу стану датчиків;
- мати можливість автоматичного та віддаленого керування подачею води;
- мати динамічну структуру, що може бути застосована для різних конфігурацій приміщень;
- мати віддалений сервер для зберігання даних щодо плану будівлі, розташування датчиків та виконавчих механізмів з описом їх призначення.

Модуль має працювати у трьох режимах:

- лише оповіщення (коли при виникненні затоплення відправляється лише повідомлення з вказанням зони затоплення для подальшого прийняття рішення з боку керуючою компанії);
- оповіщення з можливістю зміни стану електроклапанів при підтвердженні з боку диспетчера;
- оповіщення з автоматичним перекриванням відповідних клапанів.

Функції управління реалізуються на об'єктах по командам управління, що подається з комп'ютера диспетчера на виконавчі пристрої: насоси, вимикачі, регулятори і ін. Інформація, яка надходить на пульт в диспетчерську, повертається на об'єкт у вигляді керуючої команди:

- перевести об'єкт в той чи інший режим;
- змінити параметри;
- зупинити роботу (до приїзду аварійної служби).

Таблиця 1.1 – Технічні параметри модулю, що проектується

Характеристика	Значення
Напруга живлення, В	$12^{\pm}10\%$
Споживана потужність, що , Вт	Не більше 15
Кількість входів для підключення датчиків затоплення, шт.	1
Кількість датчиків затоплення, що можна приєднати, шт.	256
Кількість виходів підключення виконуючих пристроїв, що можна приєднати, шт.	256

Модуль контролю призначений для використання в наступних умовах навколишнього середовища: температура повітря, що оточує корпус приладу +1 .- .. + 35 ° С; відносна вологість повітря (при температурі + 25 ° С) не більше 80%.

2 РОЗРОБКА СТРУКТУРИ ТА КОНСТРУКЦІЇ МОДУЛЮ

2.1 Розробка структурної схеми

Перед проектуванням конструкції та програмного забезпечення модулю контролю затоплення було розроблено структуру системи контролю затоплення з можливістю автоматичного керування електроклапанами подачі води (рис.2.1).

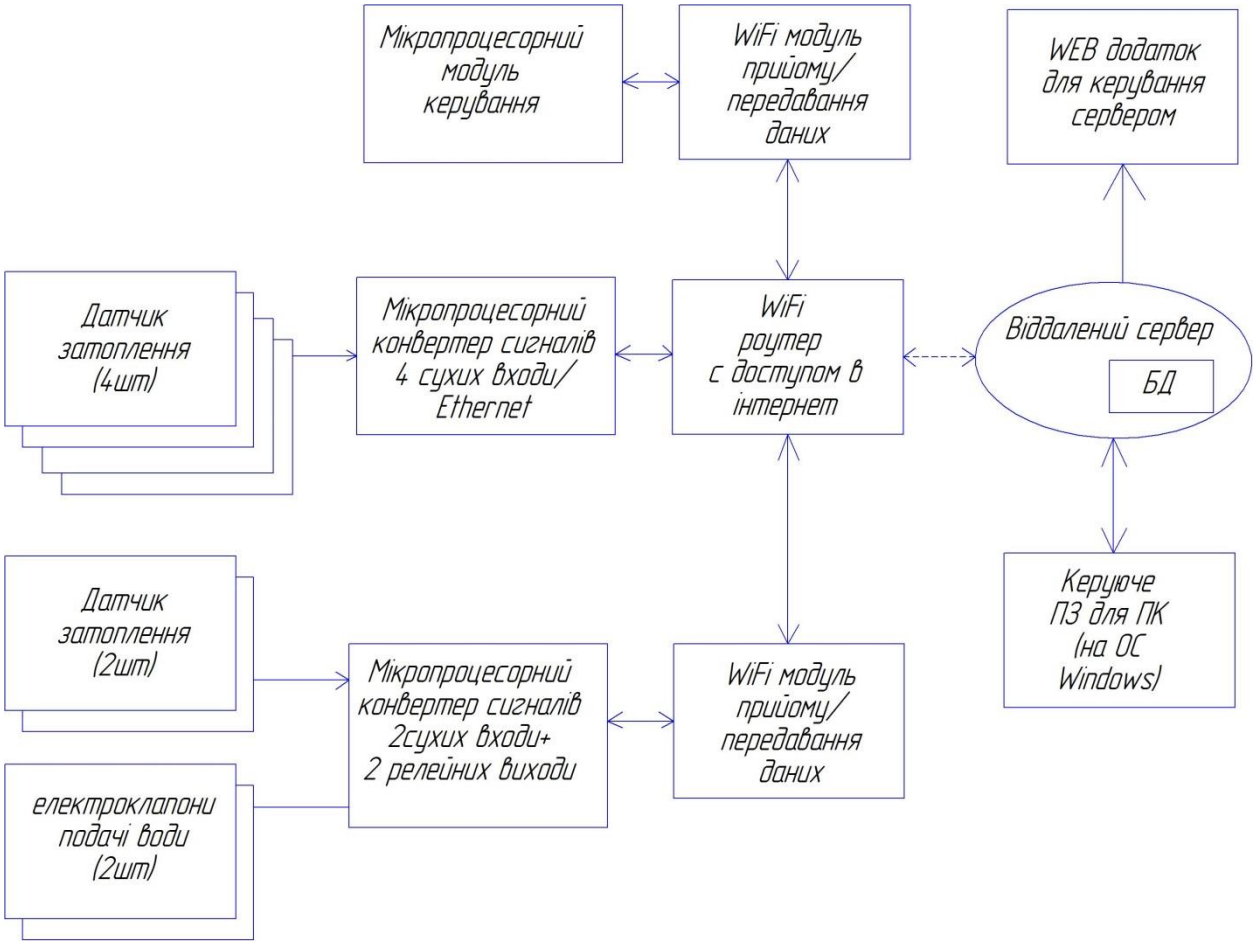

Рисунок 2.1 – Структурна схема системи контролю затоплення підвальних приміщень

Основним модулем, що підлягає розробці в кваліфікаційній роботі є мікропроцесорний модуль керування. Він збирає данні з інших стандартизованих покупних мікропроцесорних модулів та за допомогою мережі WIFI з можливістю виходу в інтернет відправляє данні на віддалений

сервер. Також від віддаленого сервера у разу необхідності він отримує команди на вимикання електроклапанів подачі води у випадку роботи в режимі віддаленого керування клапанами.

Систему можна умовно розділити на чотири блоки. Блок керування, що є основною метою розробки в кваліфікаційній роботі, типові блок збирання даних з датчиків затоплення, типові блоки збирання даних з можливістю керування електроклапанами, блок віддаленого сервера, зберігання та відображення даних.

Блок керування складається з плати керування на базі мікроконтролера та модуля обміну даними засобом бездротового зв’язку WiFi.

Типові блоки збирання даних з датчиків затоплення передбачені лише для збирання даних про стан датчиків затоплення та передачі їх стану до блоку керування без можливості впливати на процес. До такого блока можна приєднати 4 датчика затоплення. Кількість таких блоків визначається в залежності від структури будівлі. Чим більше ймовірних точок протікання при одній лінії подачі води – тим більше таких блоків.

Типові блоки збирання даних з можливістю керування електроклапанами можуть не лише збирати данні про стан датчиків затоплення а й керувати електроклапанами за наказом блоку керування. Однак до них можуть приєднуватися лише 2 датчики затоплення, але ще 2 виконуючих механізми якими як правило будуть електроклапана керування подачею води (можливе приєднання будь-якого виконуючого пристрою – наприклад звукового оповіщувателя). Кількість таких блоків також визначається структурою будівлі – чим більше точок подачі воді чи необхідність в додаткових виконуючих механізмах – тим більше таких модулів.

Загалом набор типових модулів обумовлений планом будівлі, кількістю точок подавання води, критичних точок (у яких може бути затоплення), бажанням кінцевого користувача додавати додаткові виконуючі механізми (звукову, світлову індикацію тощо).

Блок віддаленого сервера, зберігання та відображення даних служить для зберігання та відображення для диспетчеру контурів будівель, координат встановлення датчиків відносно контурів, стану датчиків, стану виконавчих механізмів.

2.2 Вибір компонентів та елементів реалізації

Оскільки основною метою розробки в кваліфікаційній роботі є моніторинг затоплення приміщень – найважливішим елементом системи не враховуючи блок керування є датчики затоплення.

У якості датчиків затоплення було обрано датчик Akvarate (рис.2.2).

Рисунок 2.2 – Зовнішній вигляд датчику затоплення Akvarate

Датчик затоплення Akvarate призначений для виявлення затоплення (протікання) в разі виникнення аварійної ситуації в системах водопостачання і опалювання. При виявленні затоплення (протікання) датчик формує тривожний сигнал для контролерів, а також для систем охоронної сигналізації, аварійної сигналізації, диспетчеризації та інших автоматизованих систем контролю і управління.

Датчик призначений для контролю неагресивних рідин по електропровідності відповідних воді. Датчик виконаний з пластика і складається з двох частин: корпусу, в якому знаходиться плата з електронними компонентами залитими епоксидною смолою, контрольними

штирями виконаними з нержавіючої сталі, клемами для підключення кабелю передачі сигналу, і кришкою захищає клеми для підключення кабелю від механічних пошкоджень.

Датчик призначений для використання в наступних умовах навколишнього середовища: температура повітря, що оточує корпус датчика 0 ... + 50oC, відносна вологість повітря (при температурі + 25oC) не більше 80%.

Характеристики:

- напруга живлення – 5…12В;
- струм споживання – не більш 10мА в режимі тривога;
- опір між контактними штирям датчика:
    - в режимі чергування – більше 20МОм;
    - в режимі аварії – менше 1МОм;
- габаритні розміри – 64х27х13 мм;
- маса 15г;
- ступінь захисту – IP44.

Основним виконуючим механізмом системи буде електроклапан для перекривання подачі води. У якості базового електроклапану було обрано електроклапан ARWDFG DN15 AC (рис.2.3).

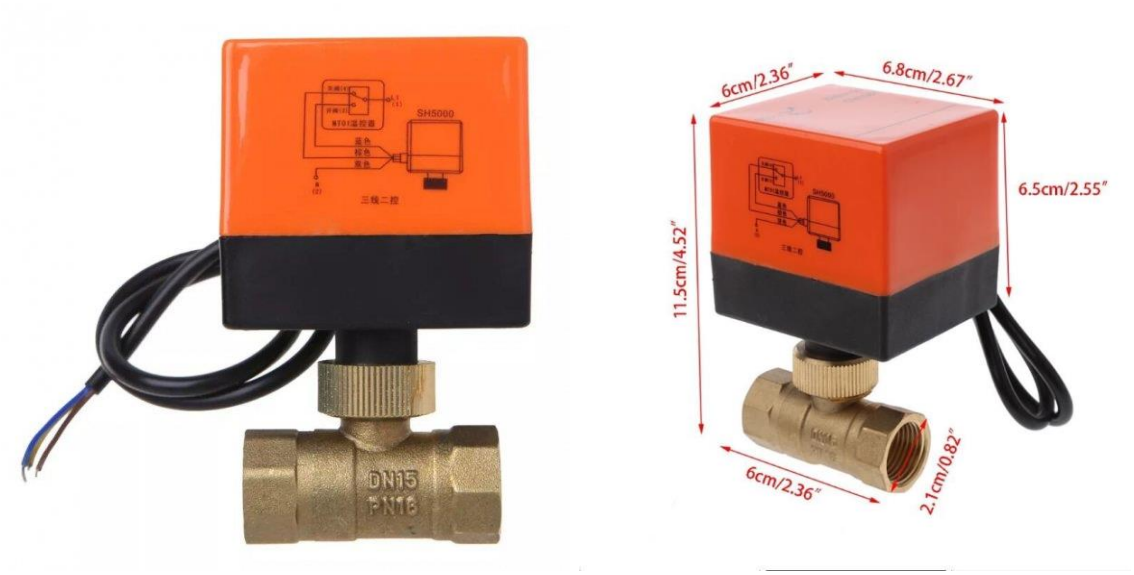

Рисунок 2.3 – Електричний моторизований латунний кран з електричним керуванням ARWDFG DN15 AC

Електричний кран складається з двох елементів - власне самого кульового крана і електродвигуна.

Привід легко розбирається і збирається. Привід може бути встановлений після установки устаткування і труб, щоб підвищити ефективність.

Світлодіодний індикатор вказує на включення / вимикання клапана, щоб визначити, чи повністю клапан включений / виключений.

Електродвигун підключається до кульового через спеціальну муфту, визначення положення закрито - відкрито, здійснюється за допомогою кінцевиків на платі управління двигуном.

Цей електрокран має три проводи керування від мережі змінного напруги 220В. Один з дротів - нуль, він підключений завжди, два інших - керуючі фази, для відкриття і закриття. Зручно підключати до стандартних мереж живлення житлових будинків.

Час потрібний для закриття вентиля - близько 17 секунд. Відкриття відбувається трохи швидше - близько 15 секунд.

У якості центрального елементу модуля керування було обрано мікроконтролер PIC16F887 оскільки він має невелику ціну та має достатньо внутрішніх ресурсів для виконання поставлених задач.

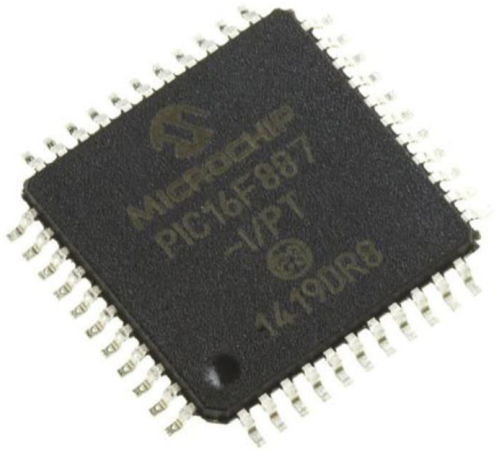

Рисунок 2.4 – Мікроконтролер PIC16F887

Мікроконтролери PIC мають RISC-архітектуру. Серед 8-бітних мікроконтролерів PIC вона складається з 3-х родин, які відрізняються архітектурою (розрядністю і набором команд).

Загальні характеристики:

- об'єм пам'яті програм – 14кБ
- обсяг EEPROM пам'яті – 256Б
- об'єм оперативної пам'яті – 368Б
- напруга живлення – 2…5,5В
- корпус – TQFP-44
- максимальна частота ядра – 20МГц
- тип пам'яті програм – FLASH
- кількість входів / виходів – 35
- інтерфейси – I2C, SPI, UART/USART
- периферія Brown-out Detect / Reset, POR, PWM, WDT
- АЦП / ЦАП – 14х10бит

Для комутації модулю керування в якості WiFi модуля передавання даних обрали модуль ESP8266-12F.

Рисунок 2.5 – Модуль ESP8266-12F

Характеристики ESP8266-12F:

- модель: ESP-12F;
- модуль зібраний на мікроконтролері: ESP8266;
- тактова частота: 80 - 160 МГц;
- флеш пам'ять: 512 кбайт;

- RAM даних: 80 кбайт;

- RAM інструкцій: 32 кбайт;

- підтримка протоколів: 802.11 b / g / n protocol з WEP, WPA, WPA2, Wi-Fi ---- Direct (P2P), soft-AP, Integrated TCP / IP protocol stack;

- посилення: + 19,5dBm в режимі 802.11 b;

- відстань прийому / передачі в ідеальних умовах: 400 м;

- інтерфейси: SDIO 2.0, SPI, UART;

- швидкість UART: 115200 бод / с;

- напруга живлення: 3,3 - 3,6;

- споживаний струм: до 215мА;

- розміри: 24 х 16 х 4 мм;

Для типових блоків збирання даних з датчиків затоплення у якості конвертера сигналів обираємо модуль "Socket-1" 4 входа.

Модуль має програмне керування по протоколу TCP/IP, керується Web-браузером по протоколу HTTP. У модулі виконана гальванічна розв’язка дискретних входів. Напруга живлення модуля 5В або 7...30В постійного струму.

Рисунок 2.6 – Зовнішній вигляд модуля Socket-1

Характеристики:

- напруга живлення 5В або 7 ... 30В постійного струму;

- струм 150 мА;

- інтерфейс Ethernet RJ-45;

- індикація:

- "PWR" - наявність живлення;

- "TCP" - встановлення зв'язку з TCP / IP;

- периферійні функції - 4 входи типу "Сухий контакт" з внутрішньої підтяжкою і гальванічною розв'язкою PC817;

- температурний діапазон роботи -20 .. + 85 С;

- габаритні розміри 70х90х65мм;

- вага 100г.

Для типових блоків збирання даних з можливістю керування електроклапанами було обрано модуль Socket-2W, що вже має вбудований WiFi прийомопередавач .

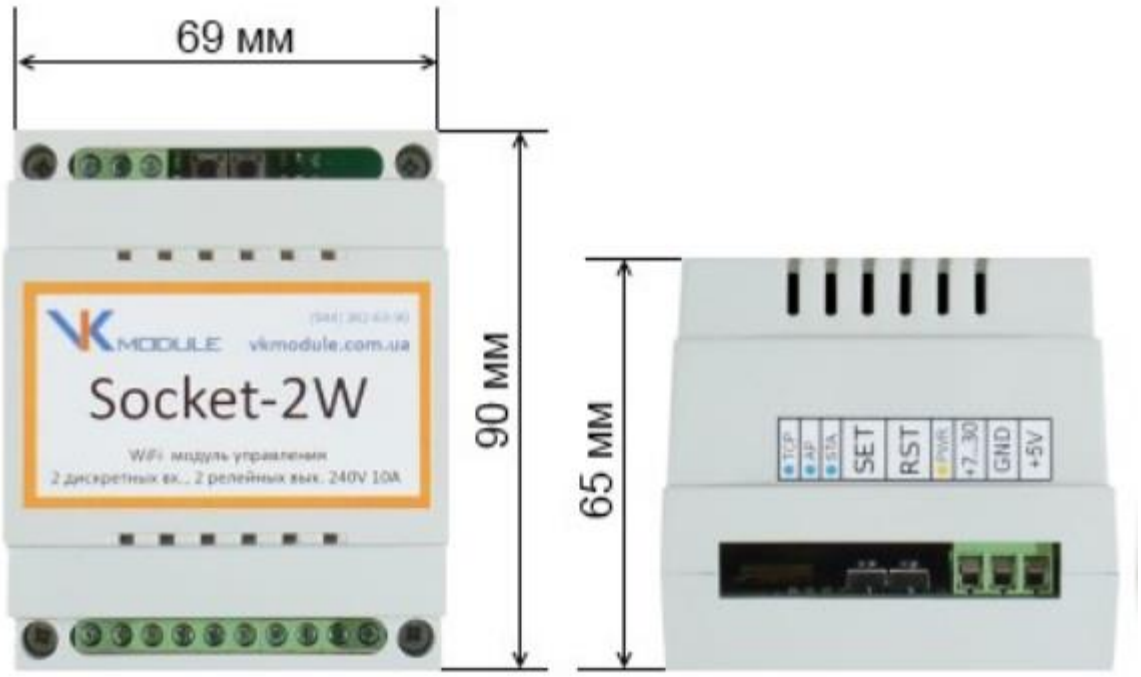

Рисунок 2.7 – Модуль Socket-2W

Модуль Socket-2W має такі особливості:

- робота по бездротовій локальній мережі WiFi;

- 2 Реле з комутованою навантаженням 240В 10А;

- 2 Дискретних входу з опторазвязкой і підтяжкою до 5В;

- 1 Аналоговий вхід (замість 1 дискретного) від 0 до 1.0 В з перетворенням до значення 0 ... 1024;

- програмне управління по протоколах TCP або UDP;

- управління Web-браузером по протоколу HTTP;

- можлива прив'язки станів входів і реле (замикання входу включає реле);

- робота 2-х пристроїв в парі (замикання входу на одному - включення реле на іншому);

- всі налаштування і стани реле зберігаються в незалежній пам'яті;

- напруга живлення 5В або 7 ... 30В постійного струму;

- корпус з можливістю кріплення на DIN-рейку.

2.3 Розробка конструкції

Наступним етапом проектування була розробка конструкції модулю. Оскільки більшість компонентів системи стандартизовані та покупні було проведена розробка конструкції модуля керування. Тобто безпосередньо мікропроцесорного модуля та WiFi модуля прийому/передавання даних.

Розробка конструкції плати виконувалась в програмі Sprint Layout. Обидві друковані плати виконані на двосторонньому фольгованому стеклотекстоліті та мають виготовлятися комбінованим методом.

Плата мікропроцесорного модуля (рис.2.8) з’єднується з платою WiFi модулю передавання даних (рис.2.9) за допомогою UART інтерфейсу.

Конструкція плат виконана таким чином, що вони з’єднуються за допомогою безпаячного роз’ємного з’єднання на плати керування знаходяться роз’їм PLS, а на платі прийомопередавача роз’їм PDS.

Розмір плати керування становить 67,5*57,5мм, а розмір плати передавання даних становить 37,5*37,5.

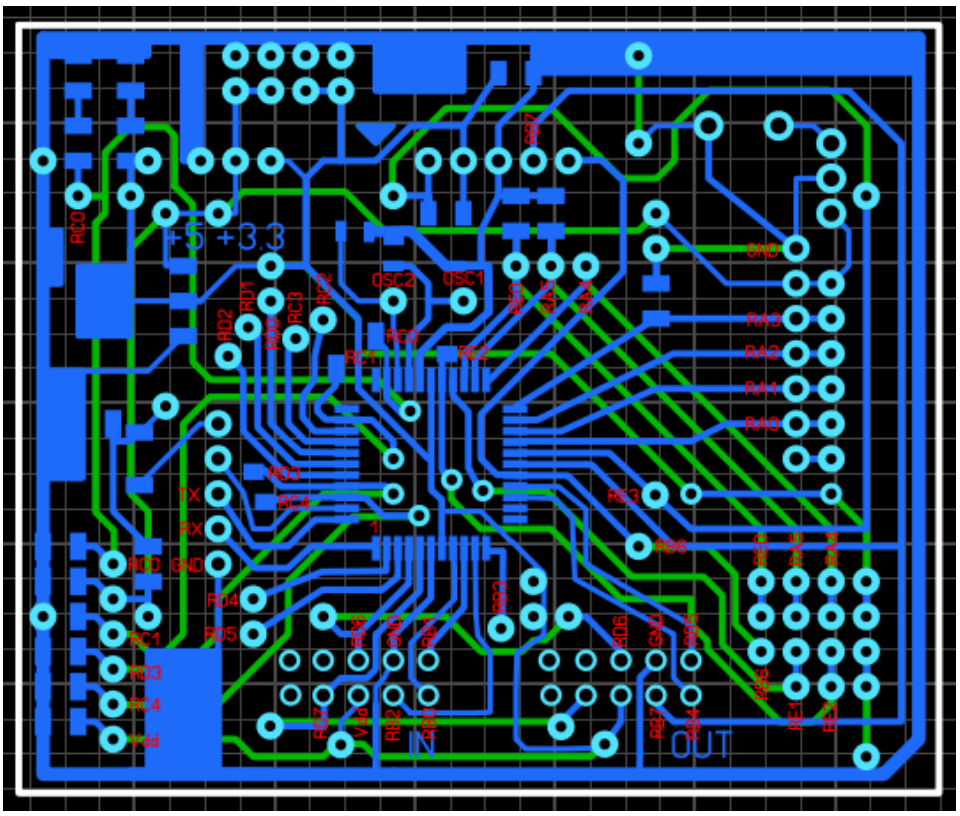

Рисунок 2.8 – Розроблена конструкція плати керування

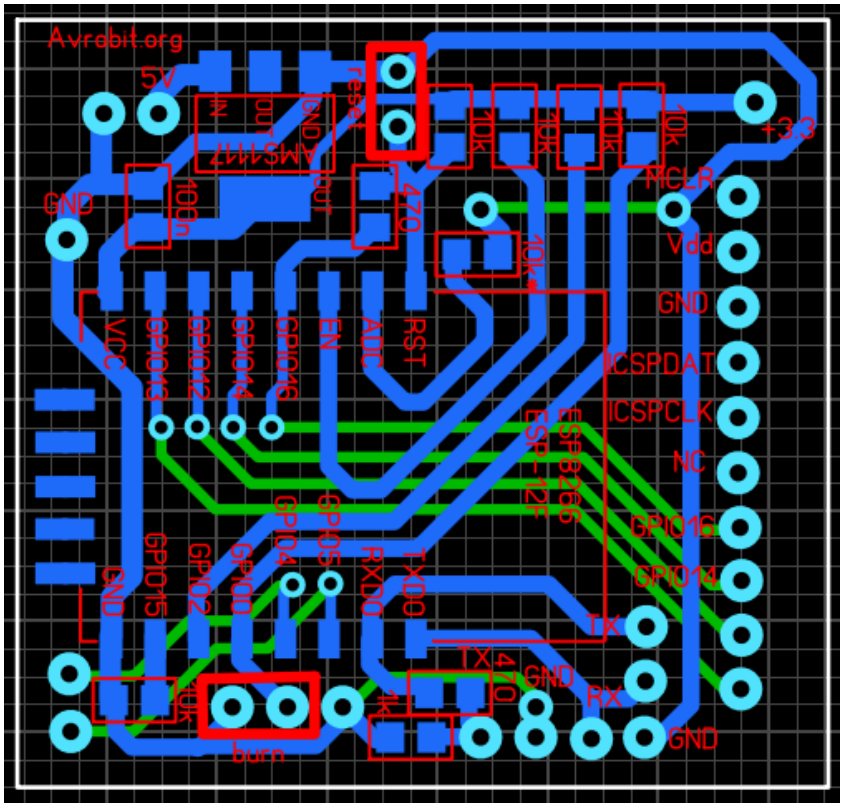

Рисунок 2.9 – Розроблена конструкція плати передавання даних

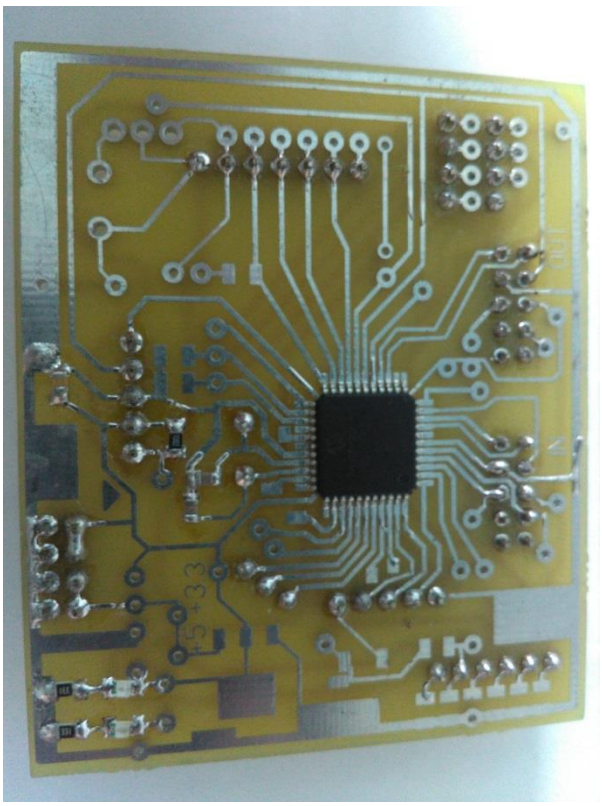

Рисунок 2.10 – Фотографія виготовленого модуля керування

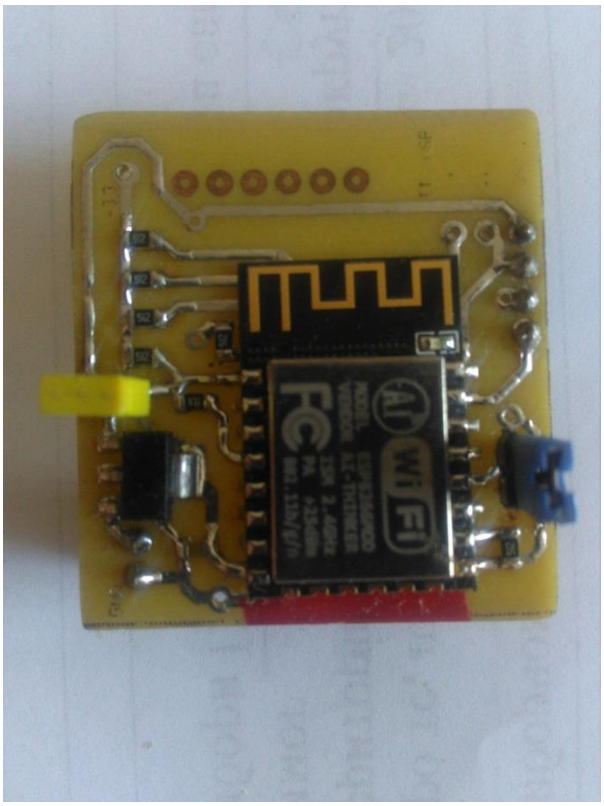

Рисунок 2.10 – Фотографія виготовленого модуля передавання даних

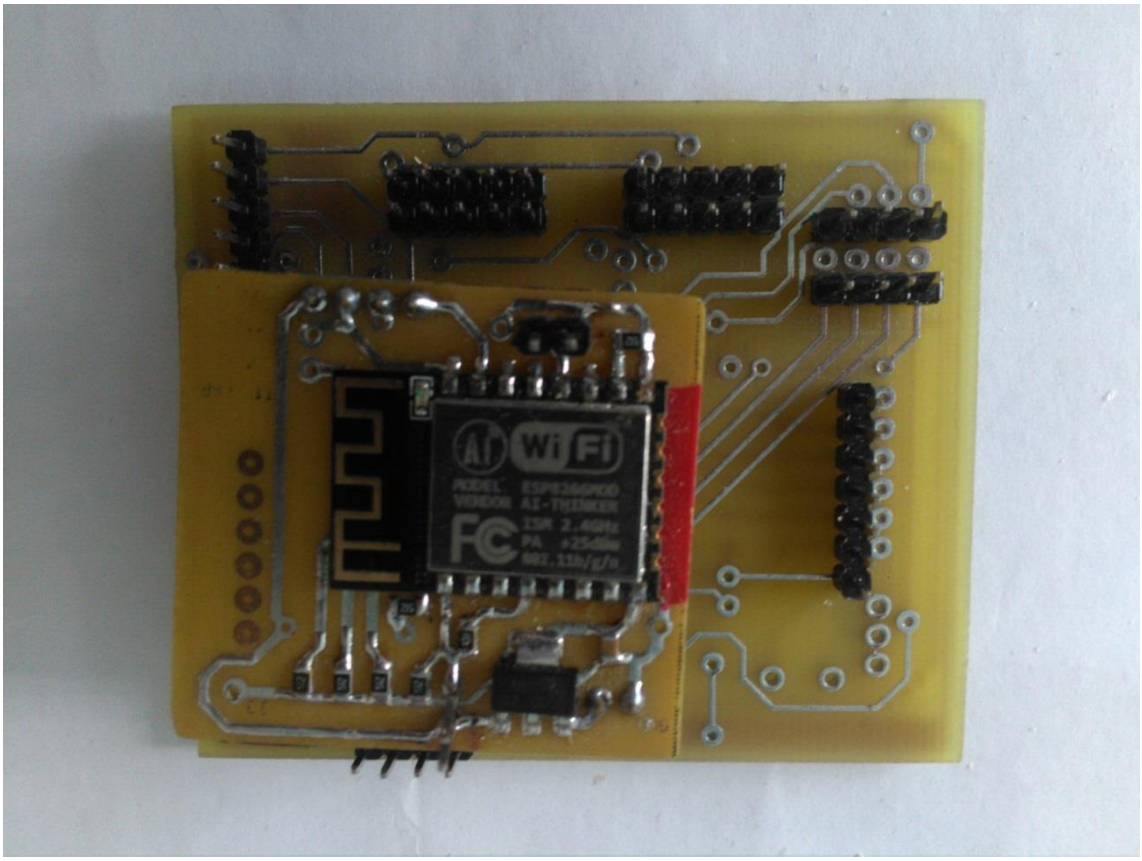

Рисунок 2.11 – З’єднання модулів керування та передавання даних

Розроблений модуль контролю від затоплення підвальних приміщень має невеликі розміри, і оригінальний дизайн. Модуль може встановлюватися як зовнішнім, так і прихованим монтажем, в стандартній електромонтажної коробці. Контролер управління призначений для контролю стану датчиків затоплення та управління виконавчими пристроями з безпечною для життя напругою 12В постійного струму.

Для живлення модуля слід використовувати стабілізоване джерело живлення 12В / 1,2А. При підключенні модуля до джерела живлення необхідно дотримуватись полярності. Як виконавчі пристрої можуть використовуватися як двопровідні, так і трипровідні виконавчі пристрої (електроклапани). Тобто можна використовувати не лише електроклапани запропоновані при виборі компонентів.

Вихід живлення виконавчих пристроїв має електронний захист. Для запобігання утворенню накипу на рухомих частинах виконавчих пристроїв,

контролер управління один раз в 14 днів дає команду на зміну стану виконавчих пристроїв (закрити / відкрити). Для зменшення помилкових відключень води при вологому прибиранні, в контролері управління передбачена затримка часу, від моменту попадання води на датчик затоплення до моменту відключення подачі води від 1 до 16 секунд. Функціональні параметри контролера записані в незалежній пам'яті і не змінюються при відключенні живлення.

Модулі керування та передавання даних було поміщено в корпус у якості якого було обрано базову несучу конструкцію з пластика з герметичним приєднанням кришки, що додатково захистить модулі від можливого вологого повітря

Рисунок 2.12 – Корпус для модулів керування та передавання даних

Кріплення плат до корпусу відбувається за допомогою силіконого клею.

3 НАПИСАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

3.1 Розробка бази даних

Для зберігання на сервері даних, щодо стану датчиків затоплення та поточного стану виконавчих пристроїв таких як електроклапани було створено базу даних в яку сервер зберігає дані для подальшого відображення користувачам у керуючій програмі та у Web додатку.

База включає 8 таблиць:

- таблиця додаткових текстових описів Descriptions;
- таблиця населених пунктів Cities;
- таблиця вулиць Streets;
- таблиця будинків Houses;
- таблиця датчиків Sensors;
- таблиця керуючих компаній Companies;
- таблиця відповідності будинків керуючим компаніям HouseOwners;
- таблиця користувачів Users.

Таблиця 3.1 – Додаткові текстові описи Descriptions

№	Назва стовбцю	Що зберігається	Тип даних	На які таблиці посилається
1	Id	Номер запису (ключ таблиці)	INT	–
2	DescrText	Текст опису	TEXT	–

SQL запит створення таблиці:

CREATE TABLE Descriptions (Id INT, DescrText TEXT, PRIMARY KEY (Id));

Оскільки розроблена система контролю затоплення має у майбутньому встановлюватись на велику кількість підвалів у різних населених пунктах було створено таблицю населених пунктів (табл.3.2) на яку будуть

посилатися наступні таблиці для більш повного встановлення місцезнаходження. В одному населеному пункті може бути розташовано сотні або тисячі систем контролю затоплення, а отже назва населеного пункту буде повторюватись – зручніше це проіндексувати та зробити у вигляді посилання.

Таблиця 3.2 – Населені пункти Cities

№	Назва стовбцю	Що зберігається	Тип даних	На які таблиці посилається
1	Id	Номер запису (ключ таблиці)	INT	–
2	CityName	Назва населеного пункту	VARCHAR(100)	–
3	CityType	Тип населеного пункту	INT	–
4	Descr	Посилання на додатковий опис	INT	Descriptions(Id)

Стовпець CityName зберігає назву населеного пункту у текстовому вигляді.

Стовпець CityType зберігає тип населеного пункту, який може приймати наступні значення:

- 0 (NULL) – тип населеного пункту не відомий;
- 1 – місто;
- 2 – селище міського типу;
- 3 – село.

Стовпець Descr це посилання на додатковий текстовий опис (таблиця Descriptions) населеного пункту у разі такої необхідності.

SQL запит створення таблиці:

CREATE TABLE Cities (Id INT, CityName VARCHAR(100), CityType INT, Descr INT , PRIMARY KEY (Id), FOREIGN KEY (Descr) REFERENCES Descriptions(Id));

Для зберігання даних про вулиці було створено окрему таблицю Streets (табл.3.3).

Таблиця 3.3 – Вулиці Streets

№	Назва стовбцю	Що зберігається	Тип даних	На які таблиці посилається
1	Id	Номер запису (ключ таблиці)	INT	–
2	StreetName	Назва вулиці	VARCHAR(100)	–
3	StreetType	Тип вулиці	INT	–
4	CityId	Посилання на населений пункт	INT	Cities(Id)
5	Descr	Посилання на додатковий опис	INT	Descriptions(Id)

Стовпець StreetName зберігає текст назви вулиці.

Стовпець StreetType зберігає тип вулиці у числовому вигляді, щоб займати менше памяті в БД та може приймати наступні значення:

- 0 (NULL) тип вулиці не встановлено;
- 1 – проспект;
- 2 – вулиця;
- 3 – бульвар;
- 4 – провулок;
- 5 – проїзд.

Стовпець CityId зберігає посилання на населений пункт у таблиці Cities оскільки назви вулиць у різних населених пунктах можуть співпадати, а отже треба ідентифікувати яка саме вулиця в якому населеному пункті мається на увазі.

SQL запит створення таблиці: CREATE TABLE Streets (Id INT, StreetName VARCHAR(100), StreetType INT, CityId INT, Descr INT ,

PRIMARY KEY (Id), FOREIGN KEY (CityId) REFERENCES Cities(Id), FOREIGN KEY (Descr) REFERENCES Descriptions(Id));

Для окремих будинків, де встановлено систему контролю затоплення підвалів було створено таблицю будинків (табл.3.4).

Таблиця 3.4 – Будинки Houses

№	Назва стовбцю	Що зберігається	Тип даних	На які таблиці посилається
1	Id	Номер запису (ключ таблиці)	INT	–
2	HouseNumb	Номер дому	VARCHAR(10)	–
3	StreetId	Посилання на вулицю на якій розташований будинок	INT	Streets(Id)
4	Latitude	Широта	FLOAT	–
5	Longitude	Довгота	FLOAT	–
6	Descr	Посилання на додатковий опис	INT	Descriptions(Id)

Стовпець HouseNumb зберігає номер будинку. Тип записів у цьому стовбці текстовий адже деякі будинки у номері мають букви (наприклад «141а»).

Стовпець StreetId зберігає посилання - індекс в таблиці вулиць, щоб ідентифікувати до якої вулиці належить будинок.

Стовбці Latitude та Longitude зберігають координати розташування будинку (широту та довготу), що дозволяє робити пошук будинку на території по карті та виводити розташування. Це спрощує роботу з системою боку адміністраторів системи та представників керуючих компаній у яких кількість домів може сягати десятків та навіть тисяч (у випадку адміністраторів системи).

SQL запит створення таблиці:

CREATE TABLE Houses (Id INT, HouseNumb VARCHAR(10), StreetId INT, Latitude FLOAT, Longitude FLOAT, Descr INT , PRIMARY KEY (Id), FOREIGN KEY (StreetId) REFERENCES Streets(Id), FOREIGN KEY (Descr) REFERENCES Descriptions(Id));

Для зберігання інформації, щодо датчиків які встановлено у домах розроблено таблицю Sensors (табл.3.5).

Ця таблиця вказує в якому домі встановлена система HouseId, дату встановлення EquipDate, унікальний ідентифікатор датчику, що встановлений у цей ліфт EquipID, тип лифтового обладнання (модель ліфту), номер під'їзду у домі де встановлено обладнання Porch.

Таблиця 3.5– Датчики Sensors

№	Назва стовбцю	Що зберігається	Тип даних	На які таблиці посилається
1	Id	Номер запису (ключ таблиці)	INT	–
2	EquipID	Унікальний ІД модулю контролю встановлений у ліфті	VARCHAR(20)	–
3	CoordX	Координата зміщення по горизонталі від верхнього лівого кута будинку на плані (в см)	INT	
4	CoordY	Координата зміщення по вертикалі від верхнього лівого кута будинку на плані (в см)	INT	
5	HouseId	Посилання на будинок	INT	Houses(Id)
6	Descr	Посилання на додатковий опис	INT	Descriptions(Id)

SQL запит створення таблиці:

CREATE TABLE Sensors (Id INT, EquipID VARCHAR(20), CoordX INT, CoordY INT, Type INT, HouseId INT, Porch INT, Descr INT , PRIMARY KEY (Id), FOREIGN KEY (HouseId) REFERENCES Houses(Id), FOREIGN KEY (Descr) REFERENCES Descriptions(Id));

Для зберігання даних щодо керуючих компаній та ОСББ було розроблено таблицю Companies (табл.3.6), що зберігає назву компанії Name, її тип (керуюча компанія або ОСББ) Type та у разі необхідності додатковий опис як посилання на таблицю описів.

Таблиця 3.6 – Керуючі компанії Companies

№	Назва стовбцю	Що зберігається	Тип даних	На які таблиці посилається
1	Id	Номер запису (ключ таблиці)	INT	–
2	Name	Назва компанії	VARCHAR(200)	–
3	Type	Тип компанії	INT	–
4	Descr	Посилання на додатковий опис	INT	Descriptions(Id)

SQL запит створення таблиці:

CREATE TABLE Companies (Id INT, Name VARCHAR(200), Type INT, Descr INT , PRIMARY KEY (Id), FOREIGN KEY (Descr) REFERENCES Descriptions(Id));

Оскільки одна керуюча компанія, ОСББ або асоціація власників як правило мають у обслугованні більш одного будинку та відповідно буде встановлено не одну систему контролю затоплення було створено таблицю відповідності будинків керуючим компаніям. Ця таблиця використовується для відображення переліку домів та ліфтів з системою керування при виборі

окремої компанії та/або вході вповноваженої особи для перегляду та редагування даних.

Таблиця 3.7 – Відповідність будинків керуючим компаніям HouseOwners

№	Назва стовбцю	Що зберігається	Тип даних	На які таблиці посилається
1	Id	Номер запису (ключ таблиці)	INT	–
2	CompanyId	Посилання на керуючу компанію	INT	Companies(Id)
3	HouseId	Посилання на будинок	INT	Houses(Id)
4	Descr	Посилання на додатковий опис	INT	Descriptions(Id)

SQL запит створення таблиці:

CREATE TABLE HouseOwners (Id INT, CompanyId INT, HouseId INT, Descr INT , PRIMARY KEY (Id), FOREIGN KEY (CompanyId) REFERENCES Companies(Id), FOREIGN KEY (HouseId) REFERENCES Houses(Id), FOREIGN KEY (Descr) REFERENCES Descriptions(Id));

Для керування входами у програму або Web додаток створено таблицю Users яка покликана зберігати дані авторизації користувачів системи при вході через термінальне ПЗ та WEB сторінку користувача/адміністратора.

Таблиця зберігає логін користувача Login, хеш паролю PassHash, рівень доступу AccLvl, посилання на керуючу компанію, що обслуговує квартиру, в якій проживає жилець (у випадку користувача) або до адміністрування якої відноситься адміністратор CompanyId.

Таблиця 3.8 – Користувачі Users

№	Назва стовбцю	Що зберігається	Тип даних	На які таблиці посилається
1	Id	Номер запису (ключ таблиці)	INT	
2	Login	Логін користувача	VARCHAR(100)	
3	PassHash	Хеш паролю	CHAR(32)	
4	ManName	Ім’я користувача	VARCHAR(100)	
5	AccLvl	Рівень доступу	INT	
6	CompanyId	Посилання на керуючу компанію	INT	Companies(Id)
7	Descr	Посилання на додатковий опис	INT	Descriptions(Id)

Рівень доступу зберігається у вигляді коду та може приймати наступні значення:

- 0 (NULL) – звичайний користувач якому доступний лише перегляд поточного стану датчиків через Web додаток;

- 1 – уповноважена особа керуючої компанії (ОСББ, асоціації власників), що має можливість перегляду стану доступів всіх квартир всіх домів, що обслуговуються компанією;

- 2 – уповноважена особа керуючої компанії (ОСББ, асоціації власників), що має можливість редагування стану доступів, додавання та видалення ключів доступу, редагування, додавання та зміни жильців всіх квартир всіх домів, що обслуговуються компанією;

- 3 – адміністратор системи, що має можливості перегляду, редагування всієї бази системи.

SQL запит створення таблиці:

CREATE TABLE Users (Id INT, Login VARCHAR(100), PassHash CHAR(32), ManName INT, AccLvl INT, CompanyId INT, FlatId INT, Descr INT

, PRIMARY KEY (Id), FOREIGN KEY (ManId) REFERENCES Man(Id), FOREIGN KEY (CompanyId) REFERENCES Companies(Id), FOREIGN KEY (FlatId) REFERENCES Flats(Id), FOREIGN KEY (Descr) REFERENCES Descriptions(Id));

3.2 Розробка протоколу обміну та написання програмного забезпечення

Після розробки апаратної частини модулю контролю затоплення та створення БД було розроблено протоколу обміну даними між окремими модулями системи.

У табл.3.14 наведено команди обміну даними між модулем керування та вузлами збору даних з датчиків та керування електроклапанами.

Для початку роботи повинні бути створені умови для забезпечення зв'язку декількох модулів одним із способів

Спосіб 1. Модулі підключаються по WiFi до одного і того ж роутера в одній локальній мережі.

Спосіб 2. Модулі підключаються по WiFi до різних роутерів і знаходяться в різних локальних мережах, а роутери забезпечують видимість модулів іншим через "перекидання портів".

Спосіб 3. Один модуль є точкою доступу, а інші підключаються до нього як клієнти. При цьому вони самі утворюють свою локальну мережу.

Команди отримання даних з датчиків та керування електроклапанами наведені в таблиці 3.9.

Обмін інформацією здійснюється пакетами. Запитальний пакет від ПК до модуля далі називається «команда», пакет від модуля називається «подія». Команди події діляться на загальні, які підтримуються кожним модулем і індивідуальні, які підтримуються тільки конкретним типом модуля. Будь-яка команда і подія складається з наступних частин: -1 байт ідентифікатора/коду команди (ID); - 0-3 байт дані команди або події. Далі наведені команди і

відповідні їм події. Не всім подіям можуть передувати команди. Наприклад подія зміни стану цифрового входу генерується автоматично.

Таблиця 3.9 – Опис команд керування та отримання даних

№	Код команди	Призначення	Данні	Результат
1	0x01	Перевірка зв’язку	Відсутні	0x01
2	0x02	Перегрузити модуль	Відсутні	Відсутній
3	0x03	Запросити версію прошивки	Відсутні	Номер версії
4	0x04	Запросити серійний номер (ідентифікатор)	Відсутні	ІД
5	0x0F	Вказання модулем, що він отримав некоректні данні	Отриманий пакет	Відсутній
6	0x30	Встановити налаштування входу	Номер входу, включення, довжина антидребезгу	Повторення отриманих даних
7	0x31	Отримати налаштування входу	Відсутні	Номер входу, включення, довжина антидребезгу
8	0x32	Отримати стан входів	Кількість байт з 0/1 що відповідає кількості входів	
9	0x22	Вкл/викл реле	Номер реле, вкл./викл, час на який включити	

Програма мікроконтролера модуля керування була написана на асемблері в середовищі розробки MPLab IDE.

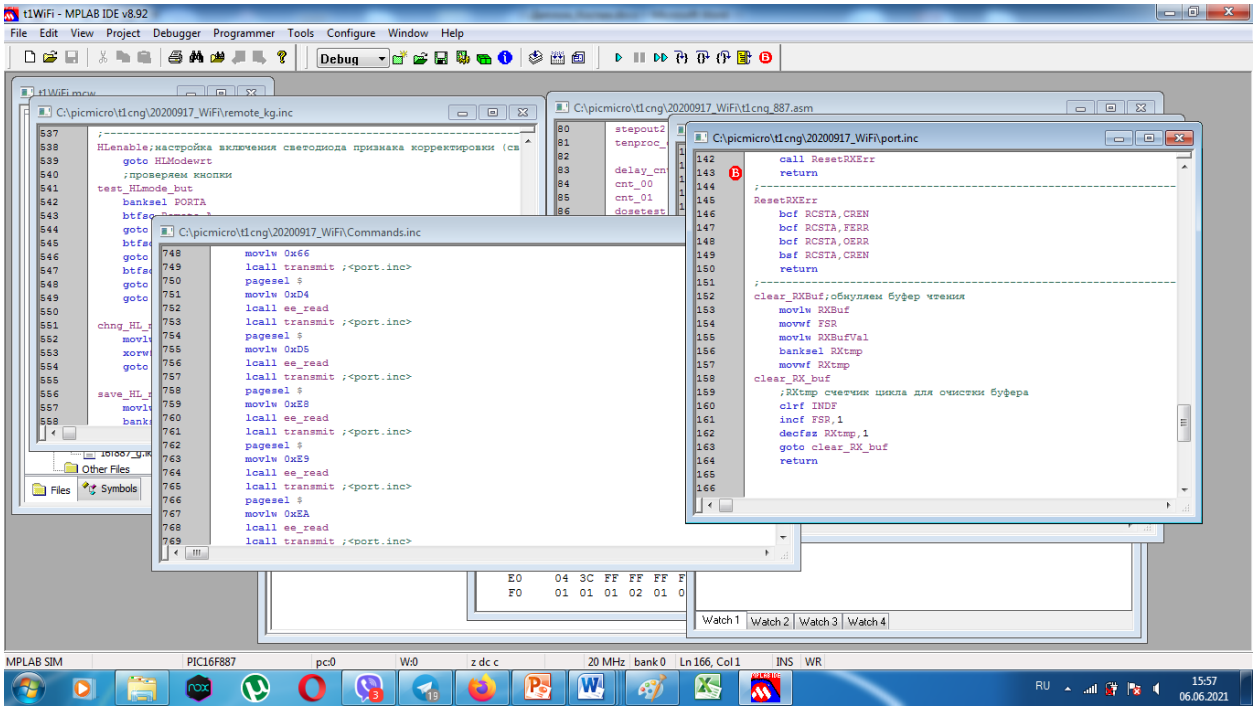

Рисунок 3.1 – Відлагодження програми мікроконтролера в MPLab IDE

Програма керування модулями була написана на мові C# в середовищі Visual Studio код програми основної форми наведено у додатку А.

ВИСНОВКИ

Розглядаючи тему автоматизації бізнес-процесів, можна сказати, що на даний момент існує багато рішень в різних сферах. Практично кожне з них, дійсно, дозволяє економити і оптимізувати використання ресурсів. Але в області автоматизації процесів саме управління житловими комплексами на сьогоднішній момент запропоновано не так багато ефективних рішень.

В кваліфікаційні роботі було розроблено систему контролю затоплення підвальних приміщень та окремо конструкцію модулю контролю затоплення підвальних приміщень з можливістю перекривання електроклапанів подачі води.

В роботі було виконано наступні роботи:

- розроблено структурну схему роботи системи контролю затоплення;
- обрано компоненти системи з врахуванням низької ціни та високої функціональності з можливістю підлаштовувати систему під різні приміщення;
- розроблено конструкцію модулю контролю затоплення (блоку керування системи);
- розроблено базу даних для віддаленого сервера диспетчерізації;
- розроблено протоколи обміну між блоками системи;
- розроблено програмне забезпечення керуючого мікроконтролера.

Основною перевагою розробленої системи в цілому та модуля зокрема є використання бездротової WiFi мережі зв’язку між блоками системи, що дозволяє скоротити витрати на встановлення системи та надає можливість змінювати конфігурацію системи під час її використання. А можливість віддаленої диспетчирізації через інтернет надасть можливості швидко реагувати на аварійні ситуації.

ПЕРЕЛІК ЛІТЕРАТУРИ

1. Аболин А.А. Основные направления реформирования ЖКХ муниципальных образований: Статья. 2005. No 2. –37-40 с.

2. Архипов В., Системы для «умного» здания: М.: «СтройМаркет», 2015. No 45. –182с.

3. Асаул А.Н., Карасев А.В. Экономика недвижимости: Учеб. пособие. МИКХиС. 2015. –247 с.

4. Богданова Ю.В., Ильиных Д.В., Патудин В.М., Подольская А.Я. Моделирование организационно-экономических механизмов системы ЖКХ в рыночной экономике // Ползуновский вестник N1. 2016. –56 с.

5. Грахов В.П., Мохначев С. А., Егорова В. Г. Эффективность энергосберегающих мероприятий в жилищном строительстве // Современные проблемы науки и образования. 2015. No 2. –273с.

6. Дементьев А. «Умный» дом XXI века. М.: Издательские решения,2017. –174с.

7. Дрозд Н.И., Симионов Ю.Ф. Жилищнокоммунальное хозяйство: Справочник, М.: ИКЦ «МарТ»; Ростов н/Д: ИЦ «МарТ», 2015. –272 с.

8. Жуков Д.М. Экономика и организация жилищно-коммунального хозяйства города, М.: ВЛАДОС-ПРЕСС, 2016. –96 с.

9. Сиваев С.Б. Как эффективно управлять жилищным фондом: теория и практика //Учеб. пособие. М.: Фонд «Институт экономикигорода», 2017. –217 с.

10. Трояновский В.М. Информационно-управляющие системы и прикладная теорияслучайных процессов, М.: Гелиос АРВ, 2017. –304 с.

11. Wellsoft. Обзор рынка автоматизации предприятий: решения для строительных и управляющих компаний в сфере ЖКХ. –М.: Habr. 2018. –13 с.

12. Диспетчеризация объектов ЖКХ [Сайт] –URL: http://cons-systems.ru/dispetcherizatciya-obektov-zhkkh (дата обращения: 18.05.2021)

13. Официальный сайт системы автоматизации процессов управления жилыми комплексами и бизнес-центрами и монетизации услуг ТСЖ/УК HESKEY[Сайт] –URL: http://heskey.ru/(дата обращения: 23.05.2021).

ДОДАТОК А – КОД ПРОГРАМИ КЕРУВАННЯ МОДУЛЯМИ ДЛЯ ПК (ОСНОВНИЙ МОДУЛЬ)

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;
using System.Net.Sockets;
using System.Threading;
using System.Text.RegularExpressions;

namespace Socket_2_Protocol
{
	public partial class MainForm : Form
	{
		//делегат для обращения к компоненту ListView из потока опроса сокета
	public delegate void listViewUpdateCallBack(string param1, string param2, string param3, string param4);

	System.Net.Sockets.TcpClient clientSocket = new System.Net.Sockets.TcpClient();
	//поток чтения-записи
	NetworkStream serverStream = default(NetworkStream);
	Thread ctThread;
	string patternIP;		//регулярное выражение для проверки правильности введенного IP-адреса


		public MainForm()
		{
			InitializeComponent();
			patternIP = "^([01]?\\d\\d?|2[0-4]\\d|25[0-5])\\.([01]?\\d\\d?|2[0-4]\\d|25[0-5])\\.([01]?\\d\\d?|2[0-4]\\d|25[0-5])\\.([01]?\\d\\d?|2[0-4]\\d|25[0-5])$";
		}

		void MainFormLoad(object sender, EventArgs e)
		{
			try
	{
	  Cursor = Cursors.AppStarting;
	  cmbCmd20_0.SelectedIndex = 0;
	  cmbCmd20_1.SelectedIndex = 0;
	  cmbCmd21_0.SelectedIndex = 0;
	  cmbCmd22_0.SelectedIndex = 0;
	  cmbCmd22_1.SelectedIndex = 0;

	  Cursor = Cursors.Default;
	}
	catch(Exception er)
	{
	  Cursor = Cursors.Default;
	  MessageBox.Show(er.Message);
	  Application.Exit();
	}
		}

		void MainFormFormClosing(object sender, FormClosingEventArgs e)
```

```
                {
            if (clientSocket.Connected)
            {
               ctThread.Abort();
               clientSocket.Client.Disconnect(true);
            }
                }



                //функция обновления компонента ListView
                public void listViewUpdate(string param1, string param2, string param3, string param4)//функция обновления компонента ListView
        {
                        try
                        {
                    if (this.lstEvents.InvokeRequired)    //если компонент вызывается из другого потока
                    {
                       listViewUpdateCallBack d = new listViewUpdateCallBack(listViewUpdate);  //создаем делегат
                       this.Invoke(d, new object[] { param1, param2, param3, param4 });
    //вызываем делегат
                    }
                    else        //иначе обновляем компонент
                    {
                       this.lstEvents.Items.Add(param1);
                       this.lstEvents.Items[lstEvents.Items.Count-1].SubItems.Add(param2);
                       this.lstEvents.Items[lstEvents.Items.Count - 1].SubItems.Add(param3);
                       this.lstEvents.Items[lstEvents.Items.Count - 1].SubItems.Add(param4);
                       this.lstEvents.Items[lstEvents.Items.Count - 1].ImageIndex = 1;
                       this.lstEvents.EnsureVisible(lstEvents.Items.Count - 1);
                    }
                        }
                        catch (Exception ex)
            {
               MessageBox.Show(ex.Message, this.Text, MessageBoxButtons.OK, MessageBoxIcon.Error, MessageBoxDefaultButton.Button1);
            }
        }
                //функция преобразования байта в 16-ричную строку
                private String ByteToHex(Byte BT)
        {
            Int32 iTmpH = BT / (Byte)16;
            Int32 iTmpL = BT % (Byte)16;
            String ret = "";

            if (iTmpH == 0) ret = "0";
            if (iTmpH == 1) ret = "1";
            if (iTmpH == 2) ret = "2";
            if (iTmpH == 3) ret = "3";
            if (iTmpH == 4) ret = "4";
            if (iTmpH == 5) ret = "5";
            if (iTmpH == 6) ret = "6";
            if (iTmpH == 7) ret = "7";
            if (iTmpH == 8) ret = "8";
            if (iTmpH == 9) ret = "9";
            if (iTmpH == 10) ret = "A";
            if (iTmpH == 11) ret = "B";
            if (iTmpH == 12) ret = "C";
            if (iTmpH == 13) ret = "D";
            if (iTmpH == 14) ret = "E";
            if (iTmpH == 15) ret = "F";
```

```
            if (iTmpL == 0) ret += "0";
            if (iTmpL == 1) ret += "1";
            if (iTmpL == 2) ret += "2";
            if (iTmpL == 3) ret += "3";
            if (iTmpL == 4) ret += "4";
            if (iTmpL == 5) ret += "5";
            if (iTmpL == 6) ret += "6";
            if (iTmpL == 7) ret += "7";
            if (iTmpL == 8) ret += "8";
            if (iTmpL == 9) ret += "9";
            if (iTmpL == 10) ret += "A";
            if (iTmpL == 11) ret += "B";
            if (iTmpL == 12) ret += "C";
            if (iTmpL == 13) ret += "D";
            if (iTmpL == 14) ret += "E";
            if (iTmpL == 15) ret += "F";

            return ret;
        }

                    //          КОМАНДЫ
                    // 0x01 Проверка связи
                    void BtnCmd01Click(object sender, EventArgs e)
                    {
                        try
                        {
                            if (clientSocket.Connected)                    //  если  соединение
установлено
                    {
                      String message = "";
                      Byte[] mes = new Byte[1];                            //переменная,  которая  будет
содержать данные для отправки

                      mes[0] = 0x01;
                      message = "01";          //сообщение для отображения в ListView

                      string  DT  =  DateTime.Now.Hour.ToString("00")  +  ":"  +
DateTime.Now.Minute.ToString("00")  +  ":"  +  DateTime.Now.Second.ToString("00")  +  "."  +
DateTime.Now.Millisecond.ToString("000");
                      listViewUpdate(DT, "Команда", message, "Проверка связи");

                      serverStream.Write(mes, 0, 1);
                      serverStream.Flush();
                    }
                    else
                    {
                        MessageBox.Show("Соединение  не  установлено.",  this.Text,
MessageBoxButtons.OK, MessageBoxIcon.Warning, MessageBoxDefaultButton.Button1);
                    }
            }
                        catch (Exception ex)
            {
              MessageBox.Show(ex.Message,  this.Text,  MessageBoxButtons.OK,  MessageBoxIcon.Error,
MessageBoxDefaultButton.Button1);
            }
                     }

                    // 0x02 Перезагрузка модуль
                    void BtnCmd02Click(object sender, EventArgs e)
                    {
```

```
                    if (MessageBox.Show("Перезагрузка приведет к потере связи!\r\nВыполнить перезагрузку?", this.Text, MessageBoxButtons.YesNo, MessageBoxIcon.Warning, MessageBoxDefaultButton.Button2) == DialogResult.No)
                        return;
                    try
                    {
                        if (clientSocket.Connected)                    // если соединение установлено
                {
                    String message = "";
                    Byte[] mes = new Byte[1];                          //переменная, которая будет содержать данные для отправки

                    mes[0] = 0x02;
                    message = "02";             //сообщение для отображения в ListView

                    string DT = DateTime.Now.Hour.ToString("00") + ":" + DateTime.Now.Minute.ToString("00") + ":" + DateTime.Now.Second.ToString("00") + "." + DateTime.Now.Millisecond.ToString("000");
                    listViewUpdate(DT, "Команда", message, "Проверка связи");

                    serverStream.Write(mes, 0, 1);
                    serverStream.Flush();
                }
                else
                {
                        MessageBox.Show("Соединение не установлено.", this.Text, MessageBoxButtons.OK, MessageBoxIcon.Warning, MessageBoxDefaultButton.Button1);
                }
            }
                        catch (Exception ex)
            {
                MessageBox.Show(ex.Message, this.Text, MessageBoxButtons.OK, MessageBoxIcon.Error, MessageBoxDefaultButton.Button1);
            }
                }

                // 0x03 Запрос типа модуля
                void BtnCmd03Click(object sender, EventArgs e)
                {
                    try
                    {
                        if (clientSocket.Connected)                    // если соединение установлено
                {
                    String message = "";
                    Byte[] mes = new Byte[1];                          //переменная, которая будет содержать данные для отправки

                    mes[0] = 0x03;
                    message = "03";             //сообщение для отображения в ListView

                    string DT = DateTime.Now.Hour.ToString("00") + ":" + DateTime.Now.Minute.ToString("00") + ":" + DateTime.Now.Second.ToString("00") + "." + DateTime.Now.Millisecond.ToString("000");
                    listViewUpdate(DT, "Команда", message, "Запрос типа модуля");

                    serverStream.Write(mes, 0, 1);
                    serverStream.Flush();
                }
                else
                {
```

```
                    MessageBox.Show("Соединение не установлено.", this.Text, MessageBoxButtons.OK, MessageBoxIcon.Warning, MessageBoxDefaultButton.Button1);
                }
            }
                    catch (Exception ex)
            {
              MessageBox.Show(ex.Message, this.Text, MessageBoxButtons.OK, MessageBoxIcon.Error, MessageBoxDefaultButton.Button1);
            }        }

                // 0x04 Запрос серийного номера
                void BtnCmd04Click(object sender, EventArgs e)
                {
                    try
                    {
                        if (clientSocket.Connected)                    // если соединение установлено
                {
                  String message = "";
                  Byte[] mes = new Byte[1];                            //переменная, которая будет содержать данные для отправки

                  mes[0] = 0x04;
                  message = "04";           //сообщение для отображения в ListView

                  string DT = DateTime.Now.Hour.ToString("00") + ":" + DateTime.Now.Minute.ToString("00") + ":" + DateTime.Now.Second.ToString("00") + "." + DateTime.Now.Millisecond.ToString("000");
                  listViewUpdate(DT, "Команда", message, "Запрос серийного номера");

                  serverStream.Write(mes, 0, 1);
                  serverStream.Flush();
                }
                else
                {
                    MessageBox.Show("Соединение не установлено.", this.Text, MessageBoxButtons.OK, MessageBoxIcon.Warning, MessageBoxDefaultButton.Button1);
                }
            }
                    catch (Exception ex)
            {
              MessageBox.Show(ex.Message, this.Text, MessageBoxButtons.OK, MessageBoxIcon.Error, MessageBoxDefaultButton.Button1);
            }
                }

                // 0x20 Установить настройки входа
                void BtnCmd20Click(object sender, EventArgs e)
                {
                    try
                    {
                        if (clientSocket.Connected)                    // если соединение установлено
                {
                  String message = "";
                  Byte[] mes = new Byte[4];                            //переменная, которая будет содержать данные для отправки

                  mes[0] = 0x20;
                  mes[1] = (byte)cmbCmd20_0.SelectedIndex;
                  mes[2] = (byte)cmbCmd20_1.SelectedIndex;
                  mes[3] = (byte)nmCmd20_2.Value;
```

```
                    message = "20 " + ByteToHex(mes[1]) + " " + ByteToHex(mes[2]) + " " + ByteToHex(mes[3]);          //сообщение для отображения в ListView

                    string DT = DateTime.Now.Hour.ToString("00") + ":" + DateTime.Now.Minute.ToString("00") + ":" + DateTime.Now.Second.ToString("00") + "." + DateTime.Now.Millisecond.ToString("000");
                    listViewUpdate(DT, "Команда", message, "Установить настройки входа");

                    serverStream.Write(mes, 0, 4);
                    serverStream.Flush();
                }
                else
                {
                        MessageBox.Show("Соединение не установлено.", this.Text, MessageBoxButtons.OK, MessageBoxIcon.Warning, MessageBoxDefaultButton.Button1);
                }
            }
                        catch (Exception ex)
            {
              MessageBox.Show(ex.Message, this.Text, MessageBoxButtons.OK, MessageBoxIcon.Error, MessageBoxDefaultButton.Button1);
            }
             }


             // 0x21 Запросить настройки входа
             void BtnCmd21Click(object sender, EventArgs e)
             {
                    try
                    {
                        if (clientSocket.Connected)                 // если соединение установлено
                {
                  String message = "";
                  Byte[] mes = new Byte[2];                         //переменная, которая будет содержать данные для отправки

                  mes[0] = 0x21;
                  mes[1] = (byte)cmbCmd21_0.SelectedIndex;
                  message = "21 " + ByteToHex(mes[1]);              //сообщение для отображения в ListView

                  string DT = DateTime.Now.Hour.ToString("00") + ":" + DateTime.Now.Minute.ToString("00") + ":" + DateTime.Now.Second.ToString("00") + "." + DateTime.Now.Millisecond.ToString("000");
                  listViewUpdate(DT, "Команда", message, "Запросить настройки входа");

                  serverStream.Write(mes, 0, 2);
                  serverStream.Flush();
                }
                else
                {
                        MessageBox.Show("Соединение не установлено.", this.Text, MessageBoxButtons.OK, MessageBoxIcon.Warning, MessageBoxDefaultButton.Button1);
                }
            }
                        catch (Exception ex)
            {
              MessageBox.Show(ex.Message, this.Text, MessageBoxButtons.OK, MessageBoxIcon.Error, MessageBoxDefaultButton.Button1);
            }
             }
```

```
                        // 0x22 Вкл/Откл реле
                        void BtnCmd22Click(object sender, EventArgs e)
                        {
                                try
                                {
                                        if (clientSocket.Connected)                     // если соединение
установлено
                        {
                          String message = "";
                          Byte[] mes = new Byte[4];          //переменная, которая будет содержать данные для
отправки

                          mes[0] = 0x22;
                          mes[1] = (byte)cmbCmd22_0.SelectedIndex;
                          mes[2] = (byte)cmbCmd22_1.SelectedIndex;
                          mes[3] = (byte)nmCmd22_2.Value;
                          message = "22 " + ByteToHex(mes[1]) + " " + ByteToHex(mes[2]) + " " +
ByteToHex(mes[3]);                  //сообщение для отображения в ListView

                          string DT = DateTime.Now.Hour.ToString("00") + ":" +
DateTime.Now.Minute.ToString("00") + ":" + DateTime.Now.Second.ToString("00") + "." +
DateTime.Now.Millisecond.ToString("000");
                          listViewUpdate(DT, "Команда", message, "Вкл/Откл реле");

                          serverStream.Write(mes, 0, 4);
                          serverStream.Flush();
                        }
                        else
                        {
                                MessageBox.Show("Соединение не установлено.", this.Text,
MessageBoxButtons.OK, MessageBoxIcon.Warning, MessageBoxDefaultButton.Button1);
                        }
                }
                                catch (Exception ex)
                {
                  MessageBox.Show(ex.Message, this.Text, MessageBoxButtons.OK, MessageBoxIcon.Error,
MessageBoxDefaultButton.Button1);
                }
                        }

                        // 0x23 Запросить состояние вх. и реле
                        void BtnCmd23Click(object sender, EventArgs e)
                        {
                                try
                                {
                                        if (clientSocket.Connected)                     // если соединение
установлено
                        {
                          String message = "";
                          Byte[] mes = new Byte[1];          //переменная, которая будет содержать данные для
отправки

                          mes[0] = 0x23;
                          message = "23";                                 //сообщение для отображения в ListView

                          string DT = DateTime.Now.Hour.ToString("00") + ":" +
DateTime.Now.Minute.ToString("00") + ":" + DateTime.Now.Second.ToString("00") + "." +
DateTime.Now.Millisecond.ToString("000");
                          listViewUpdate(DT, "Команда", message, "Запросить состояние вх. и реле");

                          serverStream.Write(mes, 0, 1);
                          serverStream.Flush();
                        }
```

```
                    else
                    {
                        MessageBox.Show("Соединение не установлено.", this.Text,
MessageBoxButtons.OK, MessageBoxIcon.Warning, MessageBoxDefaultButton.Button1);
                    }
                }
                    catch (Exception ex)
                {
                  MessageBox.Show(ex.Message, this.Text, MessageBoxButtons.OK, MessageBoxIcon.Error,
MessageBoxDefaultButton.Button1);
                }
                     }

            // 0x24 Значение аналогового входа
            private void btnCmd24_Click(object sender, EventArgs e)
            {
              try
              {
                if (clientSocket.Connected)                  // если соединение установлено
                {
                  String message = "";
                  Byte[] mes = new Byte[1];      //переменная, которая будет содержать данные для
отправки

                  mes[0] = 0x24;
                  message = "24";

                  string DT = DateTime.Now.Hour.ToString("00") + ":" +
DateTime.Now.Minute.ToString("00") + ":" + DateTime.Now.Second.ToString("00") + "." +
DateTime.Now.Millisecond.ToString("000");
                  listViewUpdate(DT, "Команда", message, "Запросить состояние вх. и реле");

                  serverStream.Write(mes, 0, 1);
                  serverStream.Flush();
                }
                else
                {
                  MessageBox.Show("Соединение не установлено.", this.Text, MessageBoxButtons.OK,
MessageBoxIcon.Warning, MessageBoxDefaultButton.Button1);
                }
              }
              catch (Exception ex)
              {
                MessageBox.Show(ex.Message, this.Text, MessageBoxButtons.OK, MessageBoxIcon.Error,
MessageBoxDefaultButton.Button1);
              }
            }
                    //        ПОТОК ПРИЕМА СООБЩЕНИЙ
                    //
                    private void getMessage()
            {
              while (true)
              {
                if (clientSocket.Connected)
                {
                  serverStream = clientSocket.GetStream();                    //получаем поток
                  int buffSize = 0;
                  int bytesRead= 0;
                  byte[] inStream = new byte[10025];                          //
инициализируем массив для приема данных
                  buffSize = clientSocket.ReceiveBufferSize;                  //получаем размер буфера
                  try
                  {
```

```
                    bytesRead = serverStream.Read(inStream, 0, buffSize);//считываем данные из потока
                }
                        catch(Exception er)
                        {
                           ;
                        }

                string DT = DateTime.Now.Hour.ToString("00") + ":" +
DateTime.Now.Minute.ToString("00") + ":" + DateTime.Now.Second.ToString("00") + "." +
DateTime.Now.Millisecond.ToString("000");
                //формируем строку с датой и временем получения данных
                string message = "";

                //формируем сообщения для вывода в ListView в 16-ричном виде
                for (int i = 0; i < bytesRead; i++)
                   message = message + ByteToHex(inStream[i]) + " ";

                // если данные получены - выводим их в ListView
                if (bytesRead > 0)
                {
                    switch (inStream[0])
                    {
                            case 0x01:
                                    listViewUpdate(DT, "Ответ", message, "Ответ на проверку связи");
                            break;
                            case 0x03:
                            if (bytesRead >= 2)
                            {
                                    switch (inStream[1])
                                    {
                                            case 1: listViewUpdate(DT, "Ответ", message, "Тип модуля:
Считыватель VRD-E"); break;
                                            case 2: listViewUpdate(DT, "Ответ", message, "Тип модуля:
Socket-2"); break;
                                            case 3: listViewUpdate(DT, "Ответ", message, "Тип модуля:
Socket-1"); break;
                                            case 4: listViewUpdate(DT, "Ответ", message, "Тип модуля:
Socket-3"); break;
                                    }
                            }
                            else
                                    listViewUpdate(DT, "Ответ", message, "Тип модуля");
                            break;
                            case 0x04:
                            if (bytesRead >= 3)
                                    listViewUpdate(DT, "Ответ", message, "Серийный номер модуля: " +
((int)inStream[1] * 256 + (int)inStream[2]).ToString());
                            else
                                    listViewUpdate(DT, "Ответ", message, "Серийный номер модуля");
                            break;
                            case 0x20:
                                    listViewUpdate(DT, "Ответ", message, "Настройки входа");
                            break;
                            case 0x21:
                                    listViewUpdate(DT, "Ответ", message, "Изменилось состояние
входа");
                            break;
                            case 0x22:
                                    listViewUpdate(DT, "Ответ", message, "Вкл/Откл реле");
                            break;
                        case 0x23:
                           listViewUpdate(DT, "Ответ", message, "Состояние входов и реле");
                           break;
```

```
                        case 0x24:
                            listViewUpdate(DT, "Ответ", message, "Значение аналогового входа");
                            break;
                        }
                    }
                }
            }
        }
```

```
        ////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
///
                        //
                        //          ОТКРЫТИЕ И ЗАКРЫТИЕ СОЕДИНЕНИЯ
                        //

                        void BtnConnectClick(object sender, EventArgs e)
                        {
                                try
            {
                int port = Int32.Parse(txtIPport.Text);                 //чтение из текстового поля номера порта
                if ((port > 64000) || (port < 20))                      //если это число и оно не между 20 и 64000
- выводим сообщение об ошибке
                {
                    MessageBox.Show("Разрешенный диапазон номера порта 20...64000.", this.Text,
MessageBoxButtons.OK, MessageBoxIcon.Warning, MessageBoxDefaultButton.Button1);
                    return;
                }
            }
            catch (Exception ex)
            {
                                        //если в текстовом поле не число - выводим сообщение об ошибке
                MessageBox.Show("Ошибка указания порта." + ex.Message, this.Text,
MessageBoxButtons.OK, MessageBoxIcon.Warning, MessageBoxDefaultButton.Button1);
                return;
            }

            //проверяем правильно ли введен IP-адрес в текстовое поле
            Match m = Regex.Match(txtIPaddress.Text, patternIP);
            if (!(m.Success))
            {
                MessageBox.Show("Не верно указан IP=адрес", this.Text, MessageBoxButtons.OK,
MessageBoxIcon.Warning, MessageBoxDefaultButton.Button1);
                return;
            }

            try
            {
                if (!clientSocket.Connected)
                {

                    clientSocket = new System.Net.Sockets.TcpClient();
                    clientSocket.Connect(txtIPaddress.Text, Int32.Parse(txtIPport.Text));
                    //подключаемся к IP и порту, введенных в текстовые поля
                    serverStream = clientSocket.GetStream();
                    //получаем поток
                    string DT = DateTime.Now.Hour.ToString("00") + ":" +
DateTime.Now.Minute.ToString("00") + ":" + DateTime.Now.Second.ToString("00") + "." +
DateTime.Now.Millisecond.ToString("000");
```

```
                    //формируем строку с датой и временем при котором произошло подключение и добавляем в ListView
                    listViewUpdate(DT, "", "", "Открыто соединение");
                    picConState.BackgroundImage = imgListConnection.Images["Conn_Green"];
                    lblConState.Text = "Открыто";

                    this.lstEvents.EnsureVisible(lstEvents.Items.Count - 1);
                    clientSocket.ReceiveBufferSize = 8192;
                    //выставляем размер буфера - 8192 байта
                    ctThread = new Thread(getMessage);
                    //создаем поток, в котором будет происходит прием данных и запускаем его
                    ctThread.Start();
                }
            }
            catch (Exception ex)
            {
                MessageBox.Show(ex.Message, this.Text, MessageBoxButtons.OK, MessageBoxIcon.Error, MessageBoxDefaultButton.Button1);
            }
                    }

                    void BtnDisconnectClick(object sender, EventArgs e)
                    {
                            try
                            {
                        if (clientSocket.Connected)
                        {
                            ctThread.Abort();

                            //формируем строку с датой отключения, добавляем ее в ListView и отключаемся
                            string DT = DateTime.Now.Hour.ToString("00") + ":" + DateTime.Now.Minute.ToString("00") + ":" + DateTime.Now.Second.ToString("00") + "." + DateTime.Now.Millisecond.ToString("000");
                            listViewUpdate(DT, "", "", "Закрыто соединение");
                        picConState.BackgroundImage = imgListConnection.Images["Conn_Red"];
                        lblConState.Text = "Закрыто";

                            clientSocket.Client.Disconnect(true);
                        }
                            }
                            catch (Exception ex)
            {
                MessageBox.Show(ex.Message, this.Text, MessageBoxButtons.OK, MessageBoxIcon.Error, MessageBoxDefaultButton.Button1);
            }
                    }




                    void LabelURLClick(object sender, EventArgs e)
                    {
                            System.Diagnostics.Process.Start("http://www.vkmodule.com.ua");
                    }

                    void TmrConStateTick(object sender, EventArgs e)
                    {
                            try
            {
                if (clientSocket.Connected)
                {
                            picConState.BackgroundImage = imgListConnection.Images["Conn_Green"];
```

```
                lblConState.Text = "Открыто";
            }
            else
            {
                picConState.BackgroundImage = imgListConnection.Images["Conn_Red"];
                lblConState.Text = "Закрыто";
            }
        }
        catch(Exception er)
        {
            ;
        }
                }

        }
}
```

ДОДАТОК Б – ПРЕЗЕНТАЦІЯ