

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»

Інститут інформатики та радіoeлектроніки,
Факультет комп'ютерних наук і технологій
(повне найменування інституту, назва факультету)

Кафедра програмних засобів
(повне найменування кафедри)

Пояснювальна записка

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

на тему РОЗРОБКА ПРОГРАМНО-АПАРАТНОГО ЗАБЕЗПЕЧЕННЯ ДЛЯ
МОДЕРНІЗАЦІЇ ЕКСПЕРИМЕНТІВ ВІДДАЛЕНОЇ ЛАБОРАТОРІЇ RELDES.
SOFTWARE AND HARDWARE DEVELOPMENT FOR UPDATING THE
RELDES REMOTE LABORATORY EXPERIMENTS

Виконав: студент(ка) 4 курсу, групи КНТ-217
Спеціальності 122 Комп'ютерні науки
(код і найменування спеціальності)

Освітня програма (спеціалізація)
Комп'ютерні науки

Троц Є. С.
(прізвище та ініціали)

Керівник Пархоменко А.В.
(прізвище та ініціали)

Рецензент Морщавка С.В.
(прізвище та ініціали)

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

Інститут, факультет ІРЕ, ФКНТ
 Кафедра програмних засобів
 Ступінь вищої освіти бакалавр
 Спеціальність 122 Комп'ютерні науки
 (код і найменування)
 Освітня програма (спеціалізація) Комп'ютерні науки
 (назва освітньої програми (спеціалізації))

ЗАТВЕРДЖУЮ

Завідувач кафедри ПЗ, д.т.н, проф.
С.О. Субботін
 “ ” 2021 року

З А В Д А Н Н Я

НА ДИПЛОМНИЙ ПРОЄКТ (РОБОТУ) СТУДЕНТА(КИ)

Троця Євгенія Сергійовича

(прізвище, ім'я, по батькові)

1. Тема проєкту (роботи) Розробка програмно-апаратного забезпечення для модернізації експериментів віддаленої лабораторії RELDES. Software and Hardware Development for Updating the RELDES Remote Laboratory Experiments.

керівник проєкту (роботи) Пархоменко Анжеліка Володимирівна, к.т.н., доцент,
 (прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом закладу вищої освіти від “30” березня 2021 року № 103

2. Строк подання студентом проєкту (роботи) 17 травня 2020 року

3. Вихідні дані до проєкту (роботи) рекомендована література, технічне завдання

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити) 1. Аналіз предметної області. 2. Проєктування програмно-апаратного забезпечення для модернізації ВЛ. 3. Конструювання програмно-апаратного забезпечення для модернізації ВЛ. 4. Керівництво програміста. 5. Керівництво користувача.

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень) Слайди презентації

6. Консультанти розділів проєкту (роботи)

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	прийняв виконане завдання
1-5 Основна частина	Пархоменко А.В., доцент		
Нормоконтроль	Дейнега Л.Ю., ст. викладач		

7. Дата видачі завдання “28” лютого 2020 року.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломного проєкту (роботи)	Строк виконання етапів проєкту (роботи)	Примітка
1	Постановка завдання роботи	1 тиждень	Завдання, ТЗ
2	Аналіз предметної області	2-3 тижні	Розділ 1
3	Розробка архітектури програми	4 тиждень	Розділ 2
4	Розробка програмного та апаратного забезпечення	5-6 тижні	Розділ 3
5	Тестування та експериментальне дослідження програми	7 тиждень	Розділ 3
6	Оформлення пояснювальної записки та документів до неї. Нормоконтроль та рецензування	8 тиждень	Розділи 4,5. Додатки
7	Захист роботи	9 тиждень	

Студент(ка)

(підпис) Троц Є.С.
(прізвище та ініціали)

Керівник проєкту (роботи)

(підпис) Пархоменко А.В.
(прізвище та ініціали)

РЕФЕРАТ

Пояснювальна записка до дипломної кваліфікаційної роботи бакалавра: 118 с., 58 рис., 12 табл., 3 дод., 46 джерел.

ВІДДАЛЕНА ЛАБОРАТОРІЯ, ЕКСПЕРИМЕНТ, МОДЕРНІЗАЦІЯ, ARDUINO, RASPBERRY PI, РОЗУМНА ТЕПЛИЦЯ, ПРОГРАМНО-АПАРАТНЕ ЗАБЕЗПЕЧЕННЯ, ДАТЧИК, МОДУЛЬ.

Об'єкт дослідження – процес модернізації віддаленої лабораторії (ВЛ) RELDES.

Предмет дослідження – методи та засоби модернізації ВЛ RELDES.

Мета роботи – модернізація програмно-апаратного забезпечення ВЛ RELDES для розширення функціональних можливостей та підвищення ефективності її використання в навчальному процесі.

Матеріали, методи та технічні засоби: об'єктно-орієнтоване програмування, мова програмування C++, Arduino IDE, Arduino Mega, Arduino Nano, NodeMCU, мінікомп'ютер Raspberry Pi 3, платформа OpenHAB, актуатори, датчики.

Результати. Проведено модернізацію існуючої інфраструктури ВЛ за рахунок заміни програмних та апаратних компонентів та додавання нового експерименту Розумна теплиця.

Висновки. Впровадження розробленого програмно-апаратного забезпечення дозволить розширити функціональні можливості ВЛ та підвищити ефективність її використання в навчальному процесі.

Галузь використання – віддалені лабораторії, системи домашньої автоматизації.

ABSTRACT

Explanatory note for the qualifying work of the bachelor: 118 pages, 58 figures, 12 tables, 3 appendixes, 46 sources.

REMOTE LABORATORY, EXPERIMENT, MODERNIZATION, ARDUINO, RASPBERRY PI, SMART GREENHOUSE, SOFTWARE, SENSOR, MOD.

Object of study is the process of modernization of the remote laboratory RELDES.

Subject of study are methods and means of modernization of the remote laboratory RELDES.

The goal of the work is to modernize the software and hardware of remote laboratory RELDES to expand functionality and increase efficiency of its use in educational process.

Material, methods and technical means: object-oriented programming, C++ programming language, Arduino IDE, Arduino Mega, Arduino Nano, NodeMCU, Raspberry Pi 3 minicomputer, OpenHUB platform, actuators, sensors.

Results. Modernization of the existing remote laboratory infrastructure by replacing software and hardware components and adding a new smart greenhouse experiment.

Conclusions. Introduction of the developed software and hardware will expand the functionality of remote laboratory and increase efficiency of its usage in educational process.

Field of application are remote laboratories, home automation systems.

ЗМІСТ

	С.
Перелік скорочень та умовних познач.....	8
Вступ.....	9
1 Аналіз предметної області.....	10
1.1 Загальні відомості про віддалені та віртуальні лабораторії.....	10
1.2 Огляд функціональних можливостей існуючих віддалених лабораторій.....	12
1.3 Опис відомостей про системи домашньої автоматизації.....	25
1.4 Опис особливостей реалізації ВЛ.....	28
1.5 Аналіз проблем та недоліків віддаленої лабораторії.....	33
1.6 Постановка задач дипломної роботи.....	38
2 Проєктування програмно-апаратного забезпечення для модернізації ВЛ.....	40
2.1 Аналіз вимог до проєкту.....	40
2.2 Архітектура системи.....	41
2.3 Вибір апаратних компонентів.....	43
2.4 Вибір середовища розробки ПЗ.....	51
2.5 Висновки.....	61
3 Конструювання програмно-апаратного забезпечення для модернізації ВЛ.....	63
3.1 Алгоритми функціонування.....	63
3.2 Опис особливостей апаратної реалізації.....	67
3.3 Опис особливостей програмної реалізації.....	71
3.4 Висновки до розділу.....	81
4 Керівництво програміста.....	82
4.1 Призначення та умови застосування програмно-апаратного забезпечення.....	82
4.2 Характеристики програми.....	82
4.3 Звернення до програми.....	83
4.4 Вхідні та вихідні дані.....	84

4.5 Повідомлення.....	84
5 Керівництво користувача.....	87
5.1 Призначення програмно-апаратного забезпечення.....	87
5.2 Умови виконання програм.....	87
5.3 Виконання програм.....	88
5.4 Повідомлення.....	90
Висновки.....	91
Перелік джерел посилання.....	92
Додаток А Технічне завдання.....	96
Додаток Б Тексти програм.....	100
Додаток В Слайди презентації.....	111

ПЕРЕЛІК СКОРОЧЕНЬ ТА УМОВНИХ ПОЗНАК

API	– Application Programming Interface;
FSM	– Finite State Machine;
GUI	– Graphical User Interface;
IDE	– Integrated Development Environment;
PLC	– Programmable Logic Controller;
REIoT	– Remote Laboratory for Design of Embedded Systems and Smart House & Internet of Things;
RELDES	– Remote Laboratory for Design of Embedded Systems;
RTU	– Remote Terminal Unit;
SoC	– System-on-Chip;
TCP	– Transmission Control Protocol;
АЦП	– аналого-цифровий перетворювач;
BIC	– велика інтегральна схема;
BC	– вбудована система;
МК	– мікроконтролер;
ОС	– операційна система;
ПЗ	– програмне забезпечення;
ПК	– персональний комп'ютер;
ПЛІС	– програмована логічна інтегральна схема;
ЦАП	– цифро-аналоговий перетворювач.

ВСТУП

Останнім часом спостерігається тенденція віддаленого використання різноманітних технологій, зокрема зберігання інформації у віддалених базах даних, хмарні обчислення та ін. Окрему нішу займають віддалені лабораторії. Віддалена лабораторія дозволяє в реальному часі побачити, як реальні пристрої виконують поставлені задачі. Це підвищує наочність навчання та надає можливість оримати практичні навички роботи з реальними пристроями навіть в умовах дистанційного навчання. Результат дослідження зазвичай передається у вигляді потокового відео.

На сьогоднішній день вже відомо багато популярних віддалених лабораторій, серед яких виділяється лабораторія REIoT, яка об'єднує лабораторію проведення експериментів з розробки вбудованих систем RELDES та лабораторію домашньої автоматизації Smart House & IoT. Віддалена лабораторія RELDES дозволяє провести 4 експерименти на основі платформ Arduino, до яких підключені такі модулі як серводвигуни, ультразвуковий датчик відстані, дисплеї та світлодіоди. Робота виконується шляхом завантаження коду через відповідну форму на сайті лабораторії та спостереження за виконанням через вебкамеру. Віддалена лабораторія Smart House & IoT включає в себе макет розумного будинку, що містить основні підсистеми для управління освітленням, мікрокліматом, генерацією та споживанням електроенергії, доступом, безпекою та ін.

Аналіз показав, що програмне та апаратне забезпечення окремих експериментів лабораторії потребує модернізації. Окрім цього було виявлено, що відсутній експеримент Розумна теплиця. Тому, тема роботи є актуальною.

Мета роботи – модернізація програмно-апаратного забезпечення ВЛ RELDES для розширення функціональних можливостей та підвищення ефективності її використання в навчальному процесі.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Загальні відомості про віддалені та віртуальні лабораторії

Згідно з проведеними дослідженнями, віртуальна лабораторія – це програмно-апаратний комплекс, що забезпечує здійснення проведення експериментів у віртуальному середовищі як в умовах локальної мережі класу університету, так і віддалено через мережу Internet. Така можливість підкреслює методичну та економічну доцільність цієї форми навчання [1].

На відміну від віртуальної, як показав аналіз, ВЛ передбачає роботу з реальними фізичними компонентами та забезпеченням. Лабораторії з віддаленим доступом не лише дублюють лабораторний практикум на звичайній очній формі, а і дозволяють працювати з унікальним та дорогим забезпеченням та ставити реальні експерименти віддалено через мережу Internet у реальному часі з будь-якої точки земної кулі. Спостереження також надається за допомогою камер, що передають потокове відображення лабораторії та дозволяють бачити виконання експериментів [2].

Загалом, модель – це штучно створений елемент у вигляді схеми фізичних конструкцій за допомогою математичних форм або формул, що як і досліджуваний об'єкт відображає та повторює в спрощеному вигляді структуру, необхідні властивості та зв'язки й відношення між компонентами цього об'єкта. Тому модель продовжує вже існуючі правила і закони, а у реальній системі завжди є ймовірність виникнення невизначеностей та відхилень, виявлення нових взаємозв'язків системи та їх властивостей. Моделювання зазвичай представляє ідеальну ситуацію, уникаючи більшість складнощів реального світу, таких як шум і точність показань, що є досить важливим в реальному світі при звичайних фізичних експериментах [3].

Деколи саме увага до таких випадків відхилень або неточностей з теоретичною базою може бути джерелом нових знань, адже могло бути пропущеним іншими дослідниками. Тому створена руками іншого дослідника модель зазвичай є позбавленою цієї ймовірності виникнення

відхилень, які, якщо й виникають, є зазвичай показником неточності моделі при побудові та не може бути розглянута як можливість нових властивостей та взаємозв'язків елементів [4].

Таким чином, ВЛ – це використання зв'язку для дистанційного або локального проведення, на відміну від віртуальних, реальних експериментів в реально існуючому фізично місці розташування функціонуючої лабораторії, при цьому дослідник використовує дане забезпечення, не знаходячись безпосередньо в цій лабораторії. ВЛ включає в себе як один, так і декілька дослідів [5].

Згідно з проведенням аналізом, ВЛ можуть бути також і гібридними, тобто надають можливість проводити експерименти як з моделями, так і з реальними фізичними пристроями. Таким чином гібридні онлайн-лабораторії мають безліч переваг.

Досить зручною відмінністю від віддалених є можливість усунення черги, адже якщо з реальними приладами може працювати лише один дослідник в конкретний момент часу, то з моделлю може працювати безліч.

Різновиди онлайн-лабораторій показано на рисунку 1.1.

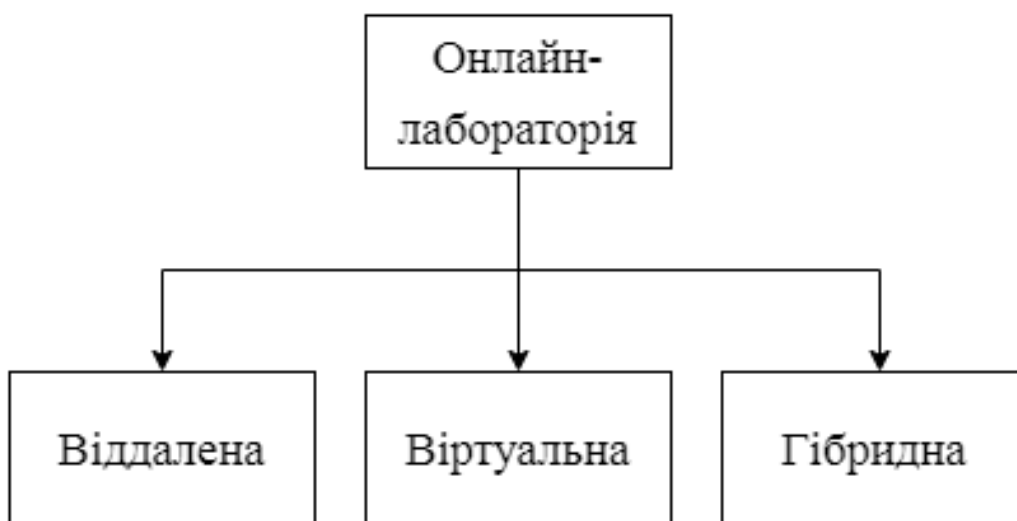


Рисунок 1.1 – Різновиди онлайн-лабораторій

Також це дозволяє підготуватися на моделі, щоб потім, коли настане момент випробовувати реальний прилад віддалено, можна було швидко поставити дослід та отримати дані, не створюючи довгої черги.

Серед переваг можна виокремити можливість оперативного усунення проблем, що іноді виникають з реальними пристроями. В такому разі експериментатори зможуть не відкладати проведення дослідів, а провести його на моделі.

1.2 Огляд функціональних можливостей існуючих віддалених лабораторій

Загалом, ВЛ використовуються зазвичай в навчанні та проведенні певних досліджень. Вони мають певні переваги:

- розвантаження часу для лаборантів та викладачів шляхом збільшення часу дослідження кожним студентом, якщо є досить високий потік студентів;
- розмиття обмежень фізичного розташування завдяки технологіям віддаленого доступу;
- покращення результатів шляхом можливості повторного виконання задля усунення похибок при проведенні;
- покращення результатів реального фізичного дослідів в віддаленій лабораторії шляхом попередніх дослідів в віддаленій лабораторії;
- відсутність ризику виникнення надзвичайної ситуації у якості помилки при проведенні дослідів, що значно впливає на безпеку;
- підвищення загальної доступності до лабораторій, адже наразі не кожен може прийняти участь у експериментах;
- зменшення витрат на фінансування до 50%, адже зменшуються витрати на персонал та інфраструктуру лабораторії, що робить віддалені лабораторії досить економними;

- підвищення результатів навчання, що підвищує загальну якість навчання;
- дозволяє здійснювати обмін знаннями та досвідом [6].

Проте з цього й випливають недоліки. Загалом, в порівнянні зі звичайною фізичною лабораторією є два недоліки:

- відсутність досвіду усунення неполадок і налагодження;
- відсутність досвіду налаштування обладнання [6].

Далі наведено короткий огляд функціональних можливостей існуючих ВЛ.

Однією з таких лабораторій є LabsLand. LabsLand з'єднує школи, університети та інші навчальні заклади зі справжніми фізичними лабораторіями, до яких є доступ через мережу Інтернет. Лабораторія може бути або роботом з живленням від Arduino в Іспанії, або кінематичною установкою в Бразилії чи навіть лабораторією з тестування радіоактивності в Австралії. Ці лабораторії є матеріальним представленням, а не математичними моделями: лабораторії фізично існують, а тому студенти мають до них реальний віддалений доступ [7].

Приклад експерименту з роботом ВЛ LabsLand зображено на рисунку 1.2 [8]. Рухи робота проводяться на спеціальній платформі, спостерігати за поведінкою якого надається через вебкамеру.

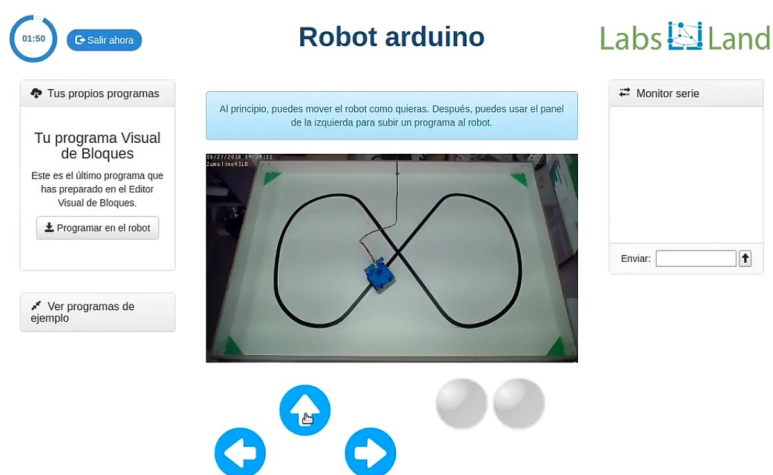


Рисунок 1.2 – Приклад експерименту з роботом ВЛ LabsLand [8]

В ВЛ надається можливість проводити експерименти з реальним роботом, запрограмувавши мікроконтролер Arduino та завантаживши свою програму в робота, який буде виконувати певні завдання. Спостерігати за поведінкою дозволяється через вебкамеру. Серед можливостей є здатність робота уникати перешкоди, брати участь в перегонках, що проходять по лінії та знаходити вихід з лабіринту. В найближчому майбутньому буде надана можливість боротися з іншими роботами [8].

Тестування радіоактивності дозволяє дослідити кількість часток, що випромінюються різними радіоактивними матеріалами та яка їхня кількість аналізується лічильником Гейгера. Надається можливість змінювати відстань між зразком і лічильником, а також час експозиції. Також можна помістити матеріал між зразком і лічильником та подивитися, як це вплине на вимірювання [9].

Приклад експерименту лабораторії LabsLand з дослідження радіоактивності зображено на рисунку 1.3 [9].

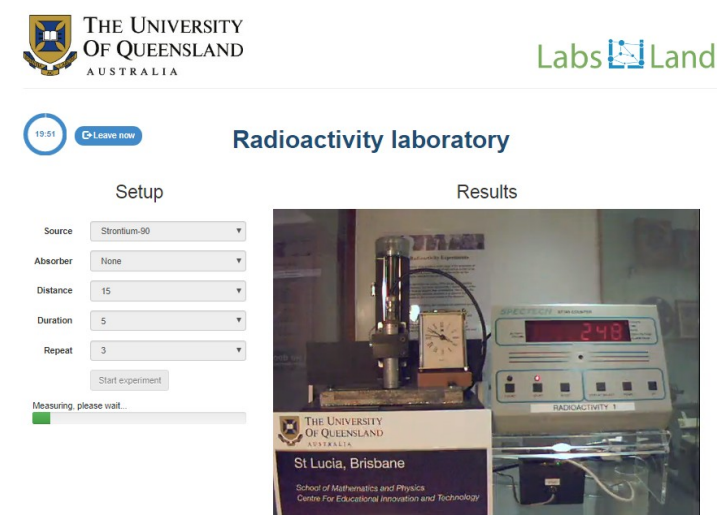


Рисунок 1.3 – Приклад експерименту з дослідження радіоактивності ВЛ LabsLand [7]

Робота у віддаленій лабораторії LabsLand виконується з використанням середовища Arduino IDE шляхом завантаження коду в реальний пристрій – робот (рис. 1.4) [8].

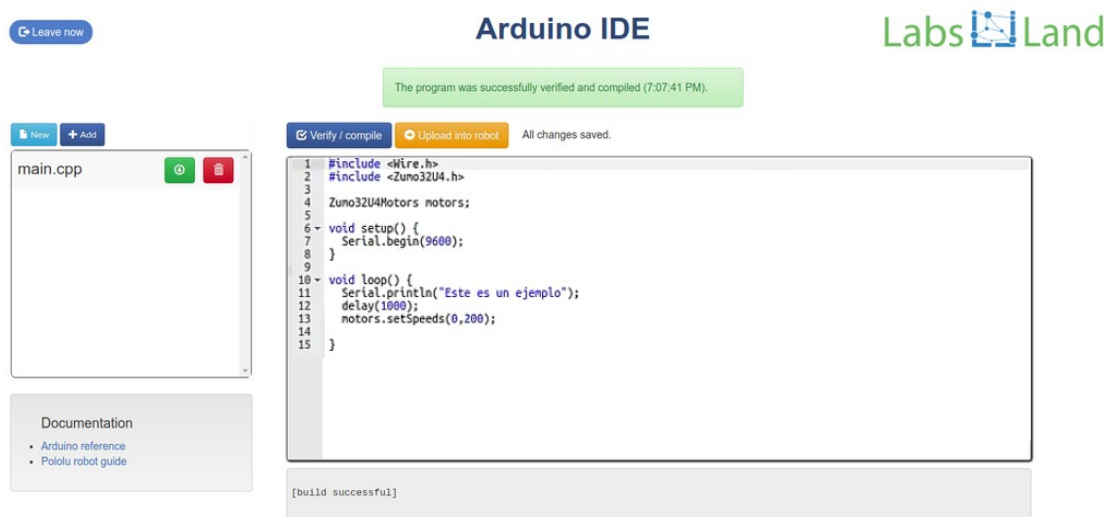


Рисунок 1.4 – Приклад інтерфейсу завантаження коду до ВЛ LabsLand [8]

Окрім цього, за бажанням можливо використовувати мову віртуального програмування схожу на scratch, а також надається можливість використовувати блочну мову Blockly. При цьому створений візуально алгоритм одразу конвертується в звичайний програмний код на мові C++. Приклад інтерфейсу програмування мовою Blockly зображено на рисунку 1.5 [10].

В цій лабораторії для роботи з такими програмованими логічними інтегральними схемами (ПЛІС) як Terasic DE2-115 і Terasic DE1-SoC відбувається за допомогою двох мов проектування: VHDL та Verilog. Кожна ПЛІС має набір вже встановлених компонентів, наприклад 10 світлодіодів та семисегментних дисплеїв. Крім того є доступ до десяти перемикачів і чотирьох віртуальних кнопок, маніпулюючи якими можна проаналізувати взаємодію з реальним обладнанням. Тому у лабораторії не потрібно установлювання жодного програмного і апаратного забезпечення на комп'ютері [10].

Приклад інтерфейсу експерименту з програмування ПЛІС зображено на рисунку 1.6 [10].

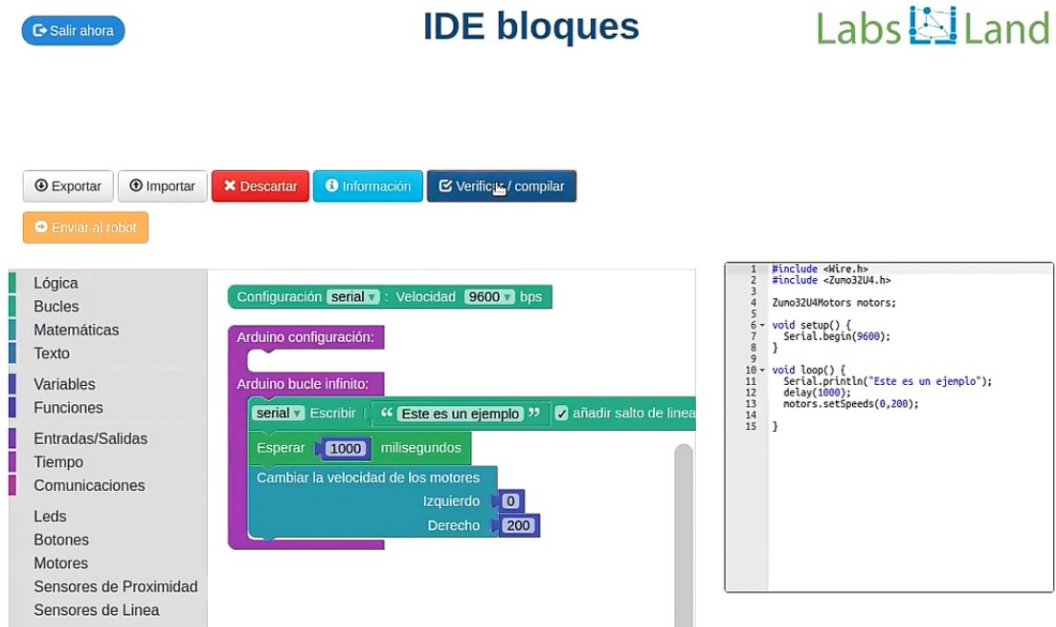


Рисунок 1.5 – Приклад інтерфейсу програмування ВЛ LabsLand [10]

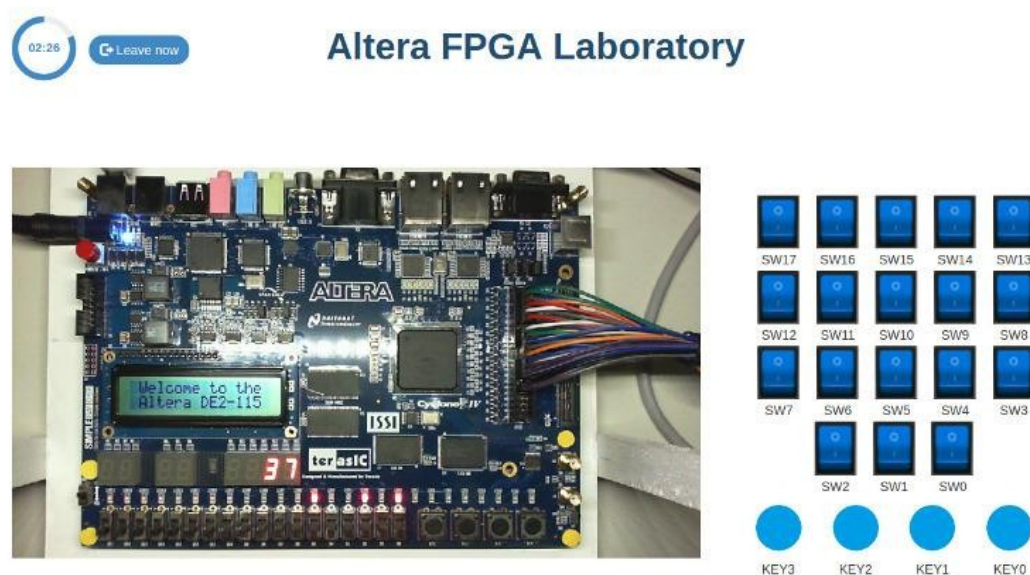


Рисунок 1.6 – Приклад інтерфейсу програмування ПЛІС віддаленої лабораторії LabsLand [10]

Віддалена лабораторія NetLab розробляється як частина Університету Південної Австралії. NetLab – це віддалена онлайн-лабораторія, яку викладачі використовують для навчання і демонстрацій, а

студенти – для віддаленого проведення експериментів на реальному лабораторному устаткуванні [11].

Застосунок надає панель GUI, що виглядає як справжні лабораторні інструменти. Взаємодія відбувається через кнопки, повороти ручки за допомогою миші, взаємодіючи з нею, як з реальним пристроєм. Приклад GUI лабораторії NetLab зображено на рисунку 1.7 [11].

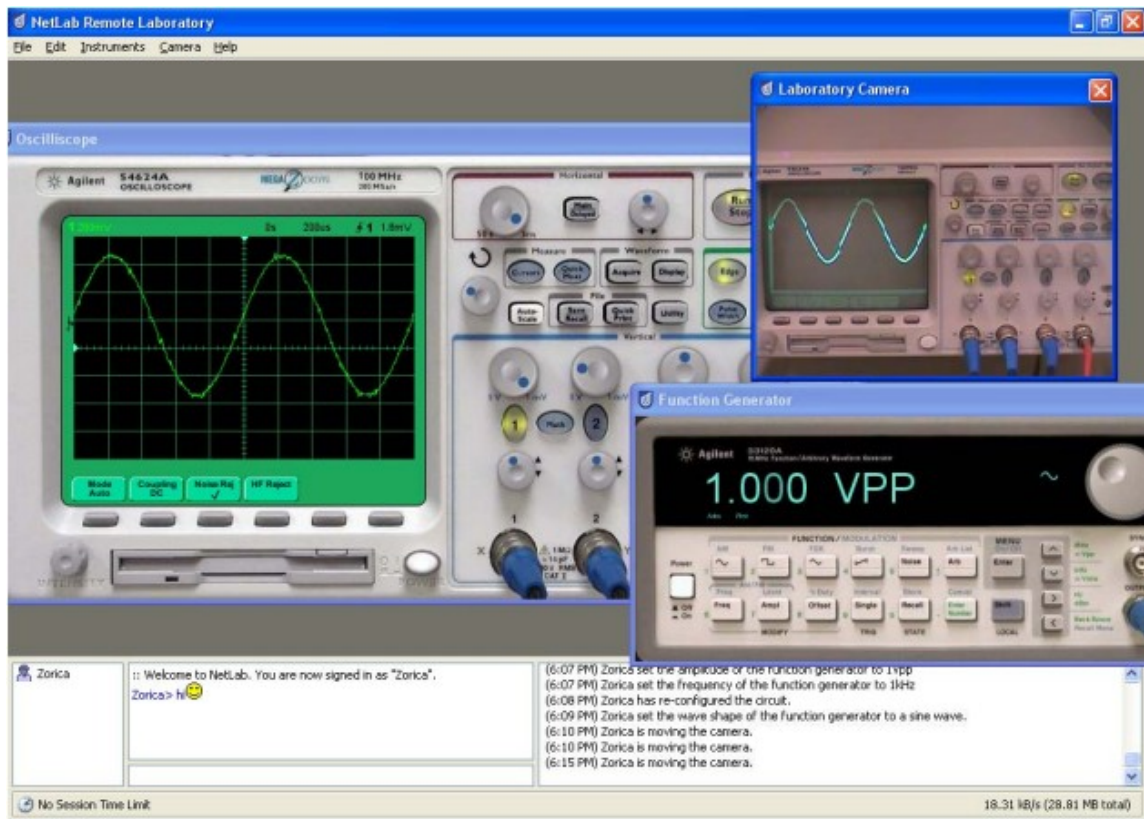


Рисунок 1.7 – Приклад GUI лабораторії NetLab [11]

Особливість NetLab в тому, при взаємодії з GUI, фактично відбувається робота над реальними приладами, що знаходяться в лабораторії в віддаленому місці. Взаємодія відбувається через Інтернет. Тому результат з екрану, є не симуляцією, а фактичними даними, що зчитуються з реальних приладів в реальному часі [12].

Загалом, NetLab в своєму асортименті містить безліч експериментів з аналізу RC кіл, роботу зі змінним струмом та послідовним резонансним колом.

ВЛ NetLab побудована на програмній платформі Java, що надає доступ до GUI, візуалізації графіки, функції доступу в мережу Інтернет і т.д., незалежно від операційної системи. Тому для роботи програми ВЛ NetLab на ПК має бути встановлена Java SE 6 або більш пізньої версії [12].

NetLab також використовує технологію Java Web Start, що дозволяє запускати Java-застосунки прямо з браузера. Таким чином, робота з ВЛ виконується у відповідній програмі, приклад якої зображено на рисунку 1.8 [12].

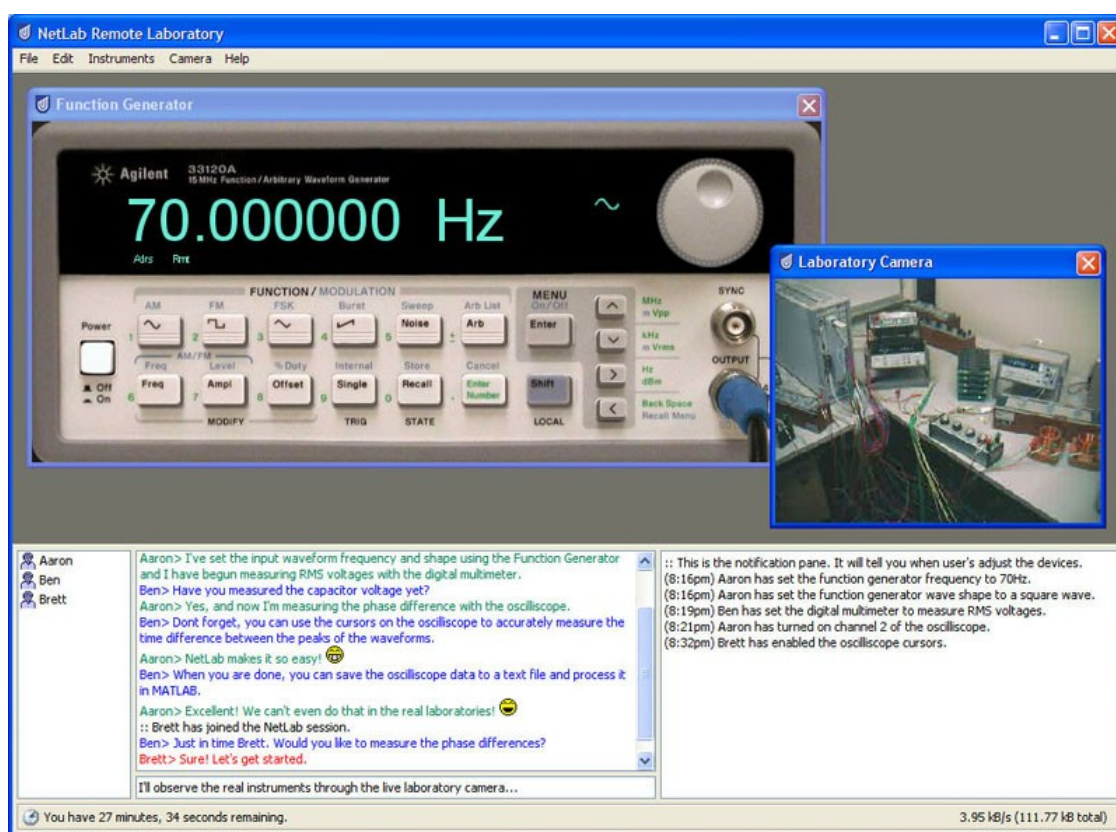


Рисунок 1.8 – Приклад інтерфейсу прикладної програми ВЛ NetLab [12]

Як видно з рисунку 1.8, інтерфейс виглядає як у віртуальній лабораторії, проте NetLab – це не просто симуляція. Користувачі взаємодіють з реальними приладами виконують реальні замірювання, проте відбувається це віддалено. При натисненні віртуальної кнопки інтерфейсу NetLab, відповідний фізично реальний лабораторний прилад одразу реагує, ніби була натиснута реальна кнопка на панелі. Оновлення зображень на

дисплеї виконується в міру того, як користувач використовує інструменти, як в реальному експерименті в реальному часі [13].

До складнощів ВЛ можна віднести складність налаштування схеми, коли дослідник фізично не присутній в лабораторії. В лабораторії NetLab проблему усувають шляхом використання інноваційної функції Circuit Builder, приклад зображено на рисунку 1.9 [13].

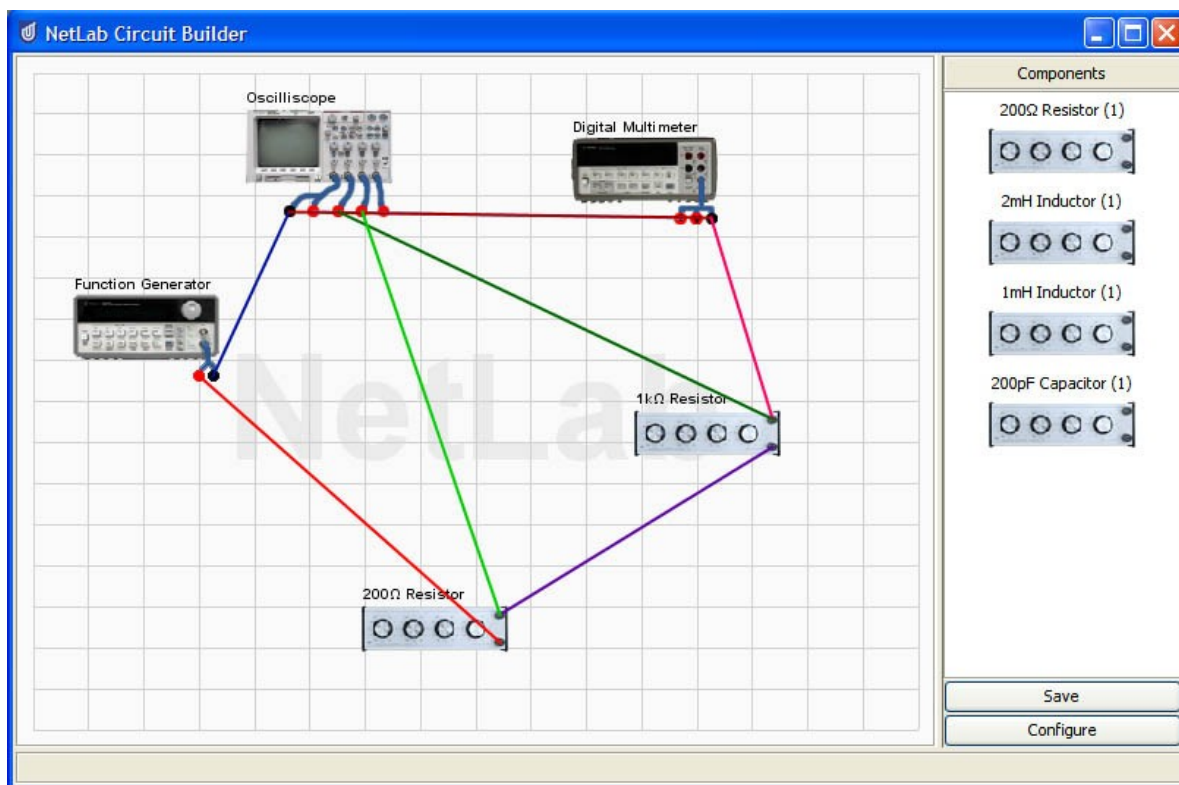


Рисунок 1.9 – Приклад інтерфейсу використання технології Circuit Builder ВЛ NetLab [13]

Лабораторія обладнана вебкамерами, що мають досить гарне масштабування, використовуючи які можна в реальному часі відслідковувати роботу реальних приладів під час роботи з ними. Камери мають двигуни, що дозволяють керувати ними, виконувати панорамування, нахиляти їх, збільшувати та зменшувати масштаб, зчитувати показники з дисплеїв, спостерігати за інструментами та бачити всі налаштування віддаленої лабораторії NetLab.

Однією з цікавих лабораторій, як показав аналіз, є віддалена лабораторія Goldi-Labs. Дана лабораторія реалізує як віртуальну модель, так і мережу віддалених лабораторій в певних країнах світу. Серед них Німеччина, Вірменія, Азербайджан, Грузія, Австралія та Україна.

На рисунку 1.10 зображено приклад проведення експерименту над віртуальною моделлю та реальним фізичним пристроєм лабораторії Goldi-Labs [14].

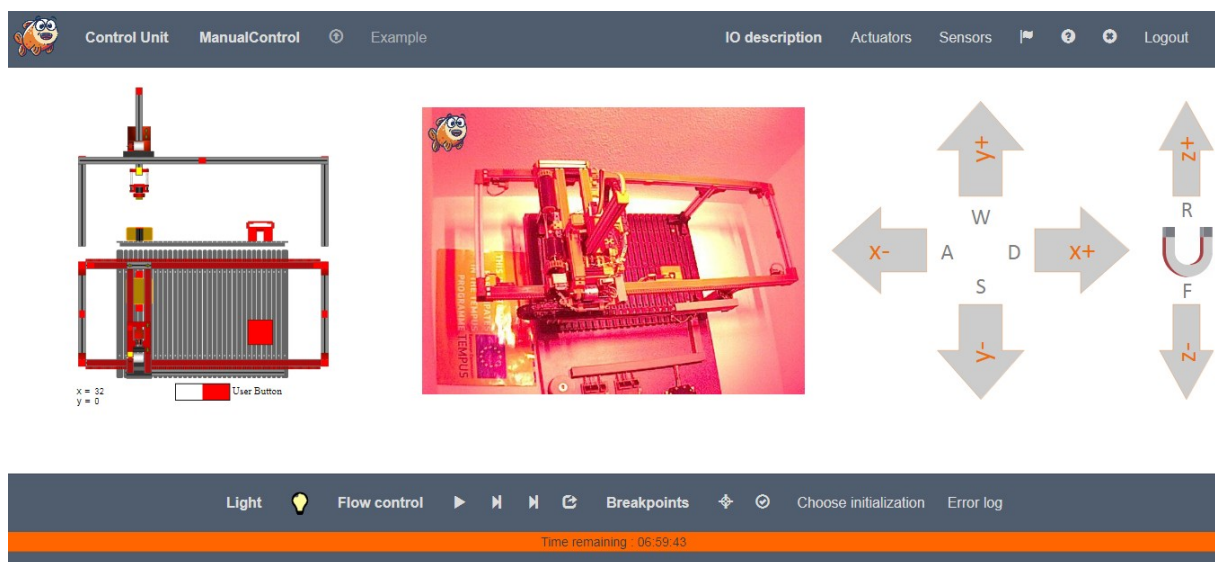


Рисунок 1.10 – Приклад проведення експерименту лабораторії Gold [14]

Загалом, лабораторія Goldi-Labs надає доступ до таких експериментів:

- 3-осьовий маніпулятор;
- 3 моделі ліфту;
- виробничий конвеєр;
- склад зберігання;
- цифрова демонстраційна плата швидкого прототипування [14].

В майбутньому буде також додано експеримент контроль рівня води та лабіринт [14].

Маніпуляції з такими експериментами, як 3-осьовий маніпулятор, модель ліфту, виробничий конвеєр та склад зберігання проводяться декількома способами.

Найпростіший спосіб це звісно ручне управління через візуальний інтерфейс користувача на сайті. З його допомогою можна виконувати базові маніпуляції, наприклад виклик ліфту на певний поверх або переміщення елемента по конвеєру. Приклад інтерфейсу зображено на рисунку 1.11.

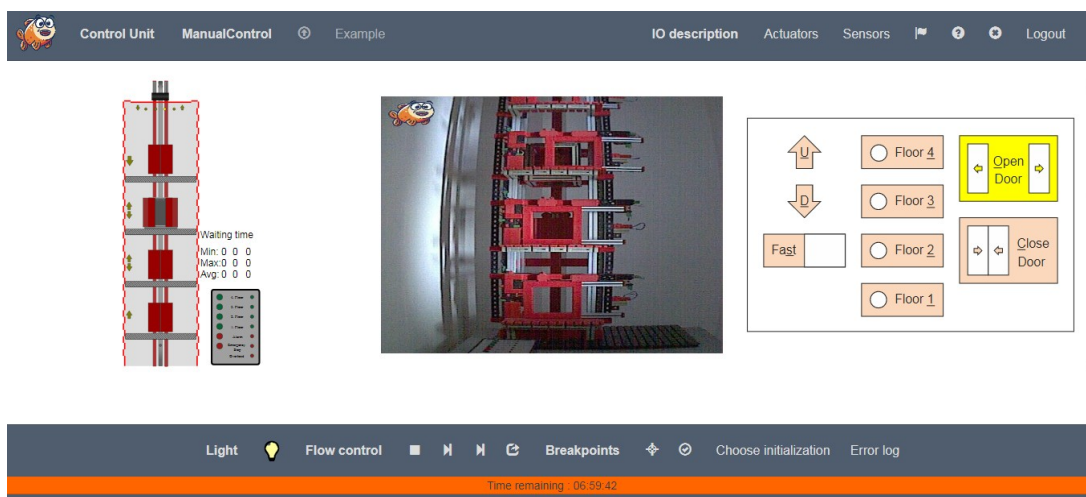


Рисунок 1.11 – Приклад інтерфейсу ручного управління віддаленої лабораторії Goldi.

Проте даним способом не можна створювати сценарії або скрипти, як наприклад за допомогою кінцевих автоматів (Finite State Machine, FSM). Кінцеві автомати є моделлю представлення, яка включає в себе безліч станів та переходів між ними. Таким чином кінцеві автомати описують алгоритмічну поведінку цифрових систем. Відповідно до поточного стану, система може реагувати на один конкретний вхід декількома різними виходами.

В віддаленій лабораторії Goldi робота зі кінцевими автоматами може проводитися за допомогою діаграм стану, таблиць переходів та у вигляді математичних формул станів вузлів та умов переходу між ними. Приклади зображено на рисунках 1.12 – 1.14 [15].

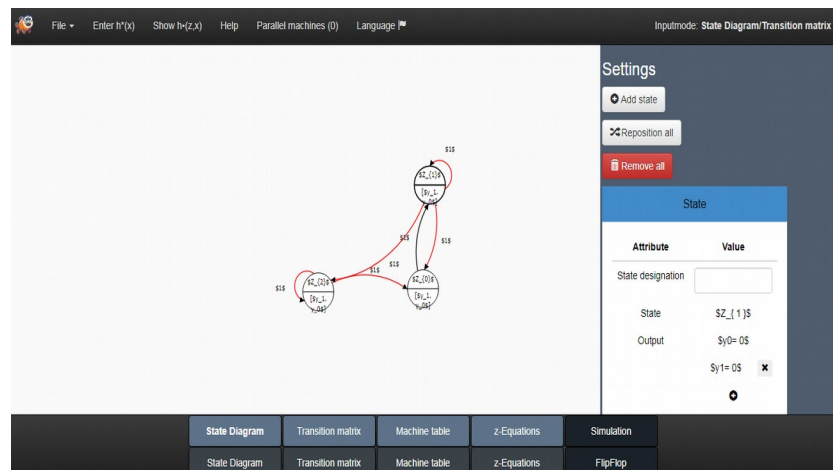


Рисунок 1.12 – Приклад інтерфейсу управління кінцевими автоматами віддаленої лабораторії Goldi за допомогою діаграм станів [15]

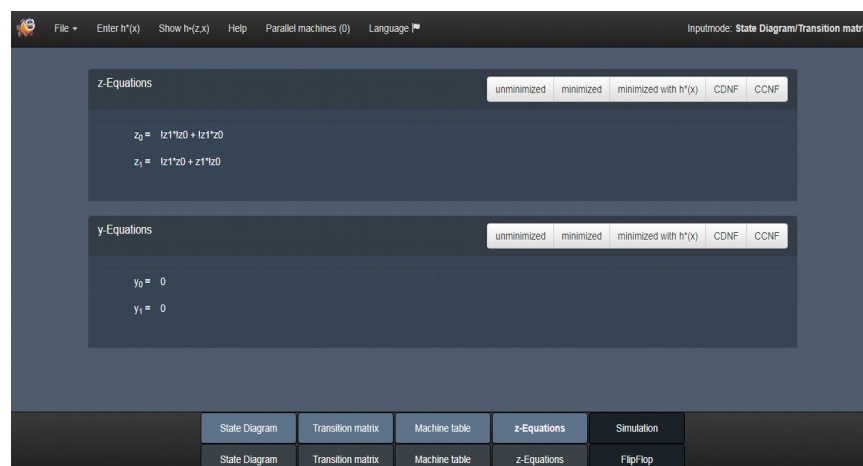


Рисунок 1.13 – Приклад інтерфейсу управління кінцевими автоматами віддаленої лабораторії Goldi за допомогою формул станів вузлів та умов переходу [15]

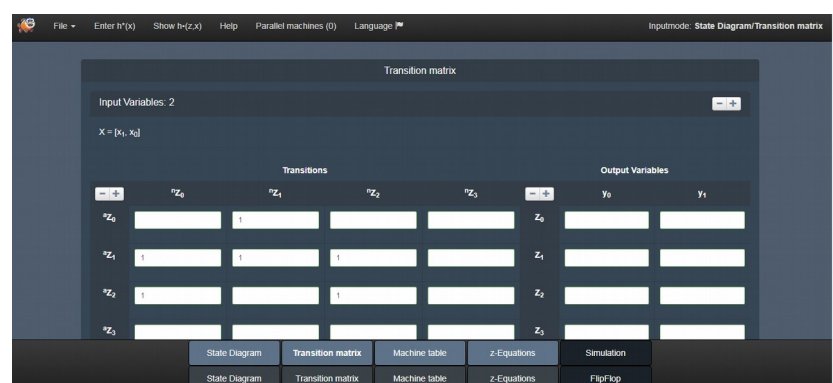


Рисунок 1.14 – Приклад інтерфейсу управління кінцевими автоматами віддаленої лабораторії Goldi за допомогою таблиць переходів [15]

Незалежно від обраного способу формування кінцевого автомату, буде створено файл формату GIFT, який необхідно завантажити до конкретного експерименту і призначити змінні автомату датчикам та виконавчим механізмам системи, як зображено на рисунку 1.15 [15].

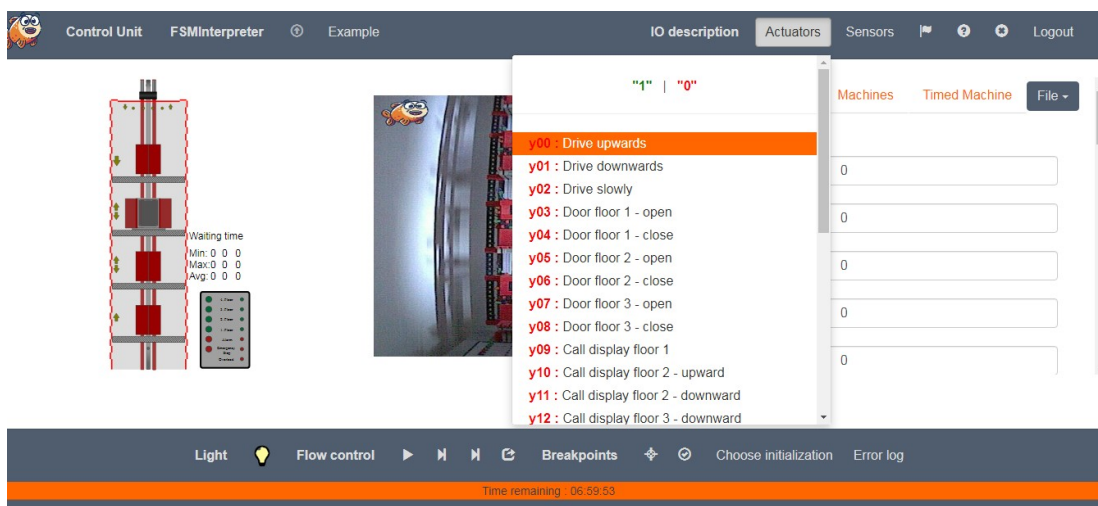


Рисунок 1.15 – Приклад налаштування для імпортованої машини в експерименті віддаленої лабораторії Goldi [15]

Після цього можна запускати експеримент та спостерігати за його виконанням за допомогою камери, що передає потокове зображення проведення експерименту [15].

Для проведення експериментів також використовують програмний застосунок Quartus II від ALTERA для програмування ПЛІС. Алгоритм управління можна задати використовуючи простий опис мовою VHDL, граф автомату з використанням вбудованого редактору FSM та схематичний дизайн. Загалом, Quartus II – це програмне забезпечення для програмування ПЛІС. VHDL – це високорівнева мова опису апаратну інтегральних схем. VHDL є базовою мовою розробки апаратів сучасних обчислювальних систем [16].

Після компіляції в Quartus II буде отримано файл виводу формату *.prof. Потім необхідно завантажити його в експеримент і запустити на

виконання. Приклад програмування мовою VHDL в Quartus II зображено на рисунку 1.16 [16].

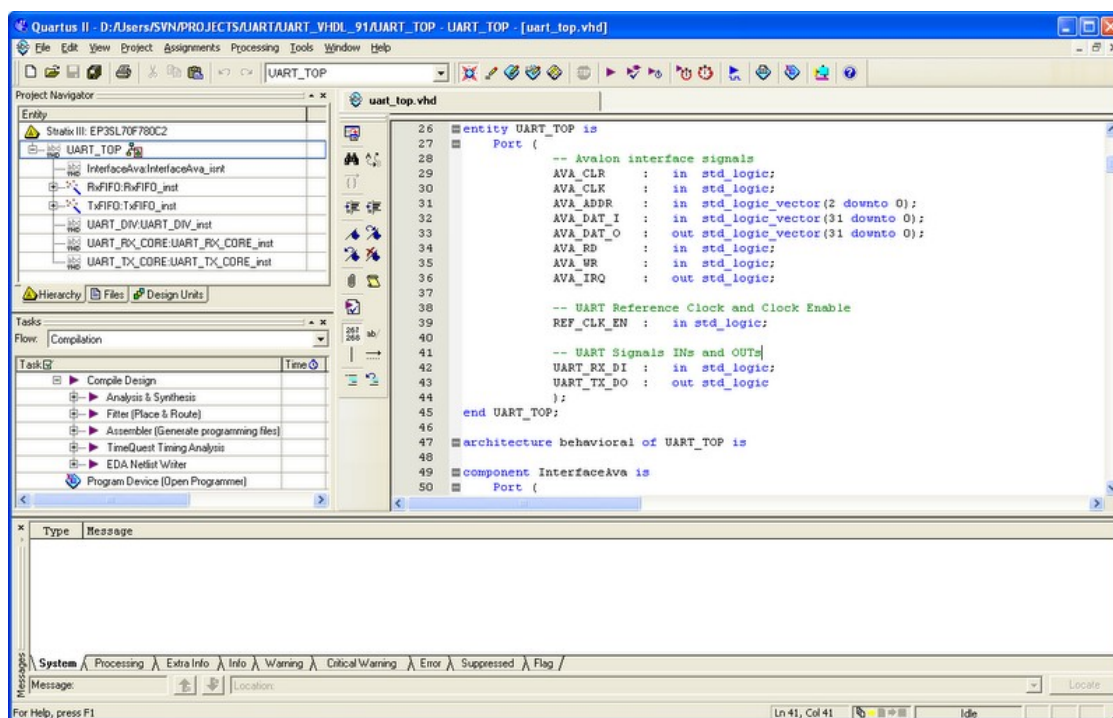


Рисунок 1.16 – Приклад програмування мовою VHDL в Quartus II віддаленої лабораторії Goldi [16]

Також Quartus II дозволяє програмувати демонстраційну плату швидкого прототипування Terasic DE2-115 і Terasic DE1-SoC. Робота з ними відбувається подібно до 3-осьового маніпулятора, моделі ліфту і т.д [17].

Окрім програмного забезпечення Quartus II, віддалена лабораторія Goldi дозволяє запрограмувати мікроконтролер Atmega644a від ATMEL використовуючи такий застосунок, як Atmel Studio. Atmel Studio – це безкоштовне інтегроване середовище, засноване на Visual Studio, для розробки забезпечення для 8- та 32-бітних мікроконтролерів сімейств AVR, а також 32-бітних мікроконтролерів сімейств ARM від компанії Atmel. Atmel Studio дозволяє писати програми мовою C, C++ та мовою асемблера. Після компіляції буде отримано файл, як і при роботі з Quartus II, проте формат буде *.hex. Далі необхідно так само завантажити його в експеримент

і запустити на виконання. Приклад програмування мовою C++ в Atmel Studio зображено на рисунку 1.17 [18].

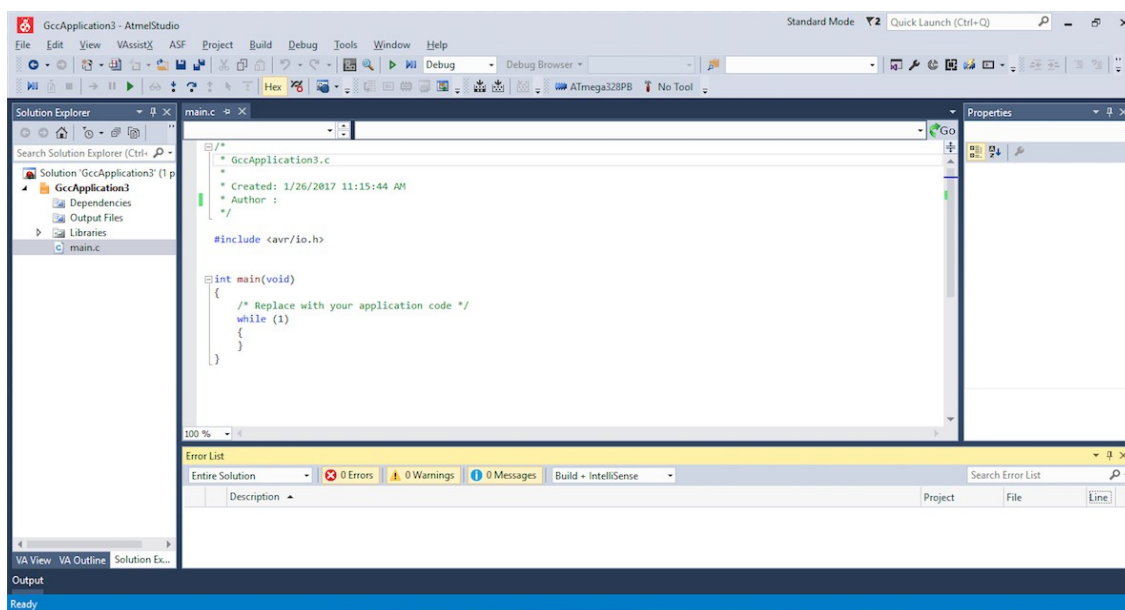


Рисунок 1.17 – Приклад програмування мовою C++ в Atmel Studio віддаленої лабораторії Goldi [18]

1.3 Опис відомостей про системи домашньої автоматизації

Окремим об'єктом дослідження ВЛ може стати модель «розумного» будинку. Системи домашньої автоматизації (СДА) або Розумний будинок являють собою централізовану, пов'язану в мережу сукупність програмного і апаратного забезпечення, що контролюють середовище на різноманітних об'єктах. Система автоматизації, керуючи різними системами будівлі, забезпечує працездатність об'єкта, зручність, комфорт і безпеку мешканців будинку, а також значно підвищує економію [19].

Зазвичай такі системи управління встановлюються в нових будівлях або в рамках реконструкції, де вони замінюють застарілу систему управління. Хоча, при бажанні, можна встановити додатково в уже функціонуючий будинок. На рисунку 1.18 зображено приклад моделі СДА.

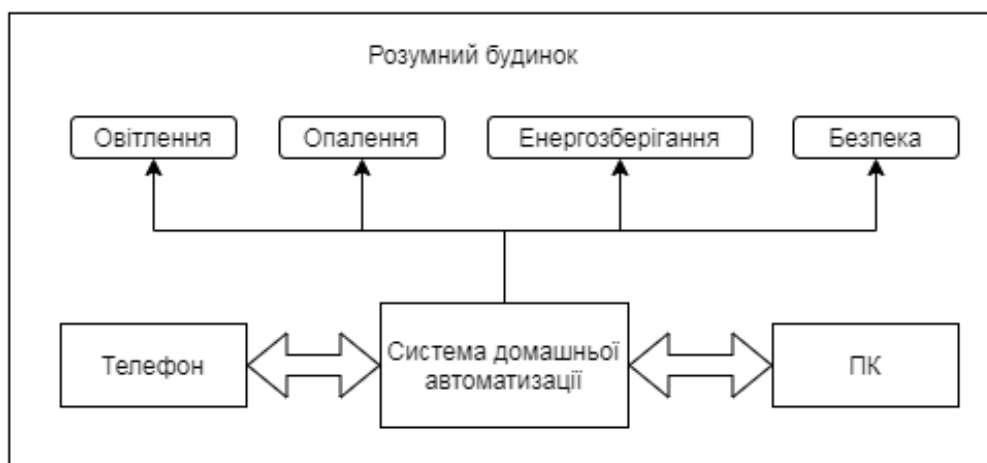


Рисунок 1.18 – Приклад моделі СДА

СДА дозволяє дистанційно керувати освітленням, кондиціонером та іншою побутовою технікою. Окрім цього, пульт керування може перебувати як на стіні у вигляді сенсорної панелі, так і бути переносним (наприклад, смартфон чи ноутбук).

Серед переваг СДА є не тільки зручність управління будинком, і а можливість такої системи значно заощадити на електроенергії. За рахунок того, що освітлення та інші електричні пристрої будуть працювати в зазначений час, а також стільки, скільки це потрібно для комфортного перебування вдома, споживання електроенергії значно зменшиться. Всі необхідні налаштування зручно коригуються, тому робота системи може бути налаштована на максимальне зниження споживання ресурсів. Також система попередження має можливість сповістити в разі виникнення проблем або вказати на можливість збоїв в роботі системи [20].

Існує декілька спроб створити ВЛ саме для вивчення систем типу Розумний будинок.

Одна з таких – це лабораторія від Intel, що надає засоби розробки технологій Інтернету речей (Internet of Things, IoT). Вона дозволяє створювати повні, масштабовані та оптимізовані рішення IoT. На рисунку 1.19 зображено приклад використання технологій від Intel для Розумного будинку [21].

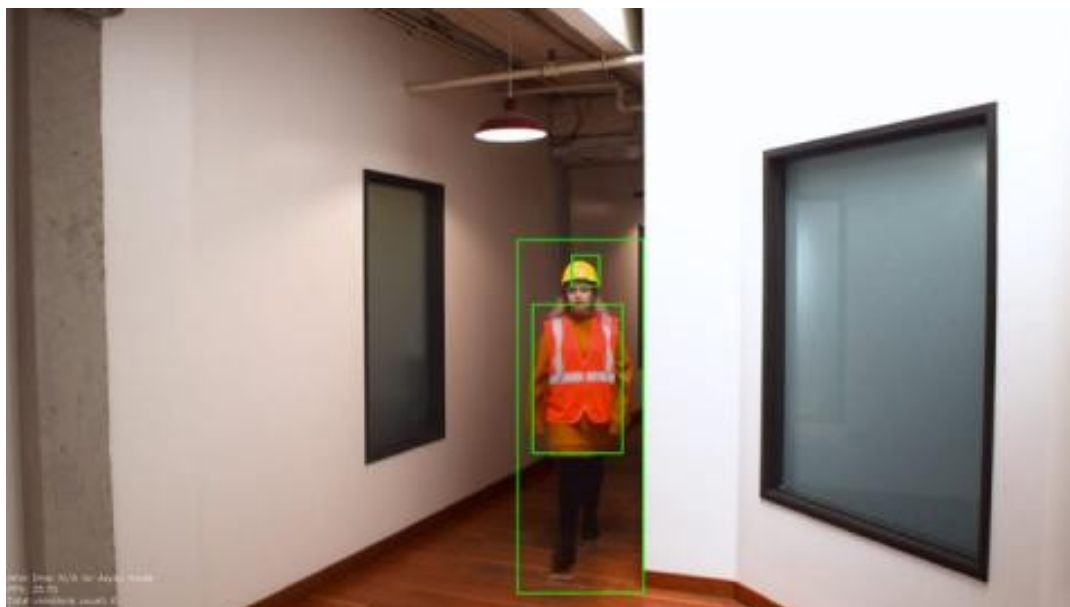


Рисунок 1.19 – Приклад використання системи розпізнавання образів від Intel [21]

Окрім неї, є ще СДА від Корнельського університету на основі бездротових технологій та з голосовим управлінням. Вона дозволяє людям керувати своїми домашніми пристроями за допомогою голосових команд, наприклад включати освітлення, вентилятори і навіть фонову музику. Узагальнена структурна схема лабораторії зображена на рисунку 1.20 [22].

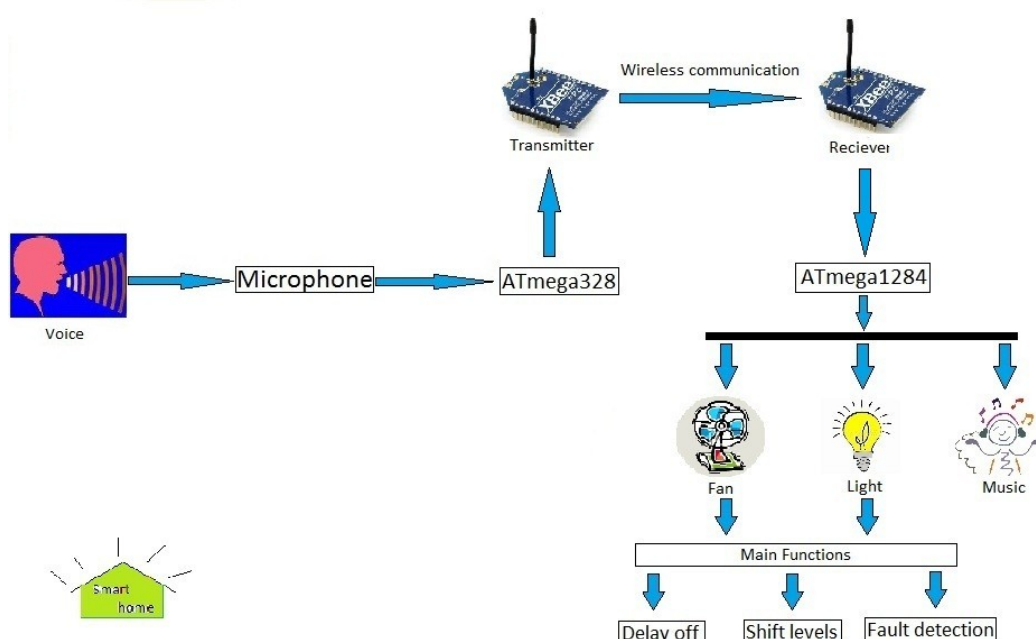


Рисунок 1.20 – Приклад СДА від Корнельського університету [22]

1.4 Опис особливостей реалізації ВЛ

Загалом, ВЛ REIoT складається з декількох складових частин, таких як RELDES, Smart House & IoT та зі стенду домашньої автоматизації MyHome від компанії Legrand, що на теперішній час інтегрується до ВЛ REIoT. Загальна структура зображена на рисунку 1.21 [23].

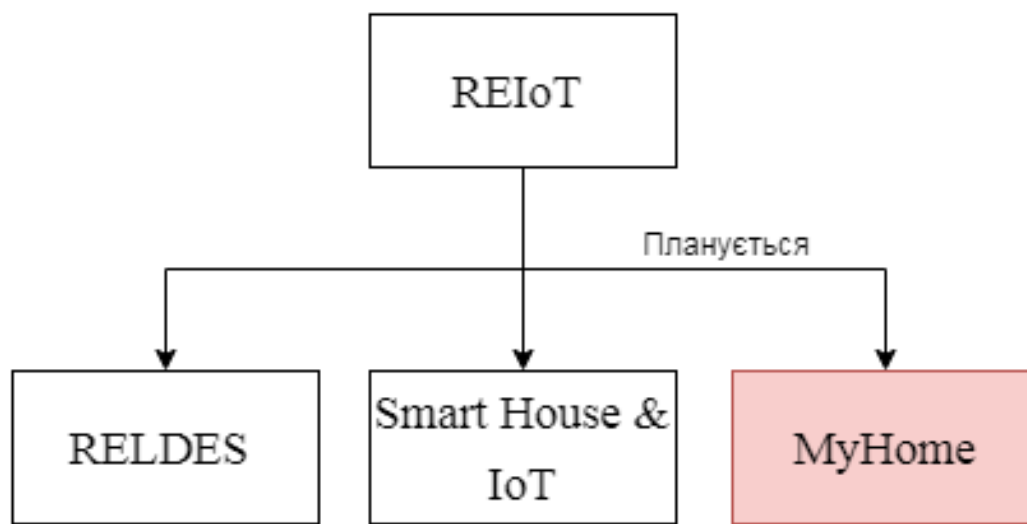


Рисунок 1.21 – Загальна структура ВЛ REIoT

Віддалена лабораторія Smart House&IoT включає набір експериментів.

Сонячна панель – експеримент передбачає ознайомлення з принципами отримання та накопичення сонячної енергії.

Контроль доступу – експеримент передбачає ознайомлення з принципами систем контролю доступу в різні приміщення з використанням кодового замку.

Вентиляція – експеримент передбачає ознайомлення з принципами систем вентиляції з регулюванням швидкості повітряного потоку та нагріванням повітря для обігріву приміщень.

Управління освітленням – експеримент передбачає ознайомлення з принципами драйвера навантажень створенням різнокольорових ефектів освітлення за допомогою RGB LED-стрічкою.

Клімат контроль – експеримент передбачає ознайомлення з принципами клімат-контролю на основі датчиків температури і вологості, а також контролю якості повітря за допомогою датчика хімічних домішок в повітрі. Також використовуючи елемента Пельтьє можна коригувати показники вологості та температури.

Контроль освітленості – експеримент передбачає ознайомлення з принципами роботи димера, використовуючи білу світлодіодну стрічку та з його допомогою змінювати яскравість.

Контроль безпеки – експеримент передбачає ознайомлення з принципами побудови і використання системи безпеки, що має бути взаємопов'язаною з усіма вузлами системи незалежно від інших.

Контроль присутності – експеримент передбачає ознайомлення з принципами систем відслідковування присутності та роботою з безпекою.

Контроль зони – експеримент передбачає ознайомлення з принципами роботи систем контролю периметра за допомогою лазерного модулю та фоторезистора.

Wi-Fi безпека – експеримент передбачає ознайомлення з принципами роботи бездротових wi-fi-датчиків та дослідити безпеку використання датчиків за допомогою одного з wi-fi-модулів. В ході експерименту виконується метод деаутентифікації клієнтів, що перешкоджає безвідмовній роботі датчиків та усієї системи в цілому [24].

В якості сервера віддаленої лабораторії виступає мінікомп'ютер Raspberry Pi 2. На ньому встановлено програмне забезпечення домашньої автоматизації OpenHAB. Для зв'язку з мікроконтролерами NodeMcu V3 та Arduino Nano V3.0 використовується відкритий комунікаційний промисловий протокол Modbus. Для передачі інформації використовується послідовна лінія RS-232 з підключеними бібліотеками Modbus TCP Binding

по протоколу за інтерфейсом USB. Кожний мікроконтролер під'єднано до модулю конвертера UART TTL для зв'язку по послідовній лінії.

На усі мікроконтролери Arduino завантажено програми, що проводять обчислення та виконують певні операції, а також передають та отримують інформацію від сервера за допомогою бібліотеки Modbus Master/Slave. Дані передаються за допомогою регістрів, що складаються з 8 або 16 елементів, з яких кожний має 16 біт. Вони зчитуються або заповнюються при опитуванні мікроконтролерів сервером платформою OpenHAB.

Спостереження за лабораторією відбувається за допомогою використання IP-камери, що передає кінцевому користувачеві потокове відеозображення. Зв'язок комп'ютера з камерою проводиться за допомогою протоколу TCP/IP по Ethernet кабелю [23].

Віддалена лабораторія RELDES надає доступ до 4 експериментів на базі платформ Arduino UNO та Arduino MEGA 2560.

Дисплей – експеримент передбачає ознайомлення з принципами роботи рідкокристалічного дисплею за допомогою завантаження власного або запропонованого коду та спостереження за результатами.

Світлодіоди – експеримент передбачає ознайомлення з принципами роботи зі світлодіодами за допомогою завантаження власного або запропонованого коду та спостереження за результатами.

Сервопривід – експеримент передбачає ознайомлення з принципами роботи з серводвигуном, який здатний обертатися на 360° , за допомогою завантаження власного або запропонованого коду та спостереження за результатами.

Датчик відстані – експеримент передбачає ознайомлення з принципами роботи з серводвигуном, який здатний обертатися на 180° , на який встановлено датчик відстані, обертаючи який можна відслідковувати відстань до певних об'єктів за допомогою завантаження власного або запропонованого коду та спостереження за результатами [25].

На рисунку 1.22 зображено приклад проведення експерименту в ВЛ Smart House & IoT.

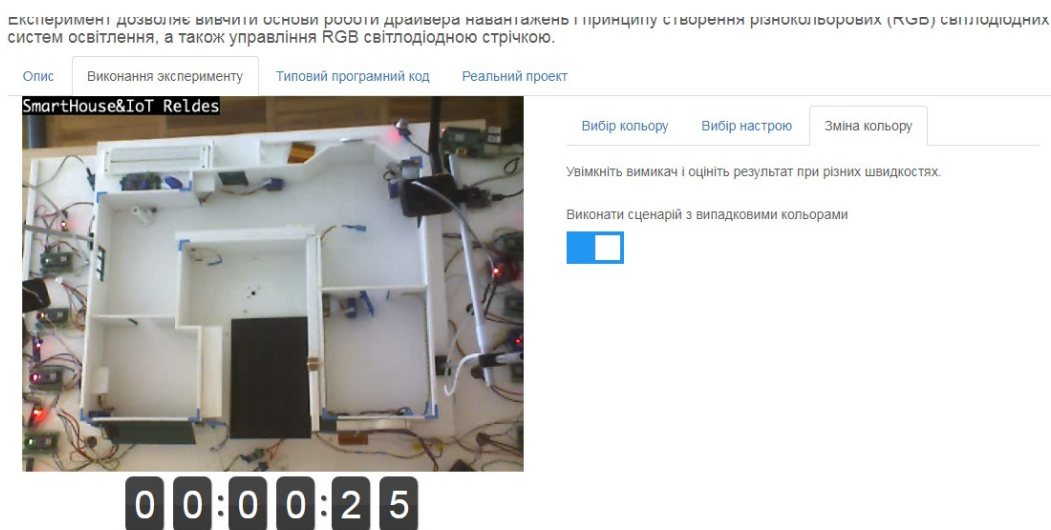


Рисунок 1.22 – Приклад проведення експерименту в ВЛ Smart House&IoT

Кожна з чотирьох плат Arduino за допомогою кабелю USB з'єднана з комп'ютером, що дозволяє безпосередньо з сайту через відповідну форму завантажувати або файл скетчу формату *.ino, або власне програмний код, який буде запрограмований у відповідний мікроконтролер. На сайті також є декілька прикладів коду для виконання експериментів. На рисунку 1.23 зображено приклад проведення експерименту в ВЛ RELDES.



Рисунок 1.23 – Приклад проведення експерименту в ВЛ RELDES

Опис складових експериментів лабораторії REIoT наведено в табл. 1.1.

Таблиця 1.1 – Опис складових експериментів ВЛ REIoT

№	Експеримент	Апаратне забезпечення
1	2	3
1	Сонячна панель	Arduino Nano (ATmega328) Сонячна панель (6В, 250мА) Резистивний дільник напруги Контроллер заряду (TP509) Акумулятор (3.7V, 1100мА)
2	Управління освітленням	Arduino Nano (ATmega328) Мостовий драйвер двигунів постійного струму (L298) Світлодіодна RGB-стрічка (SMD 5050) Блок живлення постійного струму (5V, 1A та 12V, 4A)
3	Вентиляція	Arduino Nano (ATmega328) Мостовий драйвер двигунів постійного струму (L298) Реле нормально розімкнуте (5V) Термoeлектричний перетворювач (елемент Пельтьє) Датчик вологості та температури (DHT22)
4	Контроль доступу	Arduino Nano (ATmega328) RFID-модуль (RC522) Аналоговий серводвигун на 180° (SG90)
5	Клімат - контроль	Arduino Nano (ATmega328) Датчик вологості та температури (DHT11) Датчик контролю якості повітря (MQ135)
6	Контроль освітлення	Arduino Nano (ATmega328) Мостовий драйвер двигунів постійного струму (L298) Світлодіодна біла стрічка (SMD 3528) Блок живлення постійного струму (5V, 1A та 12V, 4A)

Продовження таблиці 1.1

1	2	3
7	Контроль безпеки	Arduino Nano (ATmega328) Піроелектричний інфрачервоний PIR датчик руху (HC-SR501) Аналоговий серводвигун на 180° (SG90) Мостовий драйвер двигунів постійного струму (L298) Світлодіодна RGB-стрічка (SMD 5050)
8	Контроль присутності	Arduino Nano (ATmega328) Піроелектричний інфрачервоний PIR датчик руху (HC-SR501) Аналоговий серводвигун на 180° (SG90)
9	Контроль зони	Arduino Nano (ATmega328) Фоторезистор LDR (VT90N) Модуль лазера червоного кольору (KY-008)
10	Wi-Fi безпека	NodeMcu V3 (ESP8266) Герметичний цифровий датчик температури (ds18b20) USB WiFi модуль
11	LCD дисплей	Arduino Mega (ATmega2560) Рідкокристалічний дисплей (LCD 1602) Адаптер I2C/IIC для дисплею (PCF8574)
12	Світлодіоди	Arduino Nano (ATmega328) Світлодіоди(4 шт.)
13	Серводвигун	Arduino Nano (ATmega328) Цифровий серводвигун на 360° (MG996R-360)
14	Датчик відстані	Arduino Nano (ATmega328) Аналоговий серводвигун на 180° (SG90) Ультразвуковий датчик відстані(HC-SR04)

1.5 Аналіз проблем та недоліків віддаленої лабораторії

Результати дослідження особливостей роботи ВЛ REIoT показали наявність певних проблем та їх наслідків (табл. 1.2).

Таблиця 1.2 – Аналіз проблем та їх наслідків для ВЛ REIoT

№	Експеримент	Апаратні проблеми	Програмні проблеми	Наслідки
1	2	3	4	5
1	Сонячна панель	Не підключено датчик струму та напруги	Не налаштовано програму для коректного зчитування показань	Неможливо проаналізувати показники напруги та сили струму
2	Контроль доступу	Відсутні	Відсутня реалізація датчика RFID-модулю. Відбувається лише імітація	Візуально недоліків немає. Зчитування ключу відсутнє, тому відбувається лише його імітація
5	Клімат контроль	Відсутні	Невірно працює модуль стану повітря	Неможливо отримати правильні дані стану повітря
4	Контроль присутності	Невідповідний датчик	Відсутні	Внаслідок невідповідного датчику не завжди спрацьовує тривога
5	Wi-Fi безпека	Відсутні	Відсутня реалізація wi-fi атаки, відбувається лише її імітація	Візуально недоліків немає. Деаутентифікація клієнтів програмно відсутня, тому відбувається лише її імітація
6	Контроль безпеки	Невідповідний датчик	Відсутні	Внаслідок невідповідного датчику не завжди спрацьовує тривога
7	LCD дисплей	Недостатньо живлення. Не оптимальний кут зору до камери	На сервері відсутня необхідна бібліотека. Невірно працює код прикладів	При спробі завантаженні коду виникає помилка. При роботі яскравість замала. З камери зображення недостатньо добре видно
8	TFT дисплей	Недостатньо живлення.	На сервері відсутня необхідна бібліотека	При спробі завантаженні коду виникає помилка. При роботі яскравість замала.
9	Серводвигун	Недостатньо живлення.	Відсутні	Робота модуля приводить до просадки живлення, що порушує роботу інших модулів.
10	Датчик відстані	Недостатньо живлення.	Показники датчика не виводяться.	Неможливо ознайомитись з результатами. Робота модуля приводить до просадки живлення, що порушує роботу інших модулів.

В ході досліджень у такої досить складної системи як комплекс віддаленої лабораторії REIoT було виявлено ряд як програмних, так і апаратних проблем. Вони перешкоджають зручному використанню лабораторії, а деякі з них на даний момент повністю унеможливають виконання деяких експериментів.

Зокрема це стосується лабораторії Smart House & IoT та таких експериментів, як Сонячна панель, Контроль доступу, Клімат контроль, Контроль присутності, Контроль безпеки, Wi-Fi безпека.

Для ВЛ RELDES до проблемних належать експерименти з LCD дисплеєм, TFT дисплеєм, серводвигунами та датчиком відстані.

Однією з вагомих апаратних помилок віддаленої лабораторії RELDES є неякісне живлення системи. Внаслідок цього виникають іноді неявні аномалії. Наприклад, внаслідок цього дисплеї працюють з низькою яскравістю, що майже унеможливує спостереження за ними через камеру.

Низька потужність живлення негативно впливає на роботу серводвигунів, викликаючи виникнення артефактів при повороті валу двигуна. При роботі серводвигунів виникають сильні просадки всієї електромережі, що позначається в першу чергу на блиманнях дисплеїв, а також на роботі інших серводвигунів, які також починають працювати з затримками.

Серед помилок програмного характеру є проблема, що виникає на стороні сервера. Внаслідок певних дій з оновлення ПЗ було втрачено бібліотеку для експериментів з дисплеями. Через це при завантаженні програмного коду через відповідну форму виникає помилка, що свідчить про відсутність необхідної бібліотеки.

Програмна проблема виникає при поведенні експерименту з відслідковуванням відстані за допомогою серводвигуна та ультразвукового датчику. На сайті не виводяться значення, що отримуються через послідовний порт, а тому користувач не має змоги ознайомитися з результатами експерименту.

Також проблеми виявлено і для ВЛ Smart House & IoT. Адже вона також має певні недоліки, що викликають помилки функціонування.

В ході проведення експериментів Контроль присутності та Контроль безпеки іноді виникає проблема неспрацювання датчиків. Дана проблема є суто апаратною, адже використовується інфрачервоний датчик, що спрацьовує при знаходженні тепла наприклад людського тіла. А оскільки в макеті відбувається лише симуляція пересування людини по будинку, тобто датчик має спрацьовувати на відкривання наприклад вікна серводвигуном, то його використання є не зовсім доцільним.

Експерименти з Wi-Fi безпекою ставлять на меті проаналізувати загрозу деаутентифікації користувачів від точки доступу, проте оскільки уся модель під'єднана до маршрутизатора за допомогою кабелю, то проведення даного експерименту не є досить доцільним та наочним. В результаті виникає лише імітація атаки.

Експеримент контролю доступу візуально працює правильно, проте реалізація датчика RFID-модулю відсутня, адже відбувається лише його імітація. Через це двері завжди відчиняються після піднесення брелка серводвигуном до датчика за заданим попередньо алгоритмом незалежно від того, вірний ключ чи ні. Таким чином втрачається наочність експерименту.

Сонячна панель даного макету виконує перетворення сонячної енергії в електричну та її накопичення в акумуляторах для автономного використання. Проте в ході досліджень було виявлено, що показники таких параметрів величин як напруга та сила струму некоректно зчитуються. Це виникає через те, що модулі були невірно підключені до плати. Також після вірного підключення виявлено помилки програмної частини, адже зчитування відбувається невірно, тому вимагає обробки даних. Без неї «сирі» дані мають випадкові викиди та нестабільне зчитування.

Аналіз роботи системи також показав відсутність такого важливого експерименту як Розумна теплиця. Така система є дуже актуальною і

широко використовується в реальних будинках з домашньою автоматизацією. Саме тому актуальною задачею є додавання такого експерименту до ВЛ REIoT.

Нова версія платформи OpenHAB відрізняється від попередньої значно зручнішим використанням, більшою швидкістю, а особливо можливостями візуального інтерфейсу, що покращує сприйняття та поліпшує наочність. Тому модернізація передбачає перехід до нової версії платформи та відповідне переналаштування внутрішнього функціонування макету.

Таким чином, до вимог проєкту належить усунення проблем, описаних вище, а також розширення функціональних можливостей. Це дозволить користувачам зручно використовувати систему, отримувати результати, а головне – надасть можливість проведення віддалених експериментів з проєктування вбудованих систем на платах Arduino та вивчення технологій домашньої автоматизації.

1.6 Постановка задач дипломної роботи

У відповідності до виявлених недоліків ВЛ RELEDDES було поставлено ряд задач. Загалом, їх можна умовно розділити на модернізацію програмного забезпечення, та на модернізацію апаратного забезпечення.

Як варіанти можливих удосконалень устаткування експериментів, а також програмного забезпечення можливо розглянути наступні:

- замінити інфрачервоні датчики;
- додати додатковий модуль Wi-Fi для експерименту деаутентифікації;
- додати можливість виведення інформації з послідовного порту на екран для експерименту з датчиком відстані;
- замінити імітацію реальними діями для підвищення наочності експерименту з RFID-модулем;
- налагодити електроживлення для підвищення потужності;
- усунути проблему просадки живлення;
- налагодити правильність підключення датчиків струму та напруги;
- замінити мінікомп'ютер Raspberry Pi на нову версію та встановити останню версію OpenHAB;
- оновити бібліотеки для кращої роботи модулів;
- реалізувати коректне зчитування показників.

Окремо серед них стоїть задача програмно-апаратної модернізації – додати новий експеримент Розумна теплиця.

Для експерименту з атакою по Wi-Fi доречнішим в майбутньому було б вирішення даної проблеми за допомогою додавання ще одного модулю, що мав би під'єднання через Wi-Fi, і вже на його основі спостерігати за небезпекою такої атаки.

В майбутньому для експерименту з RFID-модулем необхідно реалізувати функцію зчитування. Це зробить макет більш наближеним до реального проєкту.

Для можливості ознайомлення з результатами експерименту з датчиком відстані в майбутньому має бути реалізоване виведення даних в відповідне поле на екранній формі проведення експерименту.

Проблема функціонування інфрачервоних датчиків може бути в майбутньому вирішена шляхом заміни на ультразвукові або лазерні модулі.

Загалом ВЛ REIoT вимагає комплексної модернізації, а тому тема даної дипломної кваліфікаційної роботи є актуальною.

В результаті роботи має бути модернізовано апаратне та програмне забезпечення ВЛ RELDES для покращення функціонування системи, вирішення наявних проблем та усунення недоліків, що дозволить користувачам зручно проводити експерименти.

Таким чином, мета роботи – модернізація програмно-апаратного забезпечення ВЛ RELDES для розширення функціональних можливостей та підвищення ефективності її використання в навчальному процесі.

Задачі роботи:

- аналіз вимог щодо модернізації програмно-апаратного забезпечення віддаленої лабораторії;
- проєктування архітектури програмно-апаратного забезпечення;
- вибір апаратних та програмних засобів розробки;
- конструювання програмно-апаратного забезпечення;
- розробка програмної документації.

2 ПРОЄКТУВАННЯ ПРОГРАМНО-АПАРАТНОГО ЗАБЕЗПЕЧЕННЯ ДЛЯ МОДЕРНІЗАЦІЇ ВЛ

2.1 Аналіз вимог до проєкту

В ході досліджень було проаналізовано апаратну та програмну складову ВЛ RELDES. Відповідно до виявлених проблем, поставлено ряд вимог, що мають бути реалізовані в ході дипломної роботи.

В результаті модернізації мають бути виконані наступні вимоги до проєкту:

- електроживлення лабораторії має бути достатньо потужним;
- просадки напруги не повинні виникати при використанні потужних модулів;
- на сервері має бути встановлена бібліотека для роботи дисплеїв, а на сайті знаходитися відповідні програмні коди прикладів;
- датчики струму та напруги мають бути правильно підключені та правильно зчитувати показники;
- додати новий експеримент Розумна теплиця;
- замінити мінікомп'ютер Raspberry Pi з поточної на нову версії, і встановити нову версію OpenHAB.

По-перше, необхідно зафіксувати дисплей в такому положенні, в якому кут зору для камери не буде обмеженим, відповідно, також підвищиться яскравість зображення.

Проблему низької потужності електроживлення системи ВЛ RELDES необхідно усунути шляхом додавання нових ліній. Для живлення серводвигунів необхідно додати електролітичні конденсатори ємністю від 2000 мкФ і напругою більше 5 В.

В результаті цих дій має бути згладжена просадка живлення, що виникає в пікові моменти роботи двигуна. Внаслідок цих дій проєкт має бути енергостійким та працювати більш стабільно.

Для вирішення проблеми відсутності бібліотеки необхідно на сервер завантажити нову більш зручну її версію, а також під неї має бути переписано приклади програмного коду, що надаються на сайті.

Однією з вагомих вимог є необхідність налагодження правильності зчитування сили струму та напруги шляхом використання фільтрів.

Така проблема виникає також і у модулів напруги та струму живлення усієї системи загалом. Таким чином модернізація вимагає роботи з цими модулями та обробки отриманих даних, в ході чого мають бути усунені дані проблеми.

Експеримент Розумна теплиця має вірно та коректно зчитувати показники з датчиків та дозволяти отримувати необхідні параметри, такі як дані про температуру, вологість та освітлення, а також дистанційно керувати освітленням, поливом та відкриттям та закриттям вентиляційних отворів. В результаті має бути реалізовано автономний набір пристроїв, що функціонують як у реальній теплиці. В ході роботи має бути написано код та зібрано макет, на якому буде перевірено функціональність даного експерименту.

Однією з важливих вимог до проєкту є заміна мікрокомп'ютеру на Raspberry Pi на версію 3.0, а також переустановлення ОС та нової версії OpenHAB.

Таким чином, в результаті модернізації має бути забезпечено описані вимоги, що дозволить користувачам зручно та без перебоїв ставити експерименти та проводити дослідження.

2.2 Архітектура системи

Архітектуру складових ВЛ REIoT, а саме RELDES та Smart House & IoT та місце майбутніх розробок показано на рисунках 2.1 та 2.2.

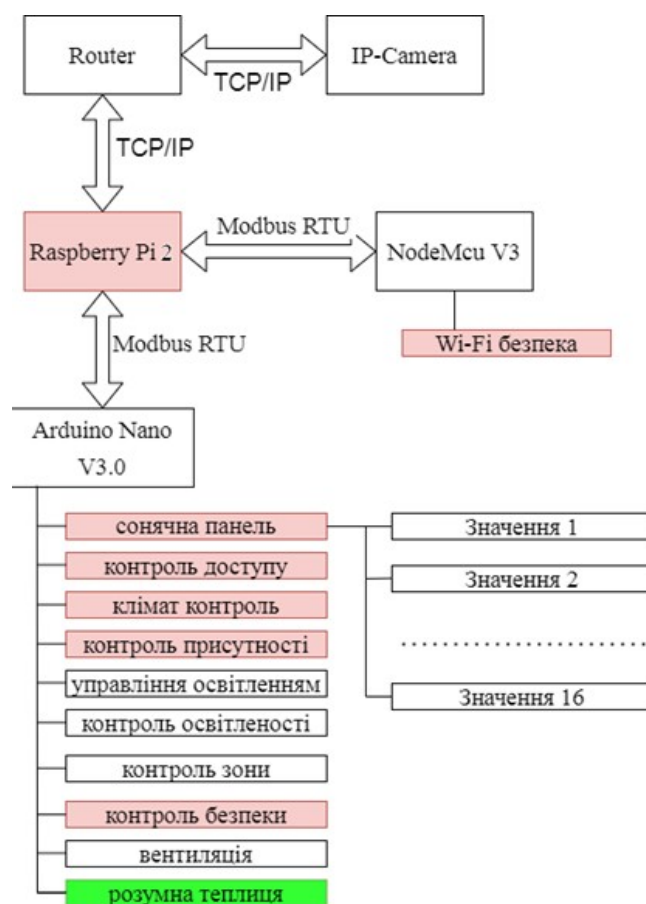


Рисунок 2.1 – Архітектура віддаленої лабораторії RELDES

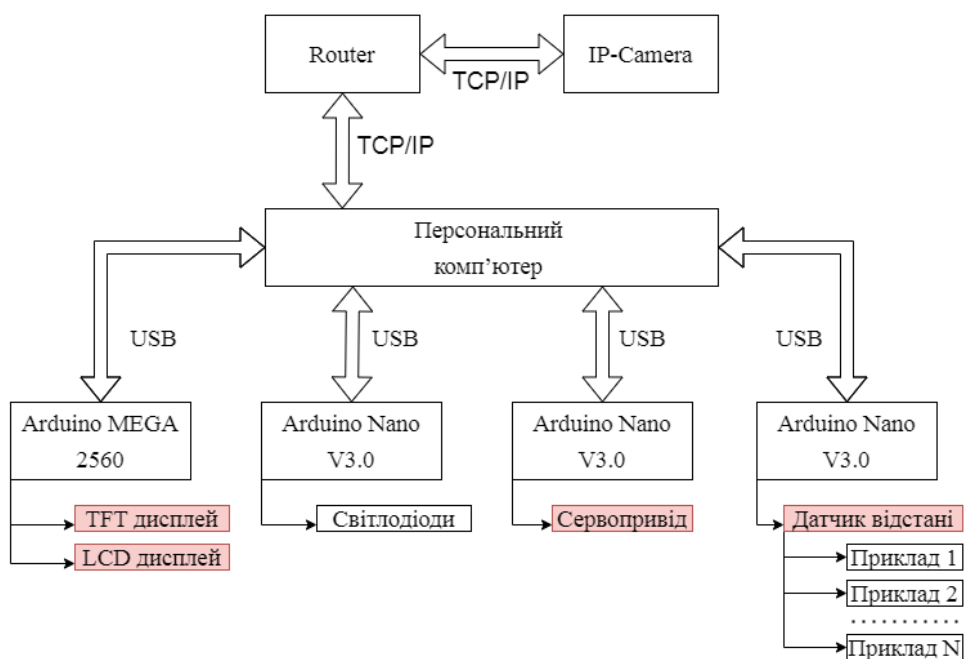


Рисунок 2.2 – Архітектура віддаленої лабораторії Smart House&IoT

2.3 Вибір апаратних компонентів

В ході модернізації віддаленої лабораторії Smart House & IoT було замінено одноплатний мікрокомп'ютер Raspberry Pi з версії Pi 2 B на Pi 3 B. Їх порівняння наведено в таблиці 2.1 [26].

Таблиця 2.1 – Порівняльна характеристика мінікомп'ютерів Raspberry Pi

Характеристика	Raspberry Pi 2 B	Raspberry Pi 3 B
CPU	Broadcom BCM2836	Broadcom BCM2837
GPU	Video Core IV 250 МГц	Video Core IV 400 МГц
Кількість ядер	4x900 МГц	4x1.2 ГГц
Wi-Fi	-	802.11n 4.1
Відео вихід	HDMI, TRRS	HDMI, TRRS, DSI

Версія Raspberry Pi 3 B новіша на рік за Pi 2 B (2016 та 2015 відповідно). Також відрізняється більш кращими характеристиками обладнання (див. табл. 2.1). Загалом, обидва мікрокомп'ютери мають по 1 Гб оперативної пам'яті, по 4 USB-порти та швидкість передачі даних по LAN 100 Мб/сек. Мають аудіо виходи I²S, HDMI, 3.5 jack. Для підключення датчиків та модулів надається 40 пінів. Обидва мають CSI вхід відео.

Для експерименту Розумна теплиця було обрано компоненти, представлені в таблиці 2.2.

Таким чином, в якості мікроконтролеру було обрано Arduino Nano v3.

Arduino Nano - одна з найпопулярніших та наймініатюрніших плат Arduino. Є повним аналогом Arduino Uno. Працює на чіпі ATmega328 або ATmega168. Містить наступний перелік характеристик:

Таблиця 2.2 – Апаратні компоненти експерименту Розумна теплиця

Компонент	Функція
Модуль реле 5В 10А з опторозв'язкою на 2 канали	Увімкнення та вимкнення освітлення та подачі води для поливу
Сервопривід SG90	Керування вентиляцією
Температурний датчик водонепроникний DS18B20	Зчитування показань температури ґрунту
Датчик вологості та температури GY-21 HTU21	Зчитування показань температури та вологості в теплиці
Датчик освітленості TEMT6000	Зчитування показань освітленості
Ємнісний датчик вологості ґрунту	Зчитування показань вологості ґрунту
Конвертер USB to RS-485	Конвертація сигналу інтерфейсу USB в інтерфейс RS-485
Конвертер RS-485 to TTL	Конвертація сигналу інтерфейсу RS-485 в інтерфейс TTL
Arduino Nano v3	Керування пристроями

- напруга живлення 5В;
- вхідне живлення 7-12В;
- цифрових пінів – 14, 6 з них – виходи ШІМ;
- аналогових входів – 8;
- максимальний струм цифрового піну – 40 мА;
- флеш пам'ять – 16 Кб/32 Кб;
- RAM 1 Кб або 2 Кб, в залежності від чіпа;
- EEPROM- пам'ять – 512 байт/1 Кб;
- частота – 16 МГц;
- розміри 19 x 42 мм;
- вага 7 г [27].

Мікроконтролер Arduino Nano v3 зображено на рисунку 2.3 [27].



Рисунок 2.3 – Мікроконтролер Arduino Nano v3 [27]

RS-485 - це асинхронний протокол послідовного зв'язку, що не потребує тактового сигналу. Максимальна відстань передачі даних є 1200 м, яку можна збільшити за допомогою підсилювачів сигналу. Проте на великих відстанях швидкість передачі – 60 кб/с. Використовується кручена пара, до однієї лінії можна підключити до 32 пристроїв. Модуль оснований на мікросхемі MAX485, що перетворює сигнал TTL в стандарт RS485 і назад. Зображений на рисунку 2.4 [28].



Рисунок 2.4 – Модуль конвертера TTL-RS485 [28]

Модуль конвертера TTL-RS485 має наступні характеристики:

- напруга 5 В;
- струм режимі очікування < 5 мА;
- частота передачі даних: < 2,5 Мбіт/с [28].

Модуль конвертеру USB to RS-485 використовується для перетворення сигналів з формату USB в сигнали інтерфейсу RS-485; заснований на мікросхемі CH340 та зображений на рисунку 2.5 [29].



Рисунок 2.5 – Модуль конвертеру USB to RS-485 [29]

Має наступні параметри та характеристики:

- живлення від USB;
- швидкість передачі даних до 921600 бод;
- інтерфейси USB та RS-485 [29].

Ємнісний датчик вологості ґрунту використовується для створення систем автоматичного поливу рослин. Датчик має наступні характеристики та зображений на рисунку 2.6 [31]:

- тип датчика вологості ґрунту: ємнісний;
- живлення: 3.3-5 В;
- струм: до 6 мА;
- інтерфейс: аналоговий сигнал;
- діапазон вихідного сигналу: 0.5-3.3В;
- глибина занурення в ґрунт: до 80 мм [30].

Датчик зовнішньої освітленості ТЕМТ6000 є кремнієвим фототранзистором NPN-типу в прозорому корпусі 1206 для поверхневого монтажу. Є чутливим до видимого світла як і людське око та має максимальну чутливість на довжині хвилі 570 нм, зображений на рисунку 2.7 [32].



Рисунок 2.6 – Ємнісний датчик вологості ґрунту [31]



Рисунок 2.7 – Датчик зовнішньої освітленості TEMT6000 [32]

Датчик вологості та температури GY-21 HTU21 на основі датчика SHT21 – це високоточний модуль для вимірювання температури і вологості. Має досить низький коефіцієнт похибки в своєму класі, адже при вимірюванні температури становить приблизно 0.4%, а вологості – 2%. Через низькі похибки та інтерфейс I2C модуль підходить для вимірювання температури і вологості в промислових приміщеннях. Має наступні параметри та характеристики:

- тип датчика вологості ґрунту: ємнісний;

- інтерфейс: I2C;
 - діапазон робочої напруги: 1.9 – 3.6 В;
 - напруга живлення: 5 В;
 - струм вимірювання: 300 μ А;
 - струм очікування: 0.15 μ А;
 - вологість: робочий діапазон: від 0 до 100%;
 - температура: робочий діапазон: від -40 до +125 ° С;
 - вбудований детектор розряду батареї, що вмикається при зниженні напруги до 2.25 В;
 - вбудований нагрівач для самодіагностики датчиків [33].
- Модуль зображено на рисунку 2.8 [33].



Рисунок 2.8 – Датчик вологості та температури GY-21 HTU21 [33]

Температурний водонепроникний датчик DS18B20 дозволяє визначити температуру в діапазоні від -55 °С до + 125 °С. Дані отримуються в форматі цифрового сигналу в 12-бітовому вигляді по протоколу 1-Wire. Це дозволяє підключити велику кількість таких датчиків, використовуючи лише 1 цифровий порт контролеру, і лише 2 дроти для всіх датчиків: землі, тобто мінусу, і сигналу – паразитне живлення, при якому енергія отримується з лінії сигналу.

Кожний датчик містить унікальний 64-бітний код, що може використовуватися контролером для передачі інформації з конкретним сенсором по загальній шині. Оскільки по двох проводах підключення датчик іноді може видавати невірні показники, можна використовувати ще й 3 дріт – живлення. Незалежно від способу підключення, сигнальний дріт необхідно з'єднати з лінією живлення резистором номіналом 4,7 кОм. Температурний водонепроникний датчик DS18B20 зображений на рисунку 2.9 [34].



Рисунок 2.9 – Температурний водонепроникний датчик DS18B20 [34]

Датчик має наступні параметри та характеристики:

- діапазон температур: від -55 до $+125$ °C;
- точність: $\pm 0,5$ °C в межах від -10 до $+85$ °C ;
- час отримання даних: 750 мс при 12-бітній передачі та 94 мс при 9-бітній;
- напруга живлення: 3 – 5.5 В;
- струм при бездіяльності: 750 нА;
- струм при роботі: 1 мА [34].

Сервопривід SG90 використовується для повороту деталей механізмів. В корпусі міститься модуль керування, що під дією вхідного сигналу подає живлення відповідної полярності на електродвигун. Сигнал

керування містить дані про необхідне положенні валу. Для визначення поточного положення валу редуктор з'єднаний зі змінним резистором. Інформація про необхідне положення валу міститься в скважності імпульсів керуючого сигналу, частота якого має бути постійною – 50 Гц. Сервопривід SG90 зображений на рисунку 2.10 [35].



Рисунок 2.10 – Сервопривід SG90 [35]

Основні характеристики сервоприводу SG90:

- тип сервоприводу: аналоговий;
- напруга живлення: 4.8 В;
- кут повороту: 180°;
- швидкість повороту: 0.1 с / 60°;
- зусилля на валу: 1.8 / см [35].

Модуль реле 5В 10А з опторозв'язкою на 2 канали перемикає стан за допомогою напруги 5В. Він спеціально розроблений для керування слабкострумowymi платами, таких як arduino чи raspberry, що на виході можуть видати струм не більше 40 мА. Містить оптопару EL817, що реалізує гальванічну розв'язку. Модуль складається з двох незалежних частин. При підключенні логічного виводу до напруги, перемикач перебуває у високому стані, тобто логічна одиниця. Для перемикання реле необхідно в

негативний стан, тобто логічну одиницю, необхідно закоротити на землю. Модуль реле зображений на рисунку 2.11 [36].

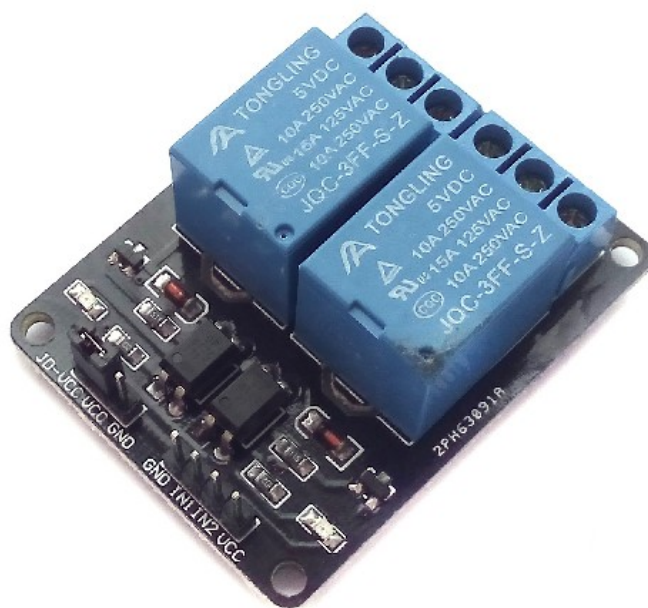


Рисунок 2.11 – Модуль реле [36]

Має наступні параметри та характеристики:

- напруга живлення: 5 В;
- струм: 30 мА;
- сигнал включення: 0 В (низький рівень) ;
- оптична гальванічна розв'язка: є;
- кількість реле: 2;
- тип реле: електромеханічне;
- номінальний струм навантаження: 10 А;
- комутована напруга: 250VAC, 30VDC [36].

2.4 Вибір середовища розробки ПЗ

Для програмування мікроконтролерів Arduino, загалом, є декілька способів.

Досить простим та зручним використанням вважається середовище розробки Arduino IDE. Існує 2 варіанти:

- при наявності стабільного підключення до мережі Internet, можна використовувати онлайн-середовище Arduino Web Editor. З його допомогою можна зберігати проекти в хмарному сховищі, що надає доступ до них з будь-якого пристрою, а також це дозволяє зберігати резервні копії. Таким чином буде усунена необхідність постійного оновлення версії, а також власне встановлення середовища безпосередньо на комп'ютер;
- для роботи в автономному режимі надається версія IDE, що встановлюється на комп'ютер. З її використанням відпадає необхідність постійного підключення до мережі Internet для роботи з мікроконтролерами [37].

Середовище розробки Arduino IDE містить вбудований редактор тексту для написання коду, вікно повідомлень, спеціально область для виведення тексту – консоль, а також панель інструментів і меню. Для завантаження програм, а також для зв'язку, необхідне підключення до апаратної частини Arduino [38]. Приклад інтерфейсу програмування в Arduino IDE наведено на рисунку 2.12 [37].

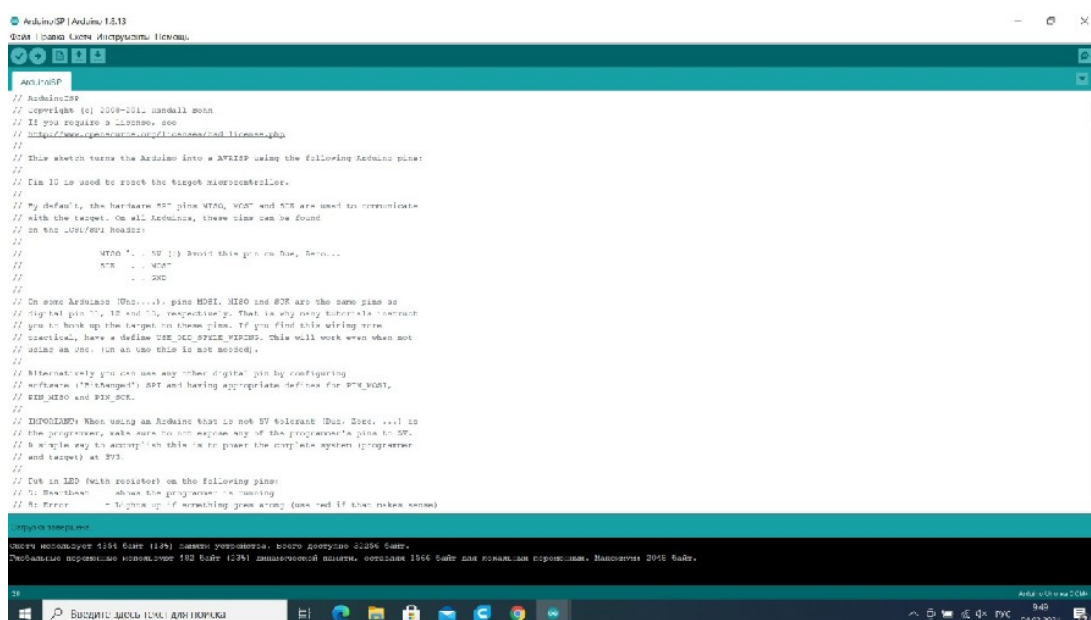
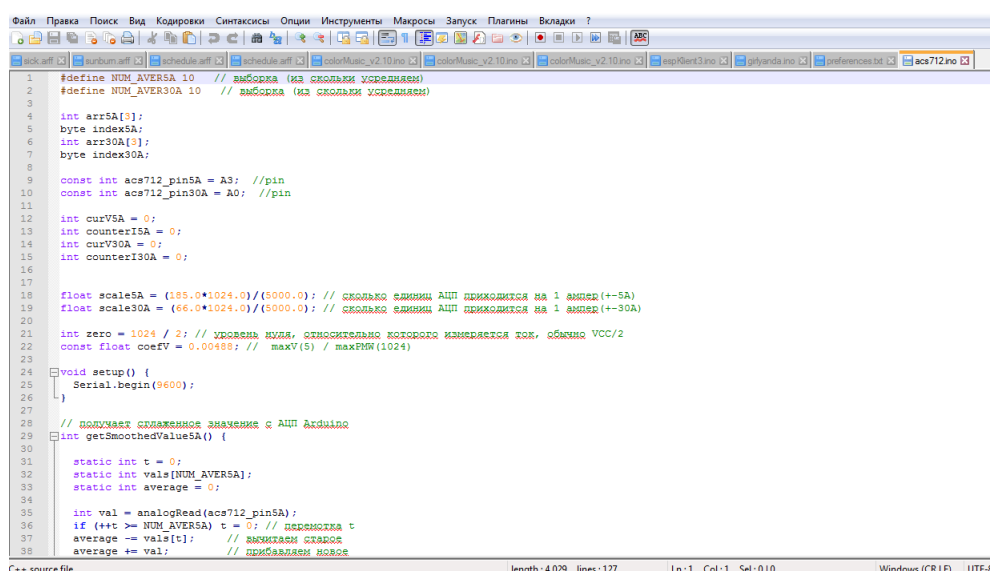


Рисунок 2.12 – Приклад інтерфейсу програмування в Arduino IDE [37]

Окрім стандартного вбудованого редактору коду можна використовувати також і сторонні, наприклад Notepad++. Notepad++ – текстовий редактор з відкритим кодом. Досить зручний, адже надає підсвічування синтаксису багатьох мов програмування, а також розмітки. Базується на компоненті Scintilla, є написаним мовою C++. Функціональність програми може бути розширена з використанням плагінів і модулів, таких як компілятори і препроцесори. Приклад інтерфейсу зображено на рисунку 2.13 [39].



```

1 #define NUM_AVERSA 10 // выборка (из скольки усредняем)
2 #define NUM_AVER30A 10 // выборка (из скольки усредняем)
3
4 int arr5A[3];
5 byte index5A;
6 int arr30A[3];
7 byte index30A;
8
9 const int acs712_pin5A = A3; //pin
10 const int acs712_pin30A = A0; //pin
11
12 int curV5A = 0;
13 int counterI5A = 0;
14 int curV30A = 0;
15 int counterI30A = 0;
16
17 float scale5A = (185.0*1024.0)/(5000.0); // сколько единиц АЦП приходится на 1 ампер(+5A)
18 float scale30A = (66.0*1024.0)/(5000.0); // сколько единиц АЦП приходится на 1 ампер(+30A)
19
20
21 int zero = 1024 / 2; // уровень нуля, относительно которого измеряется ток, обычно VCC/2
22 const float coefV = 0.00488; // maxV(5) / maxPMW(1024)
23
24 void setup() {
25     Serial.begin(9600);
26 }
27
28 // получает сложное значение с АЦП Arduino
29 int getSmoothedValue5A() {
30
31     static int t = 0;
32     static int vals[NUM_AVERSA];
33     static int average = 0;
34
35     int val = analogRead(acs712_pin5A);
36     if (++t >= NUM_AVERSA) t = 0; // переноска t
37     average += vals[t]; // вычитаем старое
38     average /= val; // прибавляем новое
39 }

```

Рисунок 2.13 – Приклад інтерфейсу написання коду в Notepad++ [39]

Найзручніше використовувати Notepad++ саме в якості стороннього текстового редактора для написання коду, який потім буде використовуватися в Arduino IDE.

Стороннім редактором також може виступати Visual Studio Code. Це легкий, проте досить потужний редактор програмного коду, до того ж є кросплатформенним. Містить вбудовану підтримку JavaScript, TypeScript і Node.js, а також має безліч розширень інших мов програмування, таких як C++, C #, Java, Python, PHP і Go [40]. Приклад інтерфейсу зображено на рисунку 2.14 [40].

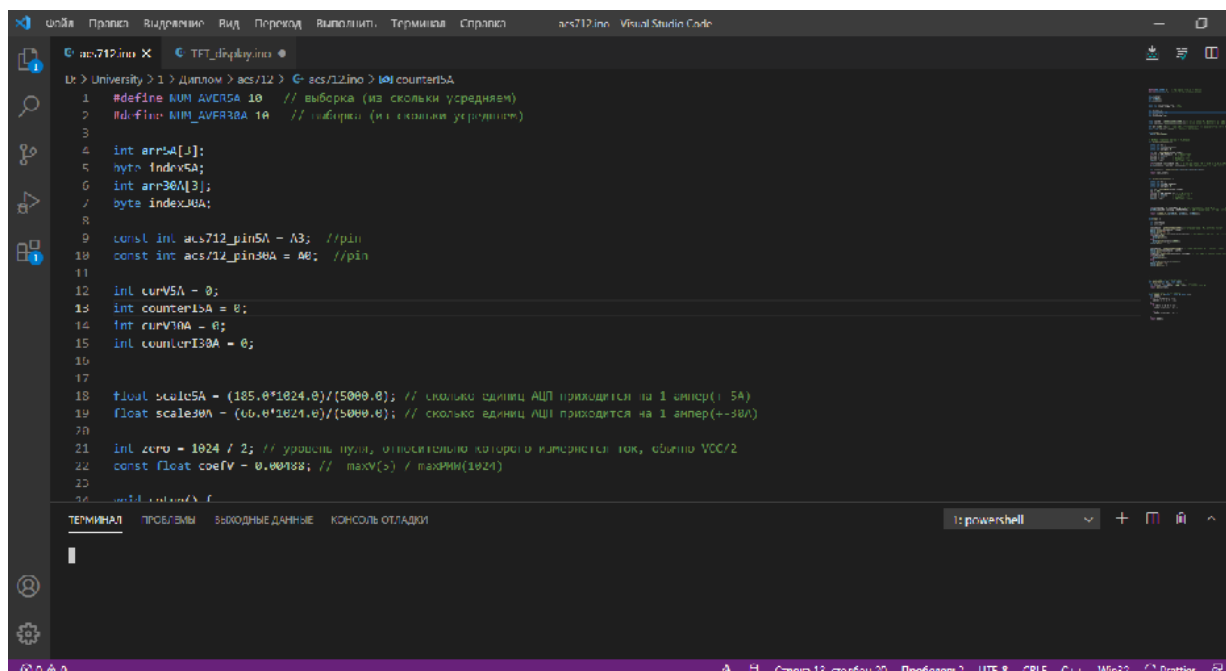


Рисунок 2.14 – Приклад написання коду в Visual Studio Code [40]

До переваг його використання в якості стороннього текстового редактора для Arduino IDE відноситься можливість автоматичного форматування тексту, включення коду в блоки, підсвічування синтаксису, а найголовніше – автодоповнення слів. Хоча, останнім часом нові версії Arduino IDE включають дані можливості в свій функціонал [40].

Проте окрім використання Visual Studio Code як редактора для Arduino IDE, за допомогою сторонніх розширень надається можливість його використання як повноцінного застосунку для програмування мікроконтролерів.

Найоптимальнішим з таких розширень є PlatformIO. Воно написане мовою Python і не має залежностей від інших додаткових бібліотек та інструментів конкретної операційної системи. Саме тому використання PlatformIO можливе як на персональному комп'ютері, так і комп'ютерами розміром з кредитну карту, як, наприклад, Raspberry Pi. Приклад його використання зображено на рисунку 2.15 [41].

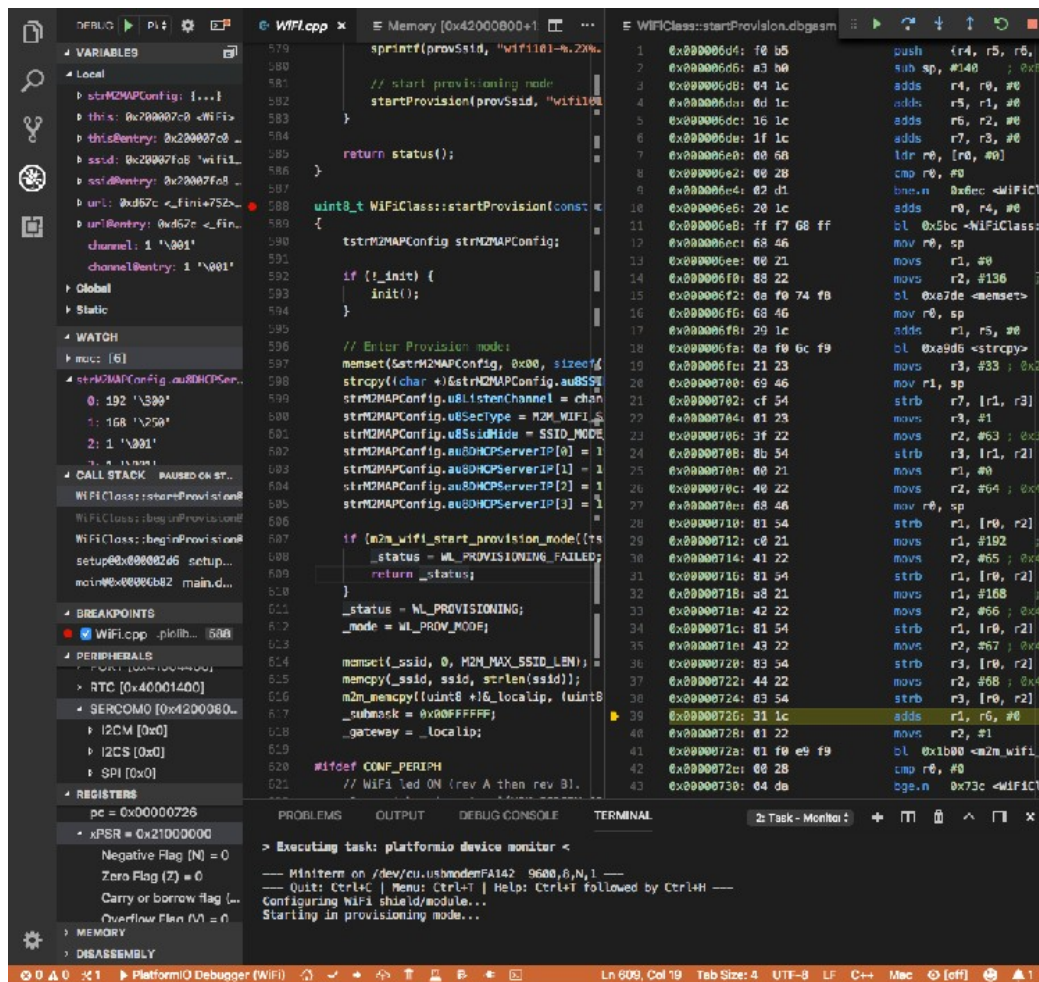


Рисунок 2.15 – Приклад інтерфейсу програмування коду в PlatformIO [41]

Дане розширення надає такі переваги:

- автоматичне завершення коду на основі типів змінних, з використанням функцій і залежностей бібліотек;
- робота з декількома проєктами супроводжується досить простою навігацією по коду та підтримкою декількох панелей;
- швидке завантаження проєктів за допомогою попередньо визначених конфігурацій для максимальної швидкодії;
- досить просто відбувається перехід в режим дизасемблювання в змішаному режимі для визначення помилок або поганої оптимізації код;
- достатньо зрозумілий майстер проєктів та багато прикладів проєктів;

- вбудований термінал і достатньо потужний монітор послідовного порту [41].

Оскільки мікроконтролери Arduino є частиною сімейства AVR, то для їх програмування можна використовувати й таке програмне забезпечення, як Atmel Studio. Це інтегроване середовище розробки від компанії Atmel для розробки проєктів під мікроконтролери ARM Cortex-M і AVR. Приклад використання Atmel Studio зображено на рисунку 2.16 [42].

5/

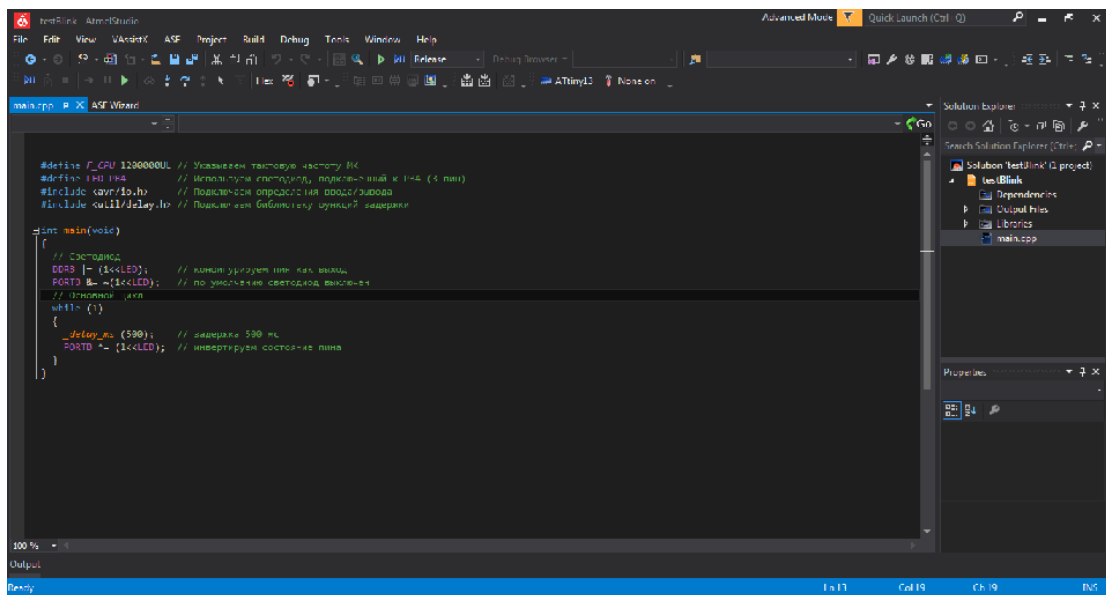


Рисунок 2.16 – Приклад використання Atmel Studio [42]

Дана програма дозволяє писати код мовою асемблера та на C/C++. Має вбудований майстер проєктів, віртуальний симулятор, редактор програмного коду і інтерфейс командного рядку. Також є підтримка компілятора GCC і плагіну AVR RTOS для операційних систем реального часу.

Окрім цього надається вибір найбільш оптимальних способів кодування для проєктів. Для прискорення написання програм дозволяється використовувати візуальні інструменти [42].

Загалом, кожне з середовищ програмування має певні переваги та недоліки, що наведені в таблиці 2.3.

Таблиця 2.3 – Порівняльний аналіз середовищ розробки

Середовище	Переваги	Недоліки
Arduino IDE	Підсвічування коду Швидка заливка скетчу в плату Досить просто позначити певний пін порту як вхід або вихід (функція pinMode) Швидко привласнюється номер порту певній змінній та позначається стан порту Є безліч вбудованих функцій	Робота тільки з платами arduino Незручно отримувати hex-файли Необхідний bootloader, що розпізнає плату підключену через USB як COM-порт Деякі вбудовані функції не завжди зручні, оскільки використовують певну апаратну частину мікроконтролера Підтримує лише певні мікроконтролери atmel Відсутність віртуального симулятора Відсутність кастомізації інтерфейсу Відсутність автодоповнення коду Незручна організація проєктів Відсутність підтримки плагінів Незручний менеджер бібліотек Незручний build-output, яким іноді складно виявити помилку
Atmel Studio	Підсвічування синтаксису Наявність CodeGuard Легко підключати зовнішні додаткові бібліотеки Швидка заливка коду в мікроконтролер Генерація вихідних hex, sof, eер файлів, для можливості заливки через зовнішній програматор Підтримує всі контролери компанії Atmel та зовнішні програматори Багато доповнень для розробок	Немає великої різноманітності вбудованих функцій Напрямок портів налаштовується безпосередньо регістрами мікроконтролерів Поганий симулятор виконання коду
PlatformIO	Зручна робота з кодом з автодоповнення слів і підстановка методів класу Жива обробка помилок при написанні Зручний навігатор по коду Файловий менеджер проєкту з навігацією по файлах і бібліотекам проєкту Швидка компіляція коду Багато плагінів Підтримка роботи безпосередньо з git Повноцінний дебаг для мікроконтролерів Юніт-тестування	Всі налаштування плати здійснюються текстом, що призводить до витрати часу на вивчення документації Відсутність штатного монітора порту Відсутність плоттера графіків, навіть у вигляді плагінів Окремі персональні налаштування конфігурації для завантаження через програматор Дрібні кнопки для завантаження і відсутність гарячих клавіш Деякі налаштування і дії виконуються через консоль Високі вимоги до коду, на відміну від Arduino IDE, дозволяє писати спрощено не дотримуючись структуру проєкту мови C Погана підтримка Arduino-бібліотек

На мінікомп'ютер Raspberry Pi, що виступає сервером лабораторії, який пов'язує усі мікроконтролери, було встановлено програмне забезпечення OpenHAB. Окрім нього можна було завантажити ще декілька найрозвиненіших Opensource систем управління домашньою автоматизацією: HomeAssistant, Majordomo, ioBroker.

Для встановлення OpenHAB на Raspberry Pi, на останній має бути встановлено певну операційну систему. На даний момент їх існує велика кількість, найпопулярніші з них наведені у таблиці 2.4.

OpenHAB – це програмна платформа, що написана мовою Java, для об'єднання різноманітних систем та технологій автоматизації в одне рішення, що надає можливість використання загальних правил автоматизації, а також пропонує уніфіковані інтерфейси. Приклад інтерфейсу OpenHAB зображено на рисунку 2.17 [43].

Таблиця 2.4 – Порівняльна таблиця ОС на Raspberry Pi

ОС	Опис	Особливості
Raspbian	Стандартна ОС на базі Debian, що була спеціально створена для Raspberry Pi	Містить стандартний набір програм і утиліт. Також містить розділ «програмування», що надає можливість використання різноманітних застосунків для програмування
OSMC	Медіацентр на базі Kodi	Має гарний функціонал ПЗ для роботи з мультимедіа
DietPi	ОС, що заснована на Debian	Займає лише 400 Мб на карті пам'яті. Має низькі вимоги до ресурсів та підтримує швидку установку програм
OpenEL EC	ОС для програвання мультимедіа	Інтерфейс схожий на Kodi, проте має вищу швидкість роботи
Windows 10 IoT Core	Версія Windows, що створена для комп'ютера Raspberry Pi	Має сумісність лише з ПК на базі Windows 10
ОС RISC	ОС, що не базується на системі Linux, внаслідок чого не схожа на інші системи	Працює як однокористувацька система. Загалом, система не використовується як основна ОС
RetroPie	Емулятор	Надає можливість грати на багатьох консолях. ОС заснована на Raspbian
Pidora	ОС, що заснована на проєкті Fedora	У порівнянні з Debian для формату пакетів має розширення RPM

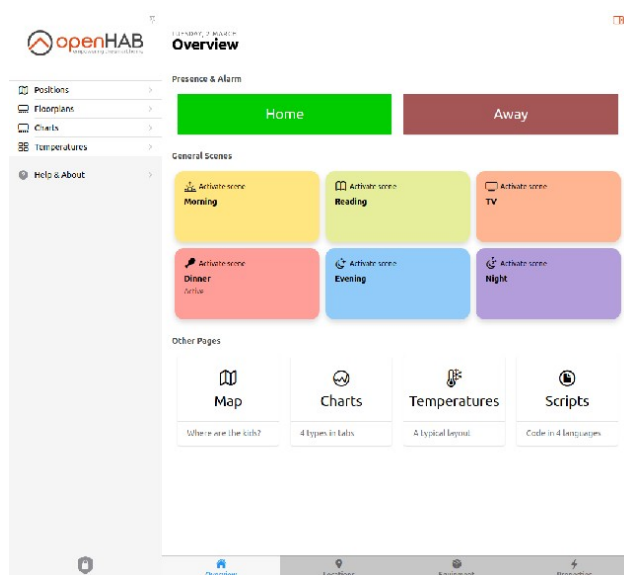


Рисунок 2.17 – Приклад інтерфейсу OpenHAB [43]

У порівнянні з аналогами OpenHAB має певні переваги:

- є незалежним від виробників;
- працює на пристроях, що підтримують Java Virtual Machine;
- надає можливість об'єднання різноманітних технологій домашньої автоматизації в одну систему;
- містить потужний механізм правил;
- поставляється з багатьма вебінтерфейсами, а також надає інтерфейси для iOS і Android;
- має відкритий код;
- надає широкий функціонал для розширення та інтеграції з новими системами і пристроями різноманітних виробників;
- надає API для інтеграції в інші системи. [44].

Таким чином, для функціонування роботи лабораторії Smart House & IoT було встановлено на Raspberry Pi операційну систему Raspbian, на яку завантажено програмне забезпечення для домашньої автоматизації OpenHAB. Для програмування мікроконтролерів використовується стандартне програмне забезпечення Arduino IDE зі стандартним текстовим редактором.

Для зв'язку окремих мікроконтролерів Arduino з Raspberry Pi використовується протокол Modbus. Modbus – відкритий комунікаційний промисловий протокол, що оснований на архітектурі master-slave. На фізичному рівні використовуються 2 типи зв'язку: послідовні інтерфейси RS-232/422/485 та канал мережі TCP/IP. Приклад реалізації на фізичному рівні зображено на рисунку 2.18 [45].

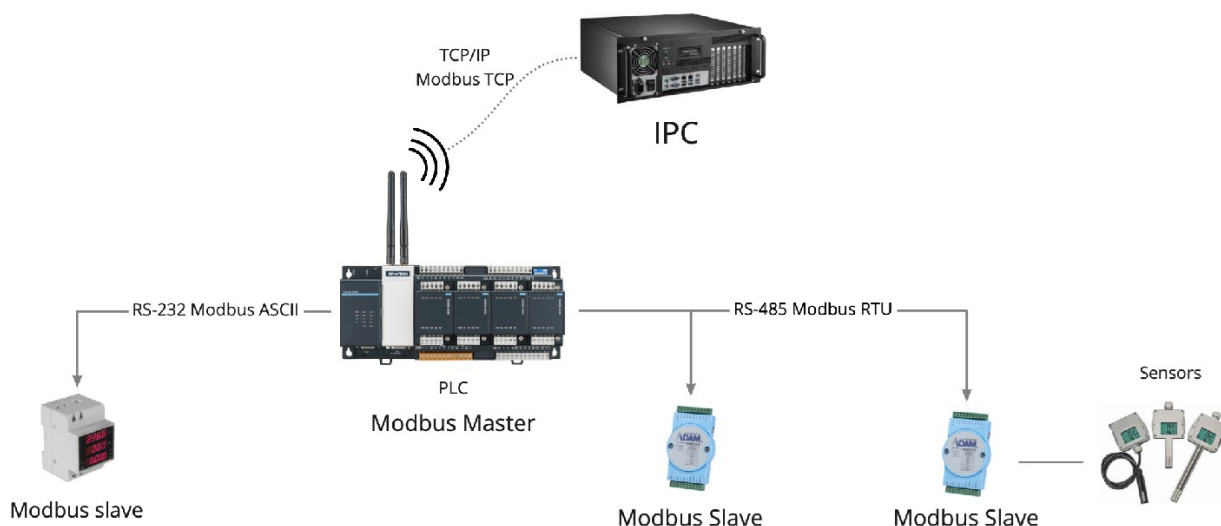


Рисунок 2.18 – Фізичний рівень протоколу Modbus [45]

На логічному рівні використовується 3 типи протоколів, в залежності від реалізації на фізичному рівні:

- Modbus ASCII. Дані кодуються символами з ASCII-таблиці та передаються в hex-форматі. Кожен пакет починається символом двокрапки, а кінець – повернення каретки і переносу рядка. В результаті це дозволяє використовувати протокол на обладнанні з менш точними таймерами, а також на лініях з великими затримками;
- Modbus RTU. Дані кодуються в бінарний формат, а розділяються пакети часовим інтервалом. Протокол є критичним до затримок та не працює, наприклад, на модемних лініях. Але витрати на передачу даних менше, ніж в Modbus ASCII, оскільки довжина повідомлень менше;

- Modbus TCP. Дані також кодуються в бінарний формат, і упаковуються в звичний TCP-пакет, що потім передається по IP-мережі. Перевірка цілісності як в Modbus RTU не застосовується, оскільки TCP вже має власний механізм контролю цілісності [45].

В лабораторії Smart House & IoT використовується інтерфейс RS-485 та протокол Modbus RTU. Всі пристрої, що використовують Modbus, сліднують моделі master-slave. Запити створює лише пристрій master, а slave тільки можуть відповісти на запити та не можуть самі починати передачу даних. В лабораторії в якості slave-пристрою виступають мікроконтролери Arduino, а опитує їх master-пристрій Raspberry Pi.

В Modbus-мережі можуть знаходитися лише до 247 пристроїв. На відміну від slave-пристроїв, що мають різні адреси, master-пристрій не має адреси. Адреса «0» використовується для широкомовних запитів від master, при цьому, slave-пристрою не можуть відповідати на ці пакети [45].

Протокол Modbus містить наступні недоліки:

- протокол не передбачає автентифікацію і шифрування даних, тому при використанні Modbus TCP необхідно використовувати VPN;
- Slave-пристрій не може почати передачу даних, а тому master має увесь час опитувати пристрої;
- Slave-пристрій не може виявити втрату зв'язку з Master [45].

2.5 Висновки

Аналіз вимог до розроблюваного з метою модернізації ВЛ REIoT програмного та апаратного забезпечення дозволив визначити місце розробки в архітектурі ВЛ REIoT та обрати апаратні компоненти, середовища розробки, ОС та протокол взаємодії складових. Плановані зміни описано в таблиці 2.5.

Таблиця 2.5 – Порівняльна таблиця «було-планується».

Було	Планується
Низька потужність системи Виникнення артефактів 5V 2A	Стабільна робота Висока яскравість дисплеїв 5V 3A
Виникнення просадок напруги Неможливість роботи потужних модулів Виклики дефектів інших модулів	Стабільна робота системи Уникнення просадок напруги Підключення на лінію живлення електролітичного конденсатора 16V 2000μF
Raspberry Pi 2 Model B OpenHAB 2	Raspberry Pi 3 Model B OpenHAB 3.0
Неможливо запрограмувати мікроконтролер з дисплеєм через відсутню бібліотеку	На сервер завантажити оновлену версію роботи з дисплеєм та додати приклади коду під нову версію
Датчики струму та напруги неправильно підключені Показники зчитуються невірно, часто виникають викиди та стрибки даних	Налагодити правильність підключення датчиків Оптимізувати код для вірного зчитування Додати до коду фільтри середнього арифметичного значення та медіанний фільтр трьох значень Показники мають відображати реальні значення
Відсутність експерименту з розумною теплицею	Додати новий експеримент Розумна теплиця

3 КОНСТРУЮВАННЯ ПРОГРАМНО-АПАРАТНОГО ЗАБЕЗПЕЧЕННЯ ДЛЯ МОДЕРНІЗАЦІЇ ВЛ

3.1 Алгоритми функціонування

Для покращення якості електроживлення підключено на лінію живлення електролітичний конденсатор 16 V ємністю 2000 μ F. Таким чином це згладжує просадки напруги, що виникають при поворотах валу сервоприводу. В результаті робота одного двигуна не позначається артефактами на інших компонентах. Оскільки окрім цього було додано нову лінію живлення шляхом додавання додаткового блока живлення на 2 А. Це покращує яскравість дисплея, та в парі з конденсаторами зменшує вплив роботи одного компонента на інші.

В експерименті з дисплеєм було додано нову бібліотеку. Оскільки сама бібліотека є частиною програмного коду, вона має бути десь завантажена. В даному випадку вона власне й знаходиться на сервері. В ході відлагоджування випробовувався спосіб використанням бібліотеки шляхом безпосереднього розміщення її коду перед головним циклом програми. Таким чином на даний момент цей спосіб залишається єдиним доступним для використання сторонніх бібліотек. Проте в цьому нема необхідності, адже розміщена на сервері оновлена версія бібліотеки надає увесь необхідний функціонал. Для її використання необхідно стандартним способом підключити її в скетчі та використовувати наявні в ній функції, адже бібліотека буде автоматично завантажена для компіляції, як би це було зроблено при локальному завантаженні коду. Загалом, алгоритм функціонування експерименту з дисплеєм наведено на рисунку 3.1.

Експерименти з вимірюванням напруги функціонує наступним чином. До мікроконтролеру під'єднується резистивний дільник напруги, що в свою чергу має контакт з дротом, на якому й проводиться вимірювання. Далі проводяться обчислення для переведення отриманих даних в значення напруги.

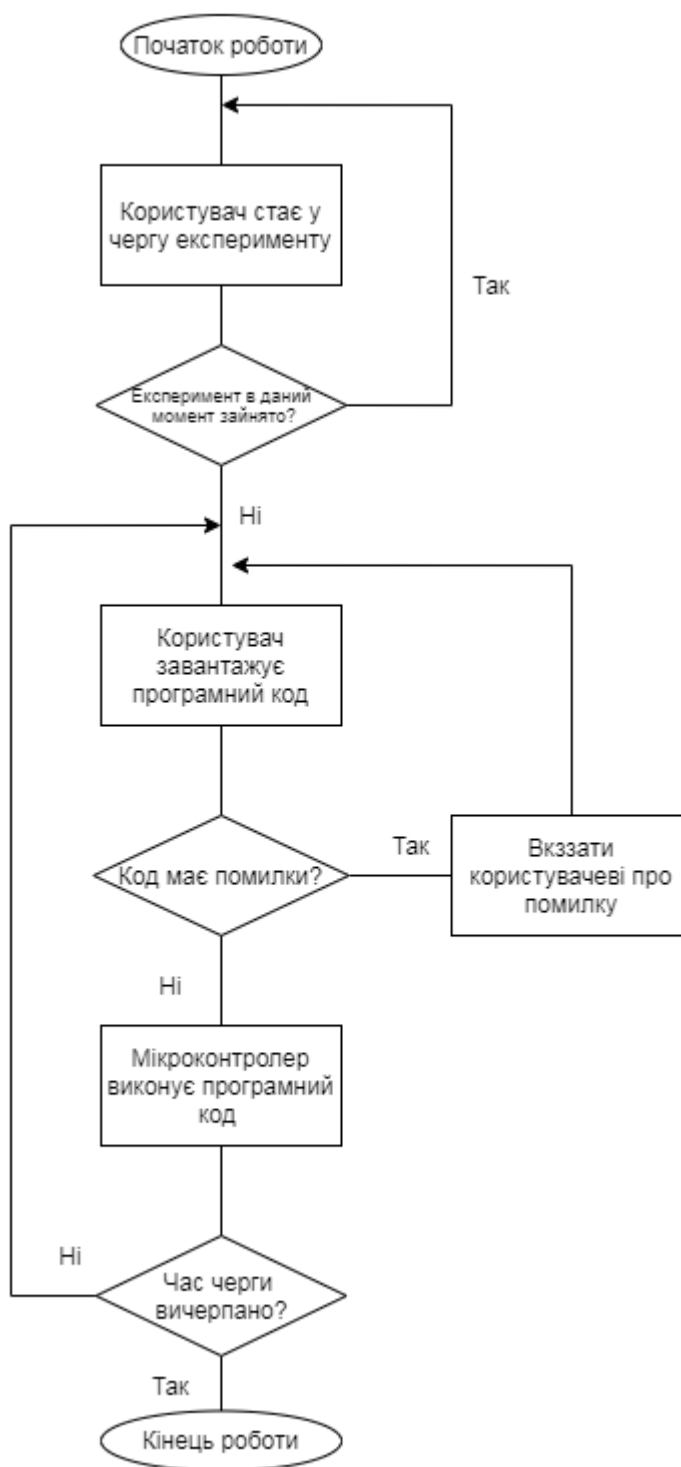


Рисунок 3.1 – Алгоритм функціонування експерименту з дисплеєм

Для експериментів з вимірюванням сили струму, окрім переводу отриманих даних з АЦП в амperi, необхідно вхідні дані направити на подвійну фільтрацію, в результаті якої отримуються очищені від викидів

дані, що дозволяють досить точно відслідкувати використання енергії на певній лінії живлення.

Експеримент з розумною теплицею працює наступним чином. Мікрокомп'ютер на основі Raspberry Pi, на якому встановлена система домашньої автоматизації OpenHAB опитує мікроконтролери Arduino Nano за допомогою протоколу Modbus RTU. В якості майстру й виступає Raspberry Pi, що для використання даного протоколу використовує конвертер, що переводить інтерфейс USB в інтерфейс RS-485, який використовується для передачі даних в протоколі Modbus RTU. Далі лінії даних під'єднані до конвертеру TTL-RS485, адже Arduino має підтримку протоколу TTL. Таким чином реалізоване спілкування між Arduino та OpenHAB на Raspberry Pi.

Сам мікроконтролер постійно опитує датчики, переводить отримані дані в зручний для передачі формат та проводить необхідні обчислення. Також при зверненні майстра по протоколу Modbus RTU передає отримані результати, а також виконує необхідні команди в залежності від отриманих команд. Наприклад виконує поворот валу серводвигуна на вказаний майстром кут для керування вентиляцією, а також вмикає та вимикає штучне освітлення та полив рослин.

Для спілкування з водонепроникним датчиком температури DS18B20 використовується двунаправлена шина зв'язку для пристроїв з передачею даних на низькій швидкості 1-Wire. Для його використання замість паразитного живлення, при якому для живлення використовується шина даних, використовується спосіб з 3 дротами. Між лінією живлення та лінією даних розміщено резистор номіналом 10 кОм.

Окрім цього, використовується також протокол I²C, що дозволяє використовувати датчик вологості та температури.

Для зчитування показань аналогових датчиків використовуються аналогові піни з АЦП. З їх допомогою отримуються дані з датчика освітленості та датчика вологості ґрунту.

Таким чином, дані зчитуються увесь час, а на вимогу керуючого пристрою мікроконтролер виконує роботу з реле та сервоприводом, а також передає отримані дані з датчиків. Загалом, алгоритм функціонування експерименту Розумна теплиця наведено на рисунку 3.2.



Рисунок 3.2 – Алгоритм функціонування експерименту Розумна теплиця

3.2 Опис особливостей апаратної реалізації

В результаті досліджень, для вирішення проблеми низької потужності було додано додаткове джерело живлення, а для проблеми пікових навантажень – електролітичний конденсатор ємністю 2000 μ F. Таким чином, апаратна реалізація, отримана в результаті, зображена на рисунку 3.3, оновлені компоненти додано в рамки.

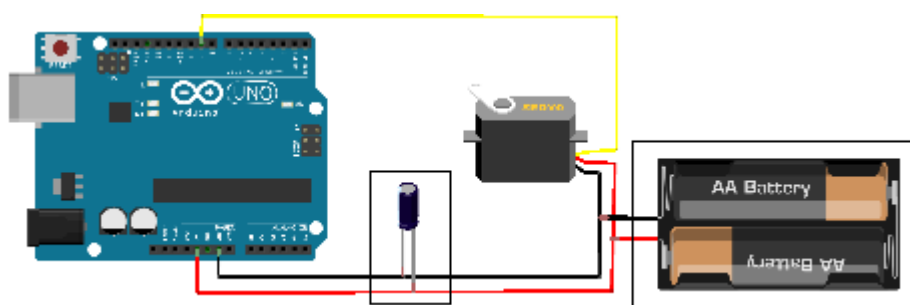


Рисунок 3.3 – Апаратна реалізація експерименту з серводвигуном після модернізації

Аналогічним чином було вирішено й проблему з низькою потужністю експерименту з дисплеєм – за допомогою нової лінії живлення. В результаті, апаратна реалізація після модернізації зображено на рисунку 3.4, оновлені компоненти додано в рамки.

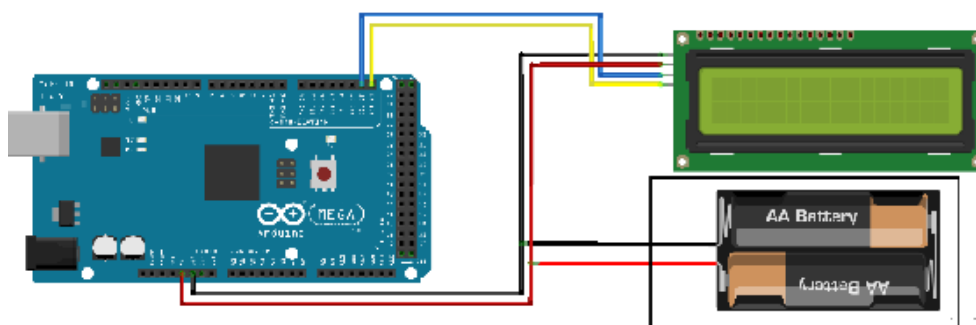


Рисунок 3.4 – Апаратна реалізація експерименту з дисплеєм після модернізації

В ході роботи налаштовано роботу датчиків струму та напруги. В якості модулю струму використовується датчик струму acs712, що може вимірювати струм в діапазоні від 0 до 30 ампер. Окрім цього, для зчитувань напруги використовується резистивний дільник напруги, з використанням якого з'являється можливість зчитувати показники напруги в діапазоні від 0 до 12 вольтів. Таким чином, апаратна реалізація цих датчиків зображена на рисунку 3.5.

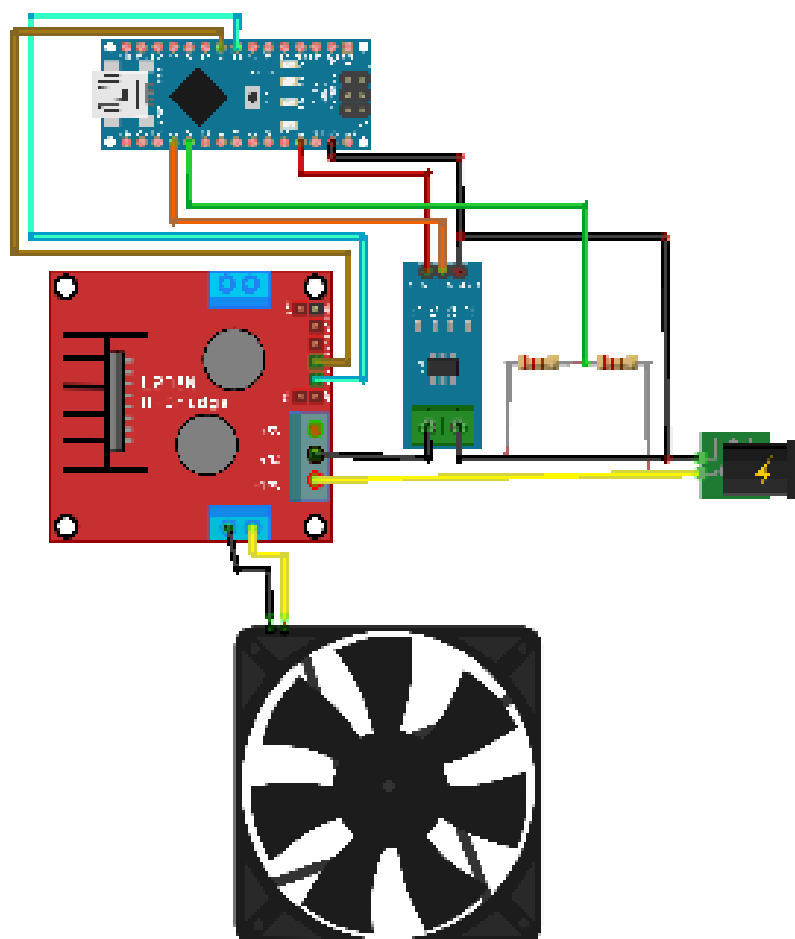


Рисунок 3.5 – Апаратна реалізація датчиків для зчитування показників напруги та струму

Загалом, мікроконтролер та усі датчики і модулі під'єднано до спільної лінії живлення 5 В. Нормально розімкнуті реле вмикаються при подачі на пін сигналу низької напруги 0 В. Таким чином, для увімкнення

зовнішнього освітлення та поливу необхідно на піни відповідно 8 та 9 подати сигнал LOW.

Для зв'язку з мікрокомп'ютером Raspberry Pi, на якому встановлено систему домашньої автоматизації OpenHAB, використовується лінія RS-485, для функціонування якої використовуються конвертери RS-485 to TTL та RS-485 to USB. Для цього до піна №2 мікроконтролера під'єднано сигнальний дріт.

Серводвигун в свою чергу під'єднано до цифровому піну №10, який використовуючи апаратні переривання за допомогою бібліотеки виконує повороти валу на вказаний кут.

Серед усіх можливих способів живлення Arduino, наприклад через USB, через порт VIN та через пін 5 В обрано останній спосіб. Це зумовлено тим, що задля захисту від короткого замикання комп'ютера на порт USB розміщено захисний діод, на якому йде просадка напруги в кулька десятих вольтів. Це може бути критично для певних датчиків. Порт VIN не було обрано, адже таким чином вхідна напруга проходить стабілізацію через окрему мікросхему, внаслідок чого виділяється зайве тепло та знижується ефективність. Таким чином найоптимальнішим варіантом і є під'єднання загальної лінії живлення разом з усіма модулями до піну 5 В.

Ємнісний датчик вологості ґрунту та датчик освітленості для прийому аналогових сигналів під'єднано до портів з АЦП A0 та A1 відповідно. Таким чином в результаті отримується сигнал в діапазоні від 0 до 1023, що потім переводиться у відповідні одиниці вимірювання.

Водонепроникний датчик температури, на відміну від дводротового «паразитного» способу підключення, при якому сигнальний дріт виступає також і лінією живлення, в даному експерименті використовується окремий дріт живлення, який пов'язаний з сигнальним за допомогою резистора номіналом 10 кОм. Для зв'язку з мікроконтролером використовується протокол 1-Wire, при якому сигнальний дріт під'єднано до піну №7. Таким

чином реалізована можливість зчитувати значення температури в місцях з підвищеною вологістю.

Датчик вологості та температури для зв'язку з мікроконтролером використовує протокол I²C, тому використовує два дроти – SDA і SCL, що в свою чергу під'єднано до аналогових пінів A4 та A5 відповідно.

В результаті отримується цілісна система з різноманітною взаємодією окремих елементів. Таким чином, апаратна реалізація експерименту Розумна теплиця зображена на рисунку 3.6.

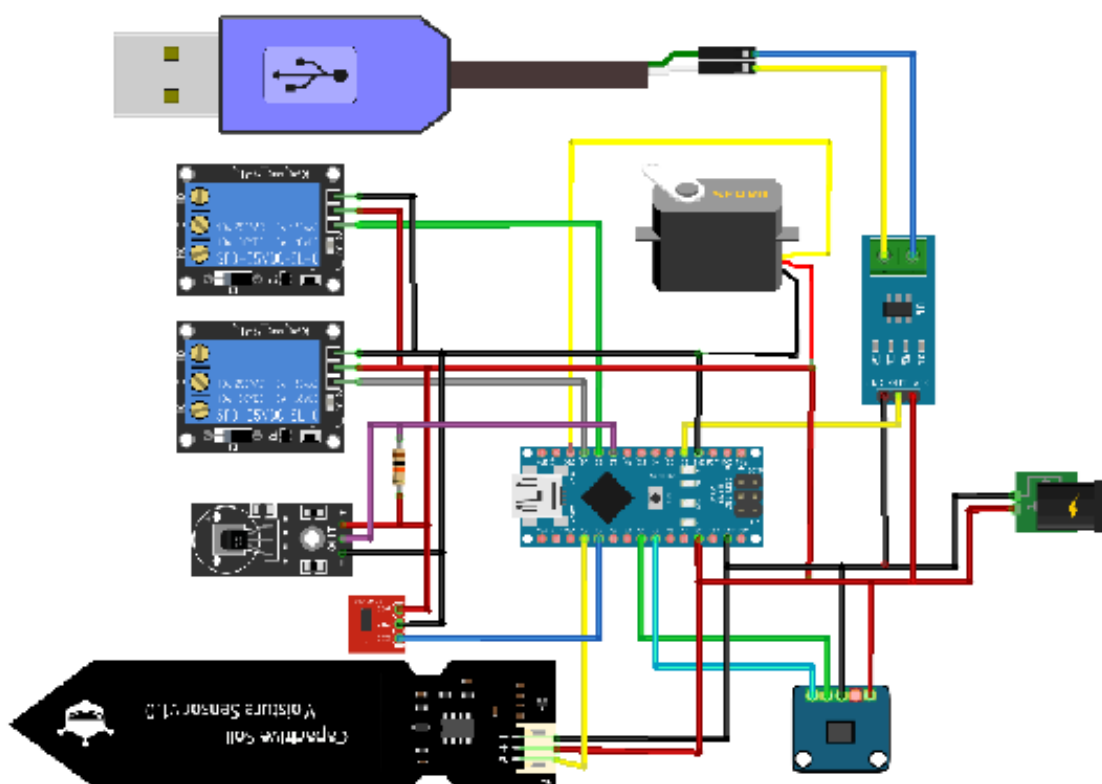


Рисунок 3.6 – Апаратна реалізація експерименту Розумна теплиця

В ході модернізацій, було зроблено макет розумної теплиці, що готовий до під'єднання до OpenHAB з інтерфейсом Modbus, приклад якого зображено на рисунку 3.7.

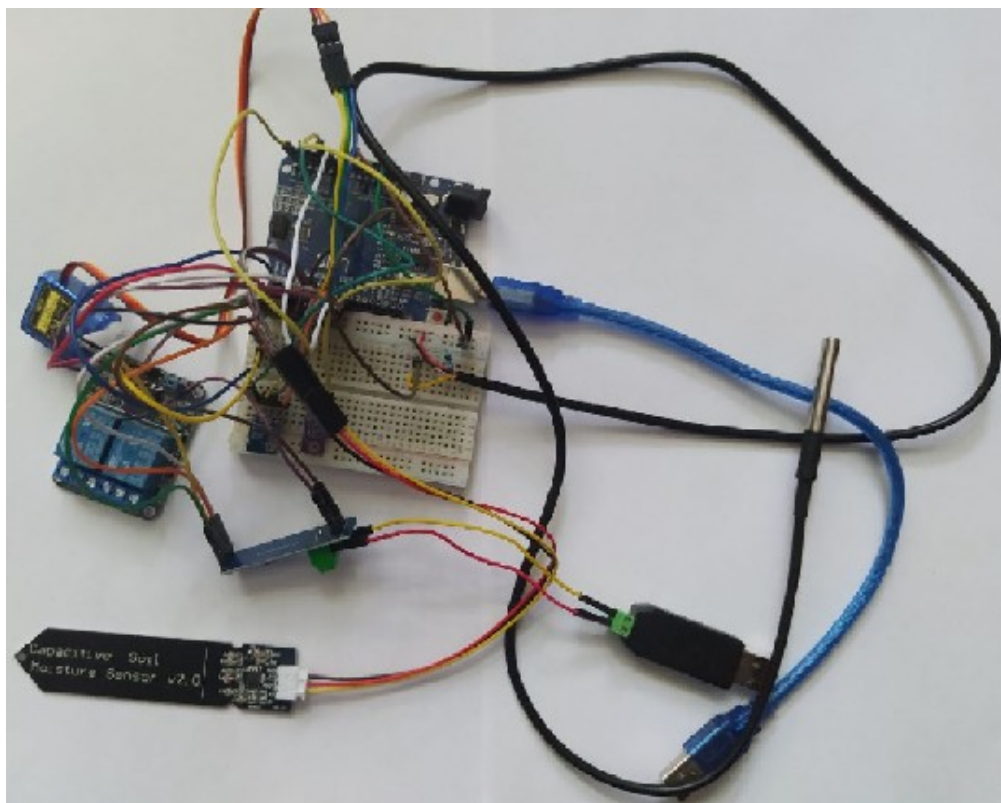


Рисунок 3.7 – Макет експерименту Розумна теплиця

3.3 Опис особливостей програмної реалізації

В ході модернізації, для експерименту з дисплеями було на сервер додано бібліотеку. Окрім того, що вона є заміною втраченій версії, більш нова є розвиненішою, містить більший функціонал та покращену швидкодію. Від попередньої версії насамперед відрізняється ініціалізацією об'єкта рідкокристалічного дисплея.

В попередній версії ініціалізація виконується наступним чином:

```
LiquidCrystal_I2C lcd(0x27, 2, 1, 0, 4, 5, 6, 7, 3, POSITIVE);
```

В новій версії використовується така конструкція:

```
LiquidCrystal_I2C lcd(0x27, 16, 2);
```

Для експериментів з використанням резистивного дільника напруги виконуються певні математичні обчислювання. Оскільки аналоговий пін приймає значення в діапазоні від 0 до 1023, що відповідає 0 та 5 вольтів

відповідно, при зчитуванні показань в більшому діапазоні необхідно отримане значення множити на максимально можливе з отримуваних показань та ділити на 1023. Алгоритм роботи програми зчитування показань напруги наведено на рисунку 3.8.



Рисунок 3.8 – Алгоритм роботи програми зчитування показань напруги

Таким чином, значення напруги вираховується за формулою (3.1):

$$V = \frac{V_{\max} * d}{1023}, \quad (3.1)$$

де V – зчитане значення напруги;

$V_{\max}$ – максимально можливе значення напруги;

d – отримане значення з аналогового піну.

Експерименти з використанням датчиків *acs712*, що дозволяють отримати значення сили струму, вимагають попередньої фільтрації отриманих даних. По-перше, слід зазначити, що використовуються 2 типи датчиків – на 5А та на 30А, а отже необхідно виконувати дещо різні коефіцієнти.

Спочатку отриманий з АЦП сигнал фільтрується. Першим виступає фільтр середнього арифметичного значення. В буфері записується 10 попередніх значень останніх вимірювань для усереднення. При кожному новому виклику буфер зсувається на 1 позицію і в нього додається нове зчитане значення, а також видаляється найстаріше. Далі буфер усереднюється за середнім арифметичним.

Окрім цього фільтру використовується ще й медіанний, адже отримане усереднене значення записується до нового буферу трьох змінних. В ході даної фільтрації знаходиться значення, що є меншим або відповідним іншим двом, а в результаті повертається найменше з цих інших.

В результаті отримується значення в діапазоні від 0 до 1023. Згідно з даташитом на модуль *acs712* [46], базовим значенням напруги датчика є 2.5 В. діапазони для вимірювань являється з 1.5 В до 3.5 В датчика на максимальних 5 А, та з 0.5 В до 4.5 В для датчику на 30 А. Алгоритм роботи програми зчитування показань сили струму наведено на рисунку 3.9.



Рисунок 3.9 – Алгоритм роботи програми зчитування показань сили струму

Таким чином, отримане значення з АЦП необхідно відняти від базового значення 512 та отримане значення поділити на відповідний коефіцієнт, що вираховується за формулою (3.2):

$$A = \frac{1023 * k}{V_{\max}}, \quad (3.2)$$

де A – зчитане значення сили струму;

$V_{\max}$ – максимально можливе значення напруги (5000 мілівольт);

k – коефіцієнт, що становить 185 та 66 для датчиків на 5 А та 30 А відповідно.

В результаті надається можливість отримати значення сили струму з досить високою точністю для бажаних експериментів.

Програмне забезпечення для експерименту Розумна теплиця функціонує наступним чином. Спочатку підключаються необхідні бібліотеки та проводиться початкова ініціалізація змінних. Потім проходить налаштування системи, такі як ініціалізація підключень та увімкнення портів мікроконтролера.

В нескінченному циклі спочатку проводиться поворот валу серводвигуна та увімкнення чи ввімкнення реле у відповідності до поточного стану. Далі проводиться зчитування з усіх датчиків за допомогою аналогових пінів та інтерфейсів 1-Wire та I²C.

Отримані дані оброблюються для подальшої передачі протоколом Modbus. Це виконується, адже отримуються дані іноді в форматі цифр з плаваючою комою, тому необхідно виділити цілу та дробну частини.

Далі виконується спілкування з ведучим пристроєм, внаслідок якого виконується зчитування показань з датчиків та задається стан реле та кут серводвигуна.

Алгоритм роботи програми експерименту Розумна теплиця наведено на рисунку 3.10.



Рисунок 3.10 – Алгоритм роботи програми експерименту Розумна теплиця

Перелік змінних для моніторингу струму та напруги наведено в таблиці 3.1.

Таблиця 3.1 – Опис змінних для моніторингу струму та напруги

Змінна	Призначення
1	2
const int voltage;	Змінна збереження номеру піну підключення датчика напруги
Serial.begin();	Ініціалізація послідовного з'єднання та задавання швидкості передачі даних в біт/с (бод)
#define TXEN	Константа збереження номеру піну підключення модуля конвертеру для протоколу Modbus
Modbus slave(,,);	Ініціалізація об'єкту для використання шини протоколу Modbus
uint16_t aul6data[15];	Масив зберігання значень для передачі по Modbus
float valueV;	Змінна збереження показань напруги, зчитаних з аналогового входу
#define NUM_AVER5A	Константа збереження кількості елементів вибірки для фільтрації даних з датчику струму з максимумом 5 А
#define NUM_AVER30A	Константа збереження кількості елементів вибірки для фільтрації даних з датчику струму з максимумом 30 А
byte index5A;	Індекс вибору елементу масиву медіанного фільтру датчику струму на максимальних 5 А
byte index30A;	Індекс вибору елементу масиву медіанного фільтру датчику струму на максимальних 30 А
const int acs712_pin5A;	Змінна збереження номеру піну підключення датчика струму на максимальних 5 А
const int acs712_pin30A;	Змінна збереження номеру піну підключення датчика струму на максимальних 30 А
int arr5A[3];	Масив збереження значень елементів медіанного фільтру датчику струму на максимальних 5 А
int arr30A[3];	Масив збереження значень елементів медіанного фільтру датчику струму на максимальних 30 А
float scale5A;	Змінна збереження величини коефіцієнту, скільки одиниць АЦП відповідає на 1 ампер датчику струму на максимальних 5 А

Продовження таблиці 3.1.

1	2
float scale30A;	Змінна збереження величини коефіцієнту, скільки одиниць АЦП відповідає на 1 ампер датчику струму на максимальних 30 А
int zero;	Змінна збереження величини значення рівня нуля, відносно якого вимірюється струм
const float coefV;	Змінна збереження величини коефіцієнту, скільки вольтів відповідає на 1 одиницю АЦП

Перелік функцій для моніторингу струму та напруги наведено в таблиці 3.2.

Таблиця 3.2 – Опис функцій для експерименту вимірювання струму та напруги

Функція	Призначення
slave.begin();	Ініціалізація з'єднання та задавання швидкості передачі даних в біт/с (бод) для передачі даних по Modbus
void setup();	Стандартна функція, яка викликається лише один раз. Зазвичай в ній проходить першочергове налаштування системи та ініціалізація усіх компонентів, що виконуються після увімкнення мікроконтролера
void loop();	Стандартна функція, в якому виконується основна робота мікроконтролера в нескінченному циклі
slave.poll();	Функція обробки отриманого сигналу по Modbus
int getSmoothedValue30A()	Функція отримання згладжених значень з АЦП Arduino датчику струму на максимальних 30 А
int getCurrent(int adc, float scale)	Функція вираховування значення сили струму у відповідності до показань з АЦП
float middle_of_3(float a, float b, float c)	Функція медіанного фільтру трьох значень
Serial.print();	Функція виводу показань в послідовний монітор

Перелік змінних для експерименту Розумна теплиця наведено в таблиці 3.3.

Таблиця 3.3 – Опис змінних для експерименту Розумна теплиця

Функція/змінна	Призначення
----------------	-------------

1	2
#define pinRelay1	Константа збереження номеру піну підключення модуля реле №1
#define pinds18b20	Константа збереження номеру піну підключення водонепроникного датчика температури
#define pinRelay2	Константа збереження номеру піну підключення модуля реле №2
#define pinCapacitiveSoil	Константа збереження номеру піну підключення ємнісного датчика вологості ґрунту
#define pinTemt6000	Константа збереження номеру піну підключення датчика освітленості
#define pinServo	Константа збереження номеру піну підключення модуля серводвигуна
#define TXEN	Константа збереження номеру піну підключення модуля конвертеру для протоколу Modbus
SHT2x;	Ініціалізація об'єкту для роботи датчика вологості та ґрунту
Servo;	Ініціалізація об'єкту для роботи модулю серводвигуна
OneWire;	Ініціалізація об'єкту для використання шини 1-Wire
Modbus slave(,,);	Ініціалізація об'єкту для використання шини протоколу Modbus
byte valueRelay1flag;	Змінна зберігання стану реле №1
byte valueRelay2flag;	Змінна зберігання стану реле №2
int valueServo;	Змінна зберігання поточного кута повороту серводвигуна
int valueCapacitiveSoil	Змінна зберігання поточного стану вологості ґрунту
float valueSHT2xHumidity;	Змінна зберігання поточного стану вологості повітря
float valueSHT2xTemperature;	Змінна зберігання поточного стану температури повітря
int valueTemt6000;	Змінна зберігання поточного стану освітленості

Продовження таблиці 3.3.

1	2
float value_ds18b20;	Змінна зберігання поточного стану температури ґрунту
uint32_t myTimer1;	Змінна для керування таймерами часу
uint16_t au16data[15];	Масив зберігання значень для передачі по Modbus

Перелік функцій для експерименту Розумна теплиця наведено в таблиці 3.4.

Таблиця 3.4 – Опис функцій для експерименту Розумна теплиця

Функція	Призначення
float temperature_ds18b20();	Функція опитування датчика водонепроникного температури на ds18b20
void setup();	Стандартна функція, яка викликається лише один раз.

	Зазвичай в ній проходить першочергове налаштування системи та ініціалізація усіх компонентів, що виконуються після увімкнення мікроконтролера
<code>void loop();</code>	Стандартна функція, в якому виконується основна робота мікроконтролера в нескінченному циклі
<code>slave.begin();</code>	Ініціалізація з'єднання та задавання швидкості передачі даних в біт/с (бод) для передачі даних по Modbus
<code>SHT2x.begin();</code>	Ініціалізація з'єднання для опитування датчиків температури та вологості
<code>myservo.attach();</code>	Ініціалізація серводвигуна на конкретному піні мікроконтролеру
<code>myservo.write();</code>	Завдання кута повороту валу серводвигуна
<code>pinMode();</code>	Налаштування піну на вхід/вихід
<code>digitalWrite();</code>	Передача високого або низького сигналу на цифровий піл
<code>analogRead();</code>	Функція зчитування високого або низького сигналу з цифрового піну
<code>SHT2x.GetHumidity();</code>	Отримання значення вологості з датчика
<code>HT2x.GetTemperature();</code>	Отримання значення температури з датчика
<code>slave.poll();</code>	Функція обробки отриманого сигналу по Modbus

Перелік бібліотек для експерименту Розумна теплиця наведено в таблиці 3.5.

Таблиця 3.5 – Опис бібліотек для експерименту Розумна теплиця

Бібліотека	Призначення
<code>#include <OneWire.h></code>	Підключення бібліотеки 1-Wire для використання датчика 18b20
<code>#include <Wire.h></code>	Підключення бібліотеки для роботи з I ² C для використання датчика вологості та ґрунту
<code>#include "SHT2x.h"</code>	Підключення бібліотеки для функціонування датчика вологості та ґрунту
<code>#include <Servo.h></code>	Підключення бібліотеки для використання модулю серводвигуна
<code>#include "ModbusRtu.h"</code>	Підключення бібліотеки для використання лінії RS-485 для протоколу Modbus

3.4 Висновки до розділу

В результаті роботи було описано програмні та апаратні компоненти модернізації ВЛ, а саме: описано функції, бібліотеки та змінні, що використовувалися в програмному коді, наведено та описано схеми алгоритмів функціонування програм та експериментів лабораторії. Також наведено та описано схеми апаратної реалізації експериментів, наведено фото макету експерименту Розумна теплиця та описано особливості функціонування програми.

4 КЕРІВНИЦТВО ПРОГРАМІСТА

4.1 Призначення та умови застосування програмно-апаратного забезпечення

Проведена робота та розроблені програмні та апаратні компоненти виступають в якості модернізації конкретного експерименту для віддаленої лабораторії RELDES або для усієї лабораторії загалом.

Основне призначення – поліпшити існуючий функціонал ВЛ, підвищити наочність, усунути наявні проблеми, а також додати новий експеримент, що в результаті надає конкретному користувачеві можливість зручно проводити дослідження та отримувати дані.

Розроблене програмно-апаратне забезпечення модернізацій стосується RELDES та Smart House & IoT. В результаті роботи налагоджено роботу експериментів у відповідності до поставлених вимог.

Для проведення експериментів з ВЛ необхідно мати комп'ютер з швидкісним Інтернет-з'єднанням, що задовольняє наступні апаратно-технічні вимоги:

- ОС Microsoft Windows 7 або вище;
- процесор з частотою 0.1 ГГц або вище;
- RAM 512 МБ або вище;
- відеокарта з підтримкою роздільної здатності екрану 1280 x 720 пікселів та вище;
- швидкісне інтернет-з'єднання від 1 Мбіт/с або вище;
- клавіатура;
- миша.

4.2 Характеристики програми

Розробка програмних застосунків для плат проводилася високорівневою мовою програмування C++ з використанням інтегрованого

середовища розробки для створення та завантаження програм на Arduino-сумісні плати. Розроблені програмні застосунки призначені для усіх Arduino-сумісних плат, зокрема для плати Arduino Nano.

Модернізація програмного застосунку для вимірювання сили струму та напруги дозволяє проводити вимірювання відповідних параметрів у усієї системи загалом та для певних потужних компонентів.

Модернізація апаратного забезпечення направлене на покращення потужності живлення системи.

Модернізація експерименту з відкривання дверей за допомогою електронного ключа дозволяю проаналізувати роботу макету реальної автоматизованої системи безпеки доступу до будинку.

Модернізація експерименту з дисплеєм проведена шляхом додавання до серверу нової версії для роботи даного модулю та переписано код прикладів.

Експеримент Розумна теплиця готовий до інтеграції в OpenHAB за інтерфейсом Modbus та надає можливість проводити наступні дослідження:

- вимірювати температуру ґрунту;
- вимірювати вологість ґрунту;
- вимірювати температуру повітря теплиці;
- вимірювати вологість теплиці;
- вимірювати освітленість теплиці;
- вмикати зовнішнє освітлення теплиці;
- вмикати полив ґрунту теплиці;
- керувати вентиляцією.

4.3 Звернення до програми

Для проведення дослідження в лабораторії спочатку необхідно бути авторизованим на сайті лабораторії. Для цього при вході на сайт одразу надається форма для реєстрації та авторизації. Потім необхідно перейти до

обраного експерименту та почати його проведення ставши у чергу. При настанні черги користувача буде запропоновано почати дослідження та буде надано доступ до маніпуляцій та результатів експерименту.

4.4 Вхідні та вихідні дані

В якості вхідних даних виступають показники значень датчиків, що зчитуються мікроконтролерами, які потім передаються до OpenHAB.

Окрім цього вхідними даними є код користувачів, що написаний для конкретного експерименту лабораторії RELDES.

Вихідними даними також є відеозображення з камери, що демонструє функціонування конкретного експерименту.

Таким чином, для експерименту Розумна теплиця вхідними даними є показники з відповідних датчиків та інструкції про ввімкнення та вимкнення поливу та освітлення, а також вентиляції. Форму введення необхідних параметрів експерименту Розумна теплиця наведено на рисунку 4.1.

4.5 Повідомлення

В якості повідомлень виступає інформація з екранної форми завантаження коду для певного експерименту лабораторії RELDES.

Наприклад, на рисунку 4.2 зображено приклад повідомлення про успішне завантаження програмного коду.

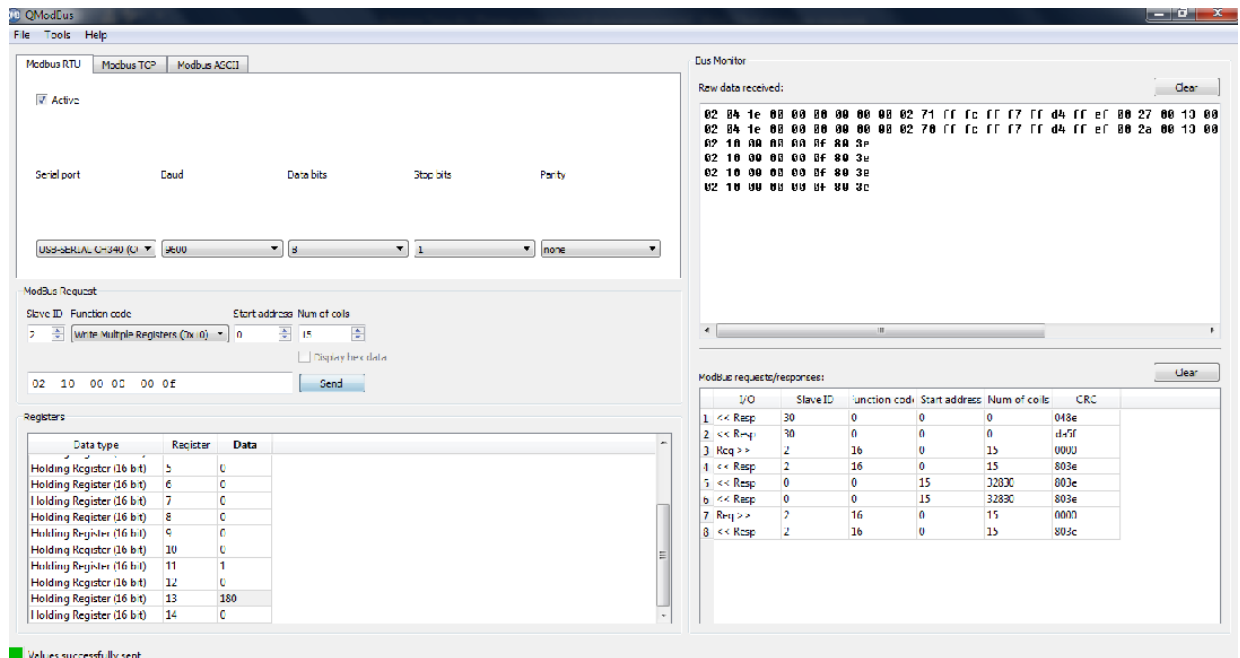


Рисунок 4.1 – Приклад внесення необхідних параметрів в експерименті Розумна теплиця

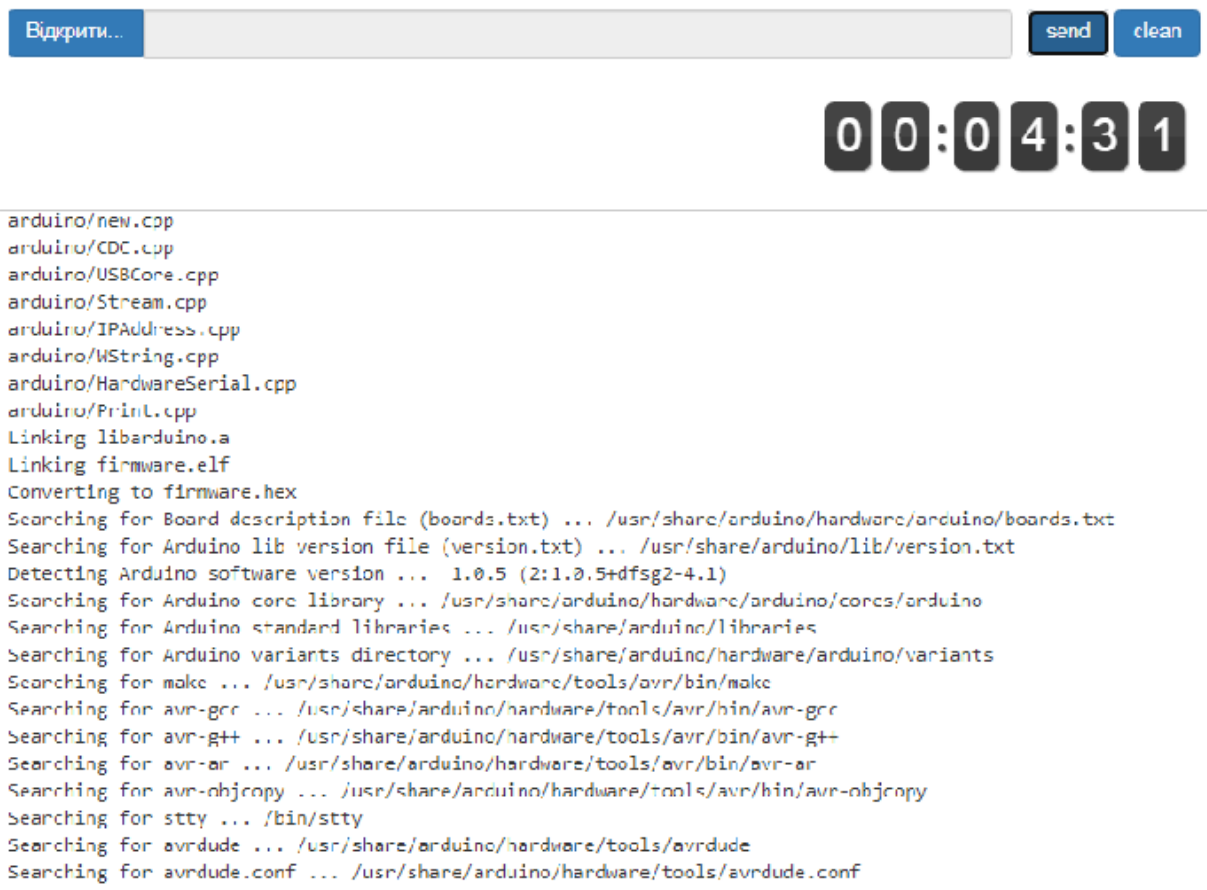
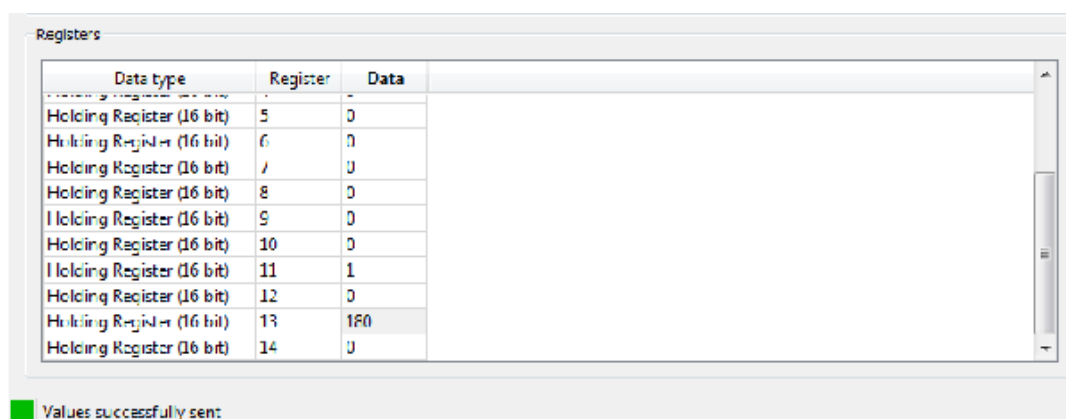


Рисунок 4.2 – Приклад повідомлення про правильність компіляції та завантаження коду в лабораторії RELDES

При успішному спілкуванні з мікроконтролером за допомогою протоколу Modbus буде виведене повідомлення, як на рисунку 4.3.



Data type	Register	Data
Holding Register (16 bit)	5	0
Holding Register (16 bit)	6	0
Holding Register (16 bit)	7	0
Holding Register (16 bit)	8	0
Holding Register (16 bit)	9	0
Holding Register (16 bit)	10	0
Holding Register (16 bit)	11	1
Holding Register (16 bit)	12	0
Holding Register (16 bit)	13	180
Holding Register (16 bit)	14	0

Values successfully sent

Рисунок 4.3 – Приклад успішного спілкування з мікроконтролером за допомогою протоколу Modbus

5 КЕРІВНИЦТВО КОРИСТУВАЧА

5.1 Призначення програмно-апаратного забезпечення

Розроблений комплекс модернізації віддаленої лабораторії направлений на покращення функціоналу існуючих програмно-апаратних засобів, усунення наявних проблем та збільшення асортименту наявних дослідів. Апаратна модернізація стосується в цілому покращення електроживлення системи, перевірка правильності підключення модулів та заміна мікрокомп'ютера, програмна – додавання бібліотеки роботи з дисплеєм, створення нового експерименту Розумна теплиця, оновлення системи автоматизації OpenHAB та правильність зчитування показань з датчиків струму та напруги. В результаті ВЛ працює стабільно та правильно у відповідності до вимог.

5.2 Умови виконання програм

Серед умов використання є наступні:

- стенд ВЛ в даний момент увімкнено;
- мережа ВЛ в даний момент має доступ до інтернет;
- на сайті або в лабораторії не проводяться технічні роботи;
- користувач авторизований у системі.

Окрім цього, для проведення дослідів з віддаленою лабораторією необхідно мати комп'ютер з швидкісним інтернет-з'єднанням, що задовольняє наступні апаратно-технічні вимоги:

- ОС Microsoft Windows 7 або вище;
- процесор з частотою 0.1 ГГц або вище;
- RAM 512 МБ або вище;
- відеокарта з підтримкою роздільної здатності екрану 1280 x 720 пікселів та вище;
- швидкісне інтернет-з'єднання від 1 Мбіт/с або вище;

- клавіатура;
- миша.

5.3 Виконання програм

Програма роботи з дисплеєм працює з використанням бібліотеки, що знаходиться на сервері. Таким чином, для роботи коду з дисплеєм слід використовувати саме ту версію бібліотеки, що й була завантажена на сайт. На рисунку 5.1 зображено приклад виконання програми з дисплеєм.

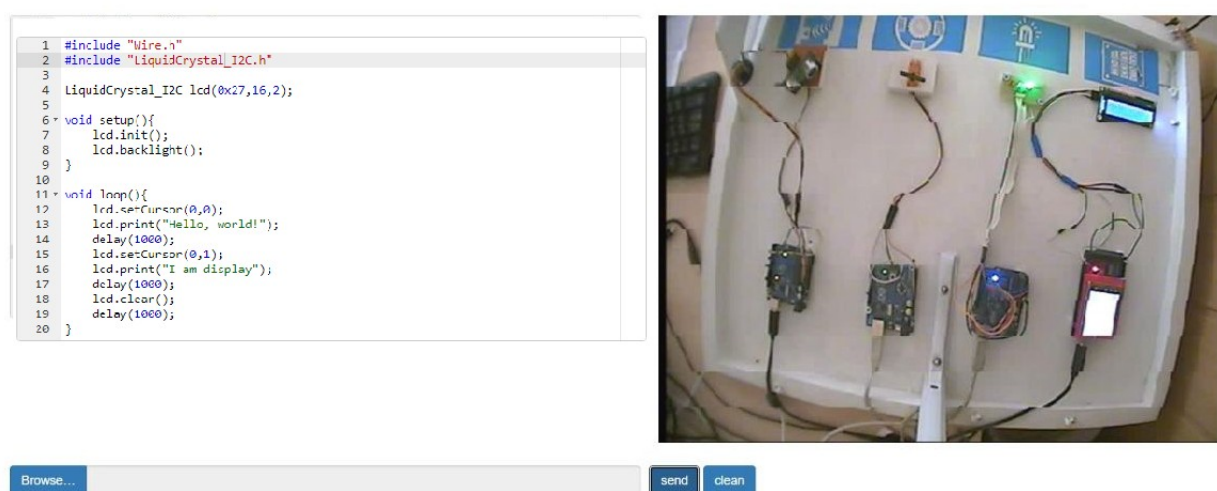


Рисунок 5.1 – Приклад виконання програми з дисплеєм

Сила струму перевіряється датчиками, що вимірюють дані показники для усього стенду Розумного будинку в цілому, а також для експерименту з вентиляцією, що також перевіряє значення поточної сили струму. На рисунку 5.2 зображено приклад виконання моніторингу сили струму.

Виконання експерименту Розумна теплиця супроводжується вимірюванням значень температури та вологості ґрунту і повітря, а також освітленості.



Рисунок 5.2 – Моніторинг сили струму

Також надається можливість вмикати та вимикати зовнішнє освітлення та полив, а також керувати вентиляцією. На рисунку 5.3 зображено приклад зчитування даних з експерименту Розумна теплиця за протоколом Modbus.

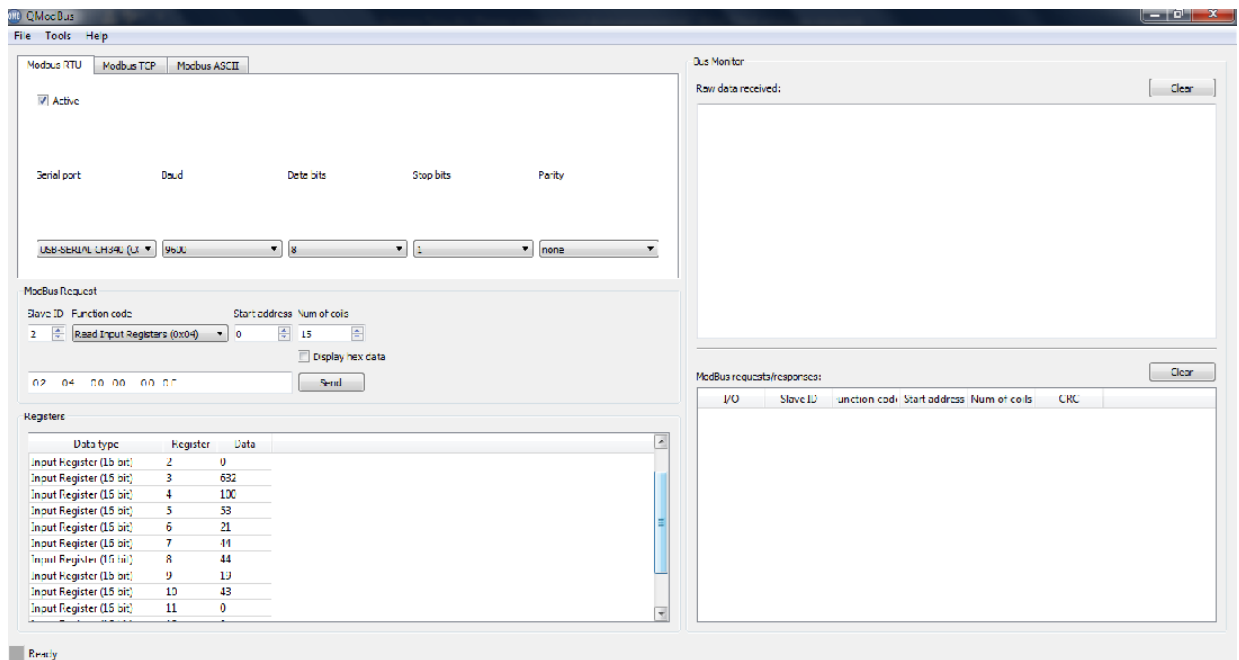


Рисунок 5.3 – Приклад зчитування даних з експерименту Розумна теплиця

5.4 Повідомлення

Повідомленнями виступає інформація з екранної форми експерименту завантаження коду для певного датчика лабораторії RELDES.

При виникненні помилкового завантаження коду, наприклад невірному коду чи відсутності бібліотеки, буде виведено стандартне повідомлення про помилку з роз'ясненням, як це зображено на рисунку 5.4.

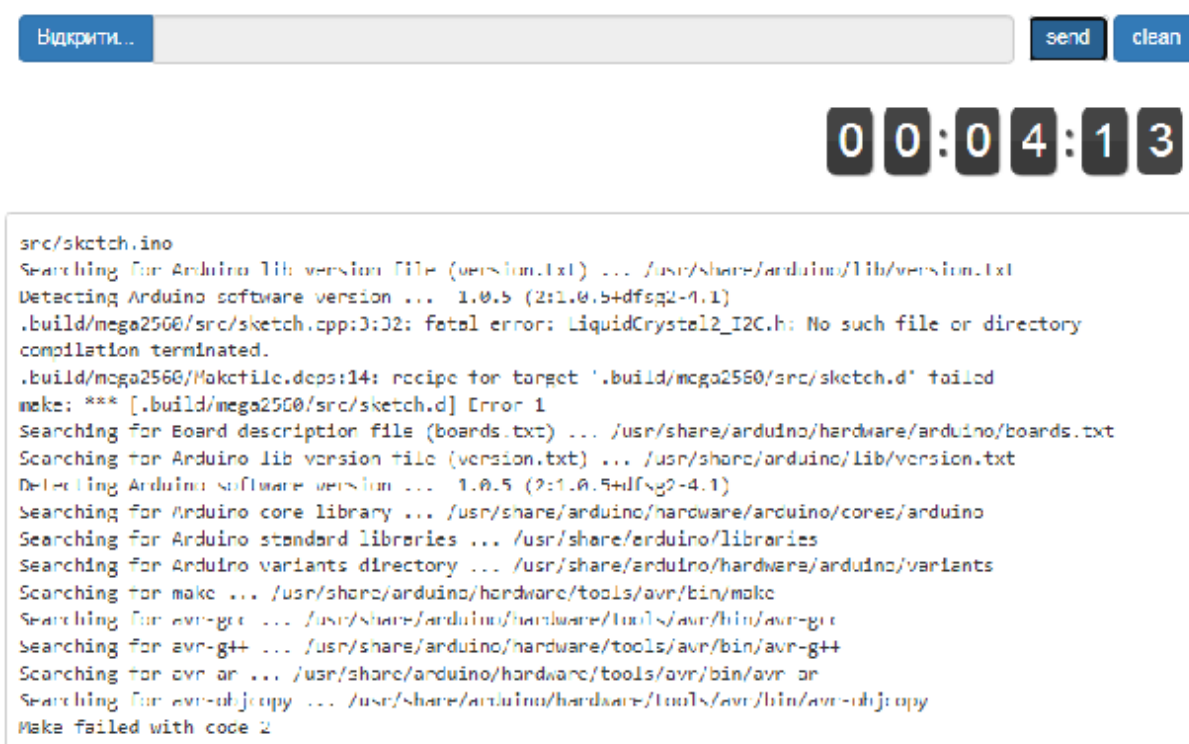


Рисунок 5.4 – Приклад повідомлення про помилку компіляції та завантаження коду в лабораторії RELDES

ВИСНОВКИ

В ході виконання дипломної роботи було проведено модернізацію апаратного та програмного забезпечення віддаленої лабораторії RELDES, в результаті якої було усунуто ряд недоліків системи, зокрема неможливість проведення експерименту з дисплеєм та вимірювання значень сили струму та напруги.

Розроблений експеримент Розумна теплиця дозволяє проводити дослідження з макетом теплиці та надає можливість вимірювати значення температури та вологості ґрунту та повітря, а також показники освітлення. Також система дозволяє керувати зовнішнім освітленням та поливом рослин, а також для регулювання температури та вологості надається можливість регулювання вентиляції. Експеримент готовий до інтеграції з системою домашньої автоматизації за протоколом Modbus.

В майбутній роботі планується подальша модернізація системи, в результаті якої має бути усунена імітація роботи датчика електронного ключа для відкривання дверей, додано новий Wi-Fi модуль для експерименту з деаутентифікацією для демонстрації загрози. Також планується замінити інфрачервоні датчики ультразвуковими або лазерними модулями, а також додати до форми на сайті можливість ознайомлення з результатами експерименту з датчиком відстані.

Усі завдання, які було поставлено у дипломній кваліфікаційній роботі бакалавра, виконано у повному обсязі.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. N+1 | Виртуальная наука [Электрон. ресурс]. – Режим доступа: <https://nplus1.ru/material/2020/10/15/virtual-laboratories>.
2. Журавлева, О. Б. Управление Интернет-обучением в высшей школе : учебное пособие / О. Б. Журавлева, Б. И. Крук, Е. Г. Соломина. – Новосибирск: Веди, 2005. – 255 с.
3. Hacking, I. Representing and Intervening: Introductory. Topics in the Philosophy of Natural Science. – Cambridge: Cambridge University Press, 1983. – 307 p.
4. Соловов, А. В. Виртуальные учебные лаборатории в инженерном образовании / А. В. Соловов. – М.: МГИУ, 2002. – 255 с.
5. Удаленная лаборатория [Электрон. ресурс]. – Режим доступа: https://ru.qaz.wiki/wiki/Remote_laboratory.
6. LabShare [Electronic resource]. – Access mode – Режим доступа: <https://www.thelabshare.com>.
7. LabsLand | What is a real laboratory? [Electronic resource]. – Access mode: <https://labsland.com/en/about#what-is-a-real-lab>.
8. LabsLand | Arduino robot (code) [Electronic resource]. – Access mode: <https://labsland.com/en/labs/arduino-robot>.
9. LabsLand | Arduino robot (visual) [Electronic resource]. – Access mode: <https://labsland.com/en/labs/arduino-visual-robot>.
10. LabsLand | FPGA Laboratory [Electronic resource]. – Access mode: <https://labsland.com/en/labs/fpga-llstd1>.
11. NetLab | Welcome to NetLab [Electronic resource]. – Access mode: <http://netlab.unisa.edu.au/index.xhtml>.
12. NetLab | Frequently Asked Questions [Electronic resource]. – Access mode: <http://netlab.unisa.edu.au/faq.xhtml>.
13. NetLab | What is NetLab? [Electronic resource]. – Access mode: <http://netlab.unisa.edu.au/about/whatIsNetLab.xhtml>.

14. Goldi-Labs | 3-Axis-Portal [Electronic resource]. – Access mode: <https://www.goldi-labs.net/index.php?Site=21>.
15. Goldi-Labs | Finite State Machine [Electronic resource]. – Access mode: <http://goldi-labs.net/index.php?Site=41>.
16. VHDL | VHDL. Эффективное использование при проектировании цифровых систем – листинг [Электрон. ресурс]. – Режим доступа: <http://vhdl.bas-net.by/archives/334>.
17. Goldi-Labs | Digital Demo Board [Electronic resource]. – Access mode: <http://goldi-labs.net/index.php?Site=33>.
18. Goldi-Labs | Microcontroller [Electronic resource]. – Access mode: <http://goldi-labs.net/index.php?Site=42>.
19. Умный дом | Что такое Умный Дом [Электрон. ресурс]. – Режим доступа: https://www.smarthouse.ua/umnyj_dom.html.
20. Умный дом | Домашняя автоматизация [Электрон. ресурс]. – Режим доступа: <https://dom-automation.ru/smart-home/avtomatizatsiya.html>.
21. Intel | Development Resources for IoT Professionals [Electronic resource]. – Access mode: <https://software.intel.com/content/www/us/en/develop/topics/iot.html>.
22. Cornell University | Smart Home System [Electronic resource]. – Access mode: https://people.ece.cornell.edu/land/courses/ece4760/FinalProjects/f2012/jw937_sz369/jw937-sz369/jw937_sz369.html.
23. Програмно-апаратна платформа для навчання технологіям інтернету речей : навчальний посібник / [А. В. Пархоменко, А. В. Туленков, О. В. Соколянський та ін.]. – Запоріжжя: Дике Поле, 2017. – 121 р.
24. Віддалена лабораторія REIoT | Wi-Fi безпека [Електрон. ресурс]. – Режим доступу: <https://swed.zntu.edu.ua/frontend/web/smart-house-experiments/iot>.
25. Віддалена лабораторія REIoT | Розумний будинок [Електрон. ресурс]. – Режим доступу: <https://swed.zntu.edu.ua/frontend/web/smart-house-experiments>.

26. Raspberry Pi | Products [Electronic resource]. – Access mode: <https://www.raspberrypi.org/products>.
27. Arduino Master | Плата Arduino Nano v 3.0 [Электрон. ресурс]. – Режим доступа: <https://arduinomaster.ru/platy-arduino/plata-arduino-nano>.
28. Robot Chip | Обзор конвертера TTL-RS485 [Электрон. ресурс]. – Режим доступа: <https://robotchip.ru/obzor-konvertera-ttl-rs485>.
29. Arduinka.Pro | Конвертер USB в RS-485 [Электрон. ресурс]. – Режим доступа: <https://arduinka.pro/progr/konverter-usb-v-rs-485>.
30. Амперка | Датчик влажности почвы (ёмкостный): инструкция по использованию и примеры [Электрон. ресурс]. – Режим доступа: <http://wiki.amperka.ru/products:sensor-soil-moisture-capacitive>.
31. All-Audio.pro | Датчики влажности почвы [Электрон. ресурс]. – Режим доступа: <https://all-audio.pro/c31/datashiti/datchiki-vlazhnosti-pochvi.php>.
32. Arduino learning | ARDUINO AND TEMT6000 LIGHT SENSOR [Electronic resource]. – Access mode: <http://arduinolearning.com/code/arduino-temt6000-light-sensor.php>.
33. 3Diy | Датчик температуры и влажности HTU-21 (GY-21) [Электрон. ресурс]. – Режим доступа: <https://3d-diy.ru/wiki/arduino-datchiki/datchik-temperature-i-vlazhnosti-gy-21/>.
34. Амперка | Герметичный датчик температуры DS18B20 [Электрон. ресурс]. – Режим доступа: <https://amperka.ru/product/sealed-temperature-sensor-ds18b20#docs>.
35. 3v3 | Сервопривод сервомашинка SG90 9g 4,8-6V 180 градусов [Электрон. ресурс]. – Режим доступа: https://3v3.com.ua/product_1070.html.
36. Robot Chip | Обзор 2-х канального модуля реле [Электрон. ресурс]. – Режим доступа: <https://robotchip.ru/obzor-modulya-rele-2-x-kanalny>.
37. Arduino | Getting Started with Arduino products [Electronic

resource]. – Access mode: <https://www.arduino.cc/en/Guide>.

38. Arduino.ru | Среда разработки Arduino [Электрон. ресурс]. – Режим доступа: http://arduino.ru/Arduino_environment.

39. Академик | Notepad++ [Электрон. ресурс]. – Режим доступа: <https://dic.academic.ru/dic.nsf/ruwiki/160636>.

40. Visual Studio Code | Getting Started [Electronic resource]. – Access mode: <https://code.visualstudio.com/docs>.

41. Platformio | Professional collaborative platform for embedded development [Electronic resource]. – Access mode: <https://platformio.org>.

42. Cxem.net | Atmel Studio [Электрон. ресурс]. – Режим доступа: https://cxem.net/software/atmel_studio.php.

43. OpenHAB | Introduction to Pages [Electronic resource]. – Access mode: https://www.openhab.org/docs/tutorial/pages_intro.html.

44. OpenHAB | Documentation [Electronic resource]. – Access mode: <https://www.openhab.org/docs>.

45. Хабр | Как общаются машины: протокол Modbus [Электрон. ресурс]. – Режим доступа: <https://habr.com/ru/company/advantech/blog/450234>.

46. Радиолис | Измеряем ток с помощью Arduino и датчика тока ACS712 [Электрон. ресурс]. – Режим доступа: <https://radiolis.pp.ua/arduino/55-datchik-toka-ac712-i-arduino/>.

ДОДАТОК А
Технічне завдання

A.1 Вступ

Метою розробки є модернізація ВЛ RELDES для усунення помилок роботи та розширення функціоналу.

A.2 Призначення розробки

Програмно-апаратне забезпечення ВЛ RELDES для проведення експериментів Розумна теплиця, Дисплей, а також моніторинг показників струму та напруги обладнання ВЛ.

A.3 Вимоги до програми або програмного виробу

A.3.1 Вимоги до функціональних характеристик

Програмно-апаратна модернізація має забезпечувати наступні функціональні можливості:

- можливість використовувати нову версію бібліотеки;
- можливість використання стенду лабораторії без просадки напруги;
- фільтрація показників датчика струму медіанним та фільтром середнього арифметичного значення з максимумом 5 вольтів та 12 вольтів;
- можливість зчитувати показники напруги;
- можливість використання нової версії Raspberry Pi та нової версії OpenHAB;
- перегляд температури, вологості та освітленості з датчиків, експерименту Розумна теплиця;
- керування освітленням, вентиляцією та поливом експерименту Розумна теплиця.

A.3.2 Вимоги до надійності

Програмне забезпечення має коректно виконувати свої функції, описані у «Вимогах до функціональних характеристик».

A.3.3 Умови експлуатації

Для використання ВЛ RELDES необхідно використовувати ПК або мобільний пристрій з доступом до мережі Інтернет.

A.3.4 Вимоги до складу та параметрів технічних засобів

ПК або мобільний пристрій з мінімальними характеристиками для підтримки ОС Windows 7 та вище з високошвидкісним доступом до мережі Інтернет.

A.3.5 Вимоги до інформаційної та програмної сумісності

Для експериментів ВЛ RELDES використовується код мови C++, що використовується звичайними платами платформи Arduino.

Експерименти використовують Arduino-сумісні датчики, а також керують модулями для даної платформи, тому необхідно використовувати відповідні бібліотеки та апаратні засобами, що підтримуються.

Експеримент Розумна теплиця використовує для передачі даних протокол Modbus, тому для його інтеграції у системи домашньої автоматизації має бути підтримка цього протоколу.

A.4 Техніко-економічні показники

Проведена модернізація дозволяє покращити продуктивність роботи з ВЛ RELDES, а також збільшує функціональність, що в результаті робить дану лабораторію кращою для навчання.

A.5 Стадії та етапи розробки

Далі наведено етапи розробки апаратно-програмного комплексу модернізацій:

- аналіз предметної області;
- вибір апаратних компонентів;
- розробка архітектури програм;
- реалізація програмного забезпечення;
- реалізація апаратного забезпечення;
- тестування комплексу модернізацій.

A.6 Порядок контролю та приймання

Програмний застосунок повинно бути протестовано на пристрої, що задовольняє вимогам експлуатації. Програмний застосунок не повинен мати програмних помилок та збоїв у роботі.

Принципи контролю виконання роботи визначаються календарним планом, розташованим у завданні на дипломну кваліфікаційну роботу, де вказано назви етапів, строки їх виконання та результат.

ДОДАТОК Б
Тексти програм

Б.1 Текст файлу «SmartGreenHouse.ino»

```
//2 канальний модуль реле
#define pinRelay1 8
#define pinRelay2 9
//Ємнісний датчик вологості ґрунту
#define pinCapacitiveSoil A0
//Датчик освітленості
#define pinTemt6000 A1
//Температурний датчик водонепроникний
#define pinds18b20 7
//Сервопривід
#define pinServo 10
//порт Modbus
#define TXEN 2

#include <OneWire.h>
#include <Wire.h>
#include "SHT2x.h"
#include <Servo.h>
#include "ModbusRtu.h"

//HTU21 Датчик вологості та температури
SHT2x SHT2x;
//Servo
Servo myservo; // створимо об'єкт сервопривода
OneWire ds(7); // Створюємо об'єкт OneWire для шини 1-Wire, с
помогою якого буде здійснюватися робота з датчиком
Modbus slave(9,0,TXEN); // RS-485

byte valueRelay1flag = 1, valueRelay2flag = 1; //Прапорці реле
int valueServo = 90; // початкова позиція сервопривода
int valueCapacitiveSoil = 0; // початкове значення датчика
вологості ґрунту
float valueSHT2xHumidity = 0.0; // вологість
float valueSHT2xTemperature = 0.0; // температура
int valueTemt6000 = 0; // Датчик освітленості
float value_ds18b20 = 0.0; // температура
```

```

volatile int state = LOW;
uint32_t myTimer1;
// data array for modbus network sharing
uint16_t au16data[15] = {
    1,2,3,4,5,6,7,8,9,10,11,0,0,0,0 };

float temperature_ds18b20();
void setup() {
    //Serial.begin(9600);    // ініціалізація монітору порту
    slave.begin( 9600 ); // baud-rate at 19200
    SHT2x.begin();
    myservo.attach(pinServo); // сервопривід
    myservo.write(valueServo);

    pinMode(pinRelay1, OUTPUT);
    pinMode(pinRelay2, OUTPUT);
    pinMode(pinTemt6000, INPUT); // налаштування temt6000Pin на вхід

    digitalWrite(pinRelay1, valueRelay1flag);
    digitalWrite(pinRelay2, valueRelay2flag);
}

void loop() {
    digitalWrite(pinRelay1, valueRelay1flag);
    digitalWrite(pinRelay2, valueRelay2flag);
    myservo.write(valueServo);

    valueCapacitiveSoil = analogRead(pinCapacitiveSoil); // вологість
    valueSHT2xHumidity = SHT2x.GetHumidity(), 1; // вологість
    valueSHT2xTemperature = SHT2x.GetTemperature(), 1; //
температура
    valueTemt6000 = analogRead(pinTemt6000); //світло

    value_ds18b20 = temperature_ds18b20();

    int valueLeft = 0, valueRight = 0;

```

```

aui6data[0]=valueRelay1flag;
aui6data[1]=valueRelay2flag;
aui6data[2]=valueServo;
aui6data[3]=valueCapacitiveSoil;
valueLeft = valueSHT2xHumidity;
valueRight = int((valueSHT2xHumidity - valueLeft)*100);
aui6data[4]=valueLeft;
aui6data[5]=valueRight;
valueLeft = valueSHT2xTemperature;
valueRight = int((valueSHT2xTemperature - valueLeft)*100);
aui6data[6]=valueLeft;
aui6data[7]=valueRight;
aui6data[8]=valueTemt6000;
valueLeft = value_ds18b20;
valueRight = int((value_ds18b20 - valueLeft)*100);
aui6data[9]=valueLeft;
aui6data[10]=valueRight;

slave.poll( aui6data, 15 );

valueRelay1flag = aui6data[11];
valueRelay2flag = aui6data[12];
valueServo = aui6data[13];

/*
if (millis() - myTimer1 >= 1000) {
    myTimer1 = millis();
    Serial.println("*****");
    Serial.print("реле 1, реле 2 ");
    Serial.print(valueRelay1flag);
    Serial.print(" ");
    Serial.println(valueRelay2flag);
    Serial.print("Серво ");
    Serial.println(valueServo);
    Serial.print("Влажность грунта ");
    Serial.println(valueCapacitiveSoil);
    Serial.print("Влажность ");
    Serial.println(valueSHT2xHumidity);
    Serial.print("Температура ");
    Serial.println(valueSHT2xTemperature);

```

```

    Serial.print("Свет ");
    Serial.println(valueTemt6000);
    Serial.print("Температура ");
    Serial.println(value_ds18b20);
    Serial.print("*****");
    Serial.println("");
}
*/
}

```

```

float temperature_ds18b20() {

    byte data[2];

    ds.reset();
    ds.write(0xCC);
    ds.write(0x44);

    ds.reset();
    ds.write(0xCC);
    ds.write(0xBE);

    data[0] = ds.read();
    data[1] = ds.read();

    return ((data[1] << 8) | data[0]) * 0.0625;

}

```

Б.2 Текст файлу «LCD_Display1.ino»

```

#include "Wire.h"
#include "LiquidCrystal_I2C.h"

LiquidCrystal_I2C lcd(0x27,16,2);

void setup(){
    lcd.init();

```

```

        lcd.backlight();
    }

    void loop(){
        lcd.setCursor(0,0);
        lcd.print("Hello, world!");
        delay(1000);
        lcd.setCursor(0,1);
        lcd.print("I am display");
        delay(1000);
        lcd.clear();
        delay(1000);
    }

```

Б.3 Текст файлу «LCD_Display2.ino »

```

#include "Wire.h"
#include "LiquidCrystal_I2C.h"

LiquidCrystal_I2C lcd(0x27,16,2);
int col_offset = 0;
int row_offset = 0;

void setup(){
    lcd.init();
    lcd.backlight();
}

void loop(){
    lcd.setCursor(col_offset, row_offset);
    lcd.print("Running text");
    delay(100);
    lcd.clear();
    col_offset = (col_offset + 1) % 16;
    if(15 == col_offset) row_offset = (row_offset + 1) % 2;
}

```

Б.4 Текст файлу «LCD_Display3.ino »

```

#include <Wire.h>
#include <LiquidCrystal_I2C.h>

LiquidCrystal_I2C lcd(0x27,16,2);  // Устанавливаем дисплей
void setup()
{
    lcd.init();
    lcd.backlight(); // Включаем подсветку дисплея
    lcd.print("swed.zntu.edu.ua");
    lcd.setCursor(8, 1);
    lcd.print("LCD 1602");
}
void loop()
{
    // Устанавливаем курсор на вторую строку и нулевой символ.
    lcd.setCursor(0, 1);
    // Выводим на экран количество секунд с момента запуска ардуины
    lcd.print(millis()/1000);
}

```

Б.5 Текст файлу «VoltageSensor.ino»

```

const int voltage = A1;  //pin
void setup() {
    Serial.begin(9600);
}

void loop() {

    float valueV = analogRead(voltage);
    valueV *= 0.0244379;
    Serial.print("");
    Serial.print(valueV);
    Serial.print("V      ");
}

```

Б.6 Текст файлу «acs712.ino»

```

#define NUM_AVER5A 10    // вибірка (із скільки усереднювати)
#define NUM_AVER30A 10   // вибірка (із скільки усереднювати)

```

```

//порт Modbus
#define TXEN 2

#include "ModbusRtu.h"

int arr5A[3];
byte index5A;
int arr30A[3];
byte index30A;

const int acs712_pin5A = A3; //pin
const int acs712_pin30A = A0; //pin

float scale5A = (185.0*1024.0)/(5000.0); // скільки одиниць АЦП
приходиться на 1 ампер(+5A)
float scale30A = (66.0*1024.0)/(5000.0); // скільки одиниць АЦП
приходиться на 1 ампер(+30A)

int zero = 1024 / 2; // рівень нуля, відносно якого вимірюється
струм, зазвичай VCC/2
const float coefV = 0.00488; // maxV(5) / maxPMW(1024)
Modbus slave(8,0,TXEN); // RS-485
// data array for modbus network sharing
uint16_t au16data[15] = {
    1,2,3,4,5,6,7,8,9,10,11,0,0,0,0 };

void setup() {
    //Serial.begin(9600); // ініціалізація монітору порту
    slave.begin( 9600 ); // baud-rate at 19200
}

// отримує сглажені значення з АЦП Arduino
int getSmoothedValue5A() {

    static int t = 0;
    static int vals[NUM_AVER5A];
    static int average = 0;

    int val = analogRead(acs712_pin5A);

```

```

        if (++t >= NUM_AVER5A) t = 0; // перемотка t
        average -= vals[t];          // віднімаємо старе
        average += val;              // додаємо нове
        vals[t] = val;              // запам'ятовуємо в масив

        if (++index5A > 2) index5A = 0; // переключаємо індекс з 0 до 2
        (0, 1, 2, 0, 1, 2...)
        arr5A[index5A] = average / NUM_AVER5A; // записуємо значення з
        датчика в масив

        int val_filter = middle_of_3(arr5A[0], arr5A[1], arr5A[2]);

        return (val_filter);
    }

    int getSmoothedValue30A() {

        static int t = 0;
        static int vals[NUM_AVER30A];
        static int average = 0;

        int val = analogRead(acs712_pin30A);
        val += 409;
        if (++t >= NUM_AVER30A) t = 0; // перемотка t
        average -= vals[t];          // віднімаємо старе
        average += val;              // додаємо нове
        vals[t] = val;              // запам'ятовуємо в масив

        if (++index5A > 2) index5A = 0; // переключаємо індекс з 0 до 2
        (0, 1, 2, 0, 1, 2...)
        arr30A[index30A] = average / NUM_AVER30A; // записуємо значення з
        датчика в масив

        return (middle_of_3(arr30A[0], arr30A[1], arr30A[2]));
    }

    void loop() {

        int sensorValue;
        int curForce;

```

```

int valueLeft = 0, valueRight = 0;

    sensorValue = getSmoothedValue5A(); // читаємо значення з АЦП та
передаємо
    sensorValue =sensorValue * coefV;
    curForce = getCurrent(sensorValue,scale5A); // читаємо значення
з АЦП та передаємо
    if(curForce<60)
        curForce =0;
    else
        curForce =curForce/1000.0;

    valueLeft = sensorValue;
    valueRight = int((sensorValue - valueLeft)*100);
    aul6data[0]=valueLeft;
    aul6data[1]=valueRight;

    valueLeft = curForce;
    valueRight = int((curForce - valueLeft)*100);
    aul6data[2]=valueLeft;
    aul6data[3]=valueRight;

    sensorValue = getSmoothedValue30A(); // читаємо значення з АЦП
та передаємо
    sensorValue =sensorValue * coefV;
    curForce = getCurrent(sensorValue,scale30A); // читаємо значення
з АЦП та передаємо
    if(curForce<60)
        curForce =0;
    else
        curForce =curForce/1000.0;

    valueLeft = sensorValue;
    valueRight = int((sensorValue - valueLeft)*100);
    aul6data[4]=valueLeft;
    aul6data[5]=valueRight;

```

```

valueLeft = curForce;
valueRight = int((curForce - valueLeft)*100);
aui6data[6]=valueLeft;
aui6data[7]=valueRight;

}

// розраховуємо струм в мА заз наченням з АЦП
int getCurrent(int adc, float scale) {
    int delta = zero - adc; // відилення віж нуля шкали
    int current = (int)delta * 1000 / scale; // хчитуємо струм в мА
    return abs(current);
}

// медианний фільтр з 3х значень
float middle_of_3(float a, float b, float c) {
    int middle;
    if ((a <= b) && (a <= c)) {
        middle = (b <= c) ? b : c;
    }
    else {
        if ((b <= a) && (b <= c)) {
            middle = (a <= c) ? a : c;
        }
        else {
            middle = (a <= b) ? a : b;
        }
    }
    return middle;
}

```

ДОДАТОК В
Слайди презентації

Національний університет «Запорізька політехніка»

Кафедра програмних засобів

РОЗРОБКА ПРОГРАМНО-АПАРАТНОГО ЗАБЕЗПЕЧЕННЯ ДЛЯ МОДЕРНІЗАЦІЇ ЕКСПЕРИМЕНТІВ ВІДДАЛЕНОЇ ЛАБОРАТОРІЇ RELDES

Виконав
студент групи КНТ-217

Є. С. Троц

Керівник
к.т.н. доцент

А. В. Пархоменко

Рисунок В.1 – Слайд 1

Актуальність дипломної роботи

2

Виявлені апаратні та програмні проблеми при функціонуванні віддаленої лабораторії RELDES

Експеримент	Апаратні проблеми	Програмні проблеми	Наслідки	Експеримент	Апаратні проблеми	Програмні проблеми	Наслідки
1. Сенсна панель	Не підключено датчик струму та напруги	Не налаштовано програму для коректного зчитування показань	Неможливо проаналізувати показання напруги та сили струму	6. Контроль безпеки	Невідповідний датчик	Відсутній	Вислідок невідповідного датчику не завжди спрацьовує тривога
2. Контроль доступу	Відсутній	Відсутня реалізація датчика RFID-модулю. Відбувається лише імітація	Візуально недоліків немає. Зчитування ключів відсутнє, тому відбувається лише його імітація	7. LCD дисплей	Недостатньо живлення. Не оптимальний кут зору до камери	На сервері відсутня необхідна бібліотека. Неявно працює код присадів	При спробі завантаження коду виникає помилка. При роботі яскравість замала. З камери зображення недостатньо добре видно
3. Клімат контроль	Відсутній	Неявно працює модуль стану повітря	Неможливо отримати правильні дані стану повітря	8. TFT дисплей	Недостатньо живлення.	На сервері відсутня необхідна бібліотека	При спробі завантаження коду виникає помилка. При роботі яскравість замала.
4. Контроль присутності	Невідповідний датчик	Відсутній	Вислідок невідповідного датчику не завжди спрацьовує тривога	9. Серводвигун	Недостатньо живлення.	Відсутній	Робота модуля призводить до просадки живлення, що порушує роботу інших модулів.
5. Wi-Fi безпека	Відсутній	Відсутня реалізація wi-fi атака. Відбувається лише імітація	Візуально недоліків немає. Деаутифікація клієнтів програмно відсутня, тому відбувається лише її імітація	10. Датчик	Недостатньо живлення.	Показання датчика не виводяться.	Неможливо ознайомитись з результатами. Робота модуля призводить до просадки живлення, що порушує роботу інших модулів.

Рисунок В.2 – Слайд 2

Мета та задачі дипломної роботи

3

Мета роботи – модернізація програмно-апаратного забезпечення віддаленої лабораторії RELDES для розширення функціональних можливостей та підвищення ефективності її використання в навчальному процесі.

Задачі роботи:

- аналіз вимог щодо модернізації програмно-апаратного забезпечення віддаленої лабораторії;
- проектування архітектури програмно-апаратного забезпечення;
- вибір апаратних та програмних засобів розробки;
- конструювання програмно-апаратного забезпечення;
- розробка програмної документації.

Рисунок В.3 – Слайд 3

Архітектура віддаленої лабораторії

4

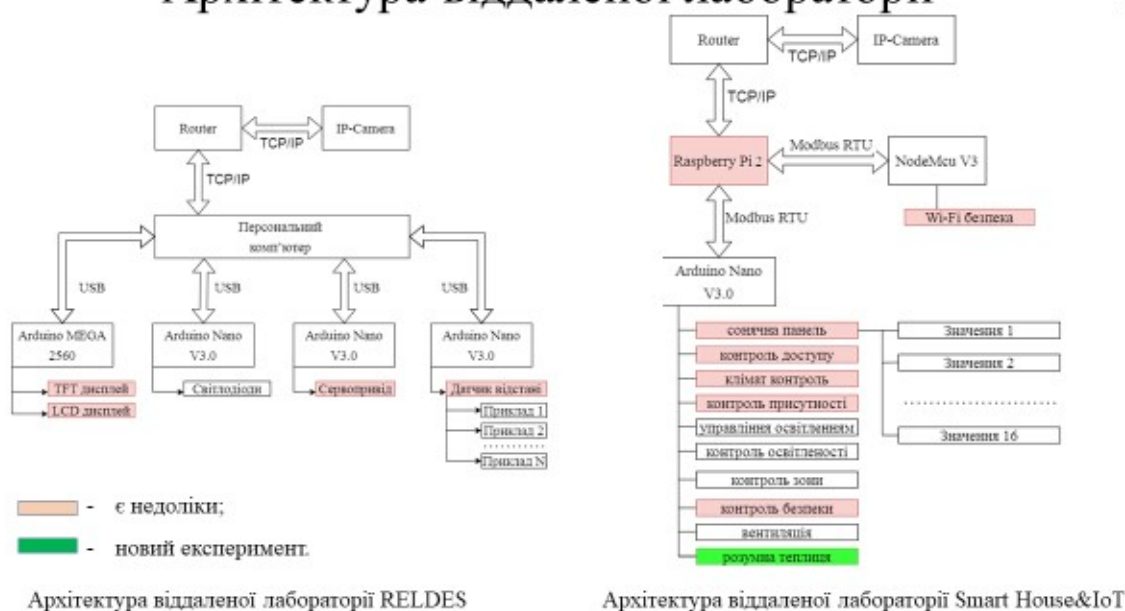


Рисунок В.4 – Слайд 4

Порівняння та вибір мінікомп'ютера Raspberry Pi ⁵

Характеристика	Raspberry Pi 2 B	Raspberry Pi 3 B
CPU	Broadcom BCM2836	Broadcom BCM2837
GPU	Video Core IV 250 MHz	Video Core IV 400 MHz
Кількість ядер	4x900 MHz	4x1.2 GHz
Wi-Fi	-	802.11n 4.1
Відео вихід	HDMI, TRRS	HDMI, TRRS, DSI

Рисунок В.5 – Слайд 5

Порівняння та вибір ОС для Raspberry Pi

6

ОС	Опис	Особливості
Raspbian	Стандартна ОС на базі Debian, що була спеціально створена для Raspberry Pi	Містить стандартний набір програм і утиліт. Також містить розділ «програмування», що надає можливість використання різноманітних додатків для програмування
OSMC	Медіацентр на базі Kodi	Має гарний функціонал ПЗ для роботи з мультимедіа
OpenELEC	ОС для програвання мультимедіа	Інтерфейс схожий на Kodi, проте має вищу швидкість роботи
DietPi	ОС, що заснована на Debian	Займає лише 400 Мб на карті пам'яті. Має низькі вимоги до ресурсів та підтримує швидку установку програм
Windows 10 IoT Core	Версія Windows, що створена для комп'ютера Raspberry Pi	Має сумісність лише з ПК на базі Windows 10
ОС RISC	ОС, що не базується на системі Linux, внаслідок чого не схожа на інші системи	Працює як однокористувальська система. Загалом, система не використовується як основна ОС
RetroPie	Емулятор	Надає можливість грати на багатьох консолях. ОС заснована на Raspbian
Pidora	ОС, що заснована на проєкті Fedora	У порівнянні з Debian для формату пакетів має розширення RPM

Рисунок В.6 – Слайд 6

Вибір апаратних компонентів для експерименту Розумна теплиця

Компонент	Функція
Модуль реле 5В 10А з опторозв'язкою на 2 канали	Увімкнення та вимкнення освітлення та подачі води для поливу
Сервопривід SG90	Керування положенням хвіртки
Температурний датчик водонепроникний DS18B20	Зчитування показань температури ґрунту
Датчик вологості та температури GY-21 HTU21	Зчитування показань температури та вологості в теплиці
Датчик освітленості TЕМТ6000	Зчитування показань освітленості
Ємнісний датчик вологості ґрунту	Зчитування показань вологості ґрунту
Конвертер USB to RS-485	Конвертація сигналу інтерфейсу USB в інтерфейс RS-485
Конвертер RS-485 to TTL	Конвертація сигналу інтерфейсу RS-485 в інтерфейс TTL
Arduino Nano v3	Керування пристроями

Рисунок В.7 – Слайд 7

Модернізація експерименту з дисплеєм

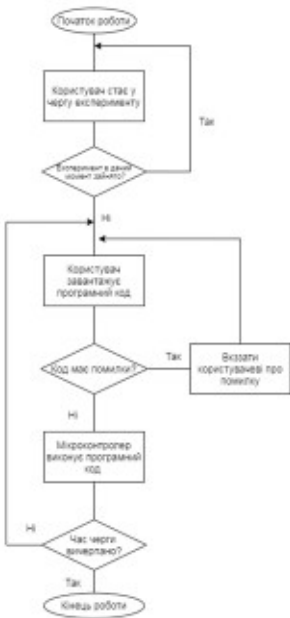
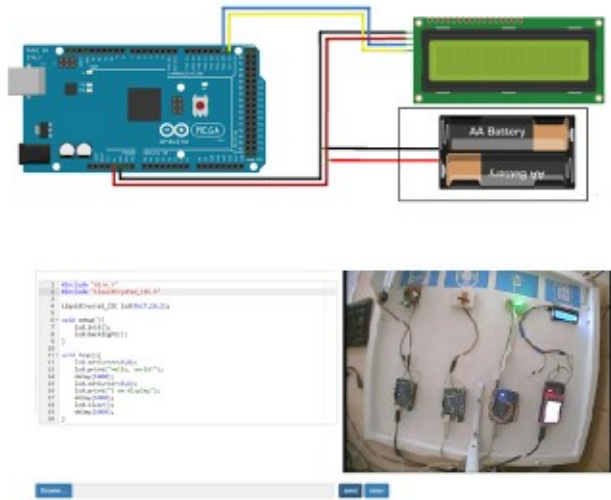
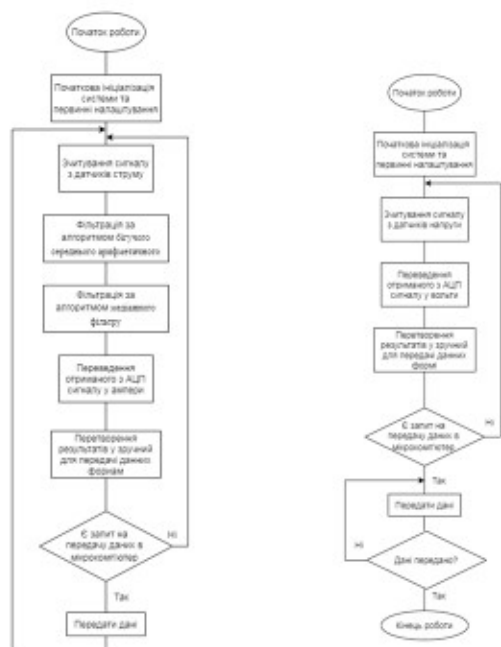
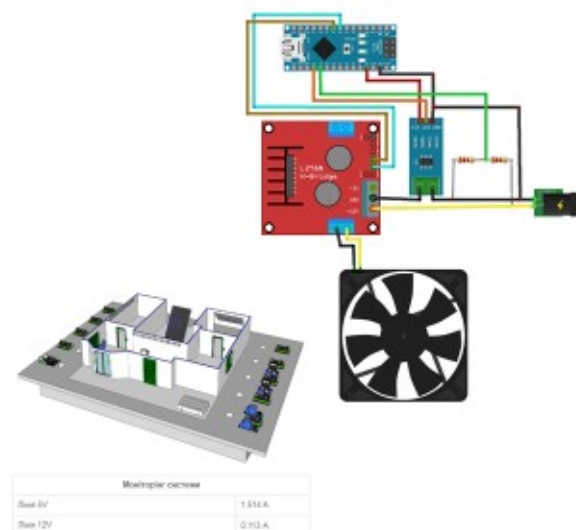


Рисунок В.8 – Слайд 8

Модернізація моніторингу напруги та струму



Алгоритми вимірювання сили струму та напруги

Рисунок В.9 – Слайд 9

Реалізація експерименту Розумна теплиця

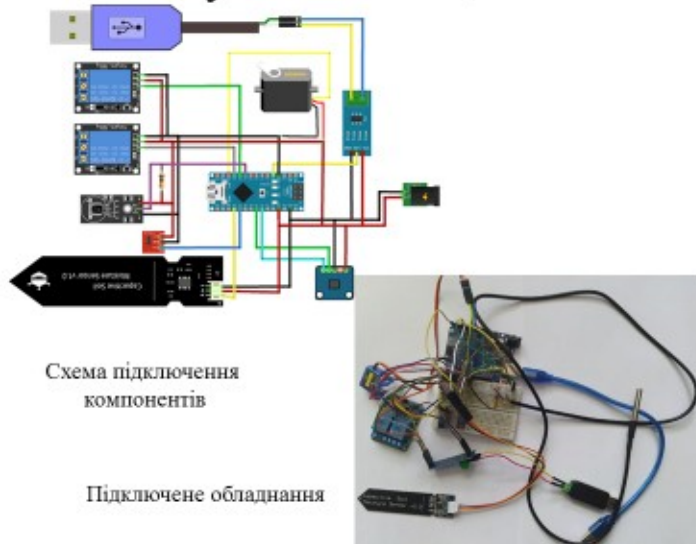


Схема підключення
компонентів

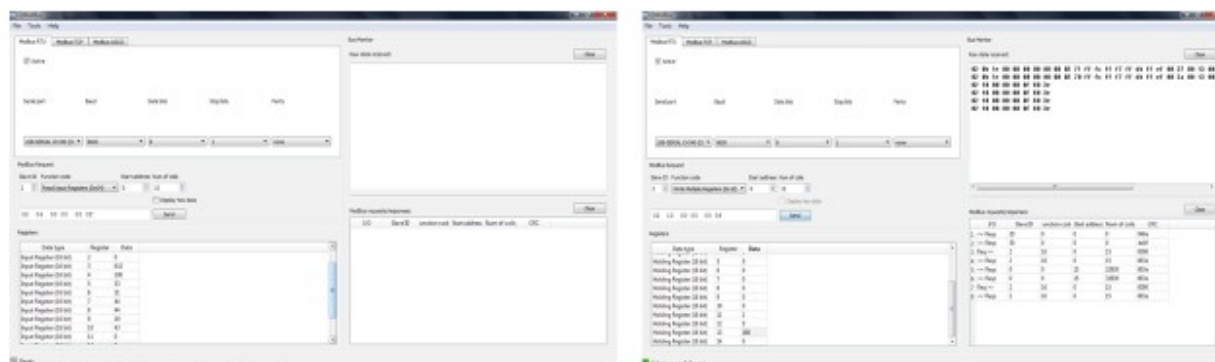
Підключене обладнання



Рисунок В.10 – Слайд 10

Екранні форми експерименту Розумна теплиця

11

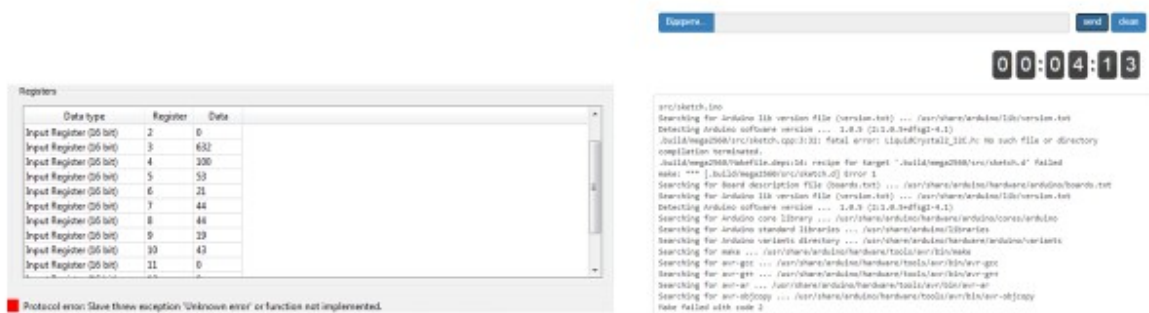


Зчитування та запис даних за протоколом Modbus

Рисунок В.11 – Слайд 11

Повідомлення про помилки

12



Приклад повідомлення про помилки при спілкуванні з мікроконтролером за допомогою протоколу Modbus

Приклад повідомлення про помилку компіляції та завантаження коду в лабораторії RELDES

Рисунок В.12 – Слайд 12

Висновки

13

- В ході виконання дипломної роботи було проведено модернізацію апаратного та програмного забезпечення віддаленої лабораторії RELED, в результаті якої було усунуто ряд недоліків системи, зокрема неможливість проведення експерименту з дисплеєм та вимірювання значень сили струму та напруги.
- Розроблений експеримент Розумна теплиця дозволяє проводити дослідження з макетом теплиці та надає можливість вимірювати значення температури та вологості ґрунту та повітря, а також показники освітлення. Також система дозволяє керувати зовнішнім освітленням та поливом рослин, а для регулювання температури та вологості надається можливість регулювання вентиляції. Експеримент готовий до інтеграції з системою домашньої автоматизації за протоколом Modbus.
- В майбутній роботі планується подальша модернізація системи, в результаті якої має бути усунена імітація роботи датчика електронного ключа для відкривання дверей, додано новий Wi-Fi модуль для експерименту з деаутентифікацією для демонстрації загрози. Також планується замінити інфрачервоні датчики ультразвуковими або лазерними модулями, а також додати до форми на сайті можливість ознайомлення з результатами експерименту з датчиком відстані.

Рисунок В.13 – Слайд 13

14

Дякую за увагу !

Рисунок В.14 – Слайд 14