МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»

Факультет інформаційної безпеки та електронних комунікацій
(повне найменування інституту, факультету)

Кафедра інформаційні технології електронних засобів
(повне найменування кафедри)

Пояснювальна записка

до дипломного проекту (роботи)
магістра
(ступінь вищої освіти (освітній ступінь))

на тему **РОЗРОБКА СИТЕМИ УПРАВЛІННЯ ДЛЯ ПІДЛОГОМИЙНИХ МАШИН І ДОК-СТАНЦІЙ**

Виконав: студент 2 курсу, групи БК-512м
спеціальності
172 «Телекомунікації та радіотехніка»
(код і найменування спеціальності)
Освітня програма (спеціалізація)
«Радіоелектронні апаратні засоби »

Іван ЗОРІН
(прізвище та ініціали)
Керівник Наталія ФУРМАНОВА
(прізвище та ініціали)
Рецензент Гаррі МОРОЗ
(прізвище та ініціали)

м. Запоріжжя
2023 рік

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

Інститут, факультет інформаційної безпеки та електронних комунікацій
Кафедра Інформаційні технології електронних засобів
Ступінь вищої освіти (освітній ступінь) магістр
Спеціальність 172 «Телекомунікації та радіотехніка»
(код і найменування)

Освітня програма (спеціалізація) «Інтелектуальні технології мікросистемної радіоелектронної техніки»
(назва освітньої програми (спеціалізації)

ЗАТВЕРДЖУЮ
Завідувач кафедри ІТЕЗ Олександр МАЛИЙ
канд.тех.наук, доцент
"____" _____ 2023 року

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

Івана ЗОРІНА
(прізвище, ім'я, по батькові)

1. Тема проекту (роботи) Розробка системи управління для підлогомийних машин і док-станцій
керівник проекту (роботи) Наталія ФУРМАНОВА, к. т. н., доцент
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)
затверджені наказом вищого навчального закладу від "21" листопада 2023 року №367
2. Строк подання студентом проекту (роботи) 21 грудня 2023 року
3. Вихідні дані до проекту (роботи) мобільні роботизовані системи роботи прибиральники. Необхідно розробити систему, здатну автономно здійснювати прибирання у палатах за графіком який заздалегідь налаштований оператором системи роботів.

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)
1) Аналіз предметної області;
2) Проєктування системи;
3) Апаратне забезпечення системи;
4) Налаштування роботи системи.
5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень)
Загальна структура схеми роботизованої системи; Загальний алгоритм функціювання підлогомийного роботу; Загальний алгоритм функціювання док-станції; Структурна схема роботу; Структурна схема док-станції; Електрична схема роботу; Електрична схема докстанції;

6. Консультанти розділів проекту (роботи)

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	прийняв виконане завдання
1-5	Наталія ФУРМАНОВА,к.т.н., доцент		
Нормоконтроль	Ірина ПОСПЄЄВА, старший викладач		

7. Дата видачі завдання 04.09.2023 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломного проекту (роботи)	Строк виконання етапів проекту (роботи)	Примітка
1	Аналіз подібних робототехнічних систем та стратегій	05.10.2022 р.	
	щодо їх розробки		
2	Постановка завдання	05.10.2023 р.	
3	Розробка вимог до системи	07.10.2023 р.	
4	Розробка загальної структурної схеми системи	12.10.2023 р.	
5	Розробка структурної схеми складових системи	17.11.2023 р.	
6	Розробка алгоритму функціювання роботів	19.11.2023 р.	
7	Розробка алгоритму функціювання док- станції	22.11.2023 р.	
8	Вибір апаратного забезпечення системи	26.11.2023 р.	
9	Налаштування роботи системи	29.11.2023 р.	
10	Аналіз результатів роботи системи	30.11.2023 р.	
11	Оформлення пояснювальної записки	01.12.2023 р.	
12	Проходження нормоконтролю	07.12.2023 р.	

Студент _____ **Іван ЗОРІН**
(підпис) (ініціали та прізвище)

Керівник проекту (роботи) _____ **Наталія ФУРМАНОВА**
(підпис) (ініціали та прізвище)

РЕФЕРАТ

ПЗ: 91 с. 52 рис., 25 джерел.

АВТОМАТИЗАЦІЯ, НА ОСНОВІ ШТУЧНОГО ІНТЕЛЕКТУ, ПІДЛОГОМИЙНИЙ РОБОТ, РОБОТ, КОМП'ЮТЕРНИЙ ЗІР, RASPBERRYPI 4, OPENCV, РОБОТИЗОВАНА СИСТЕМА.

Об'єкт розробки – автономна безпілотна роботизована система прибирання, що складається з підлогомийних роботів.

Мета роботи – створення ефективного та дієвого рішення для підтримання чистоти та гігієни в лікарнях шляхом автоматизації процесу прибирання за допомогою запланованих операцій, що виконуються підлогомийними роботами.

У першій частини дипломної робити магістра досліджується сфера готових робототехнічних систем та роботів-прибиральників, заглиблюючись у різні методології, що застосовуються при розробці подібних робототехнічних систем.

У другій частини дипломної робити магістра було визначено передумови створення системи, розроблена комплексна структурна схема роботизованої системи та її складових елементів, а також сформульовані загальні алгоритми, що керують її функціонуванням.

У третій частини розроблено апаратну складову роботизованої системи, підлогомийного роботу та док-станції з доступних на ринку модулів та пристроїв.

У четвертій частині описано налаштування системи та встановлення основного програмного забезпечення, необхідного для автономної роботи підлогомийного роботу.

У п’ятій частині описано встановлення інструментів й бібліотек, завдяки яким відбувалася розробка програмного забезпечення для курування розробленою системою керування. Налаштоване програмне забезпечення дозволяє системі функціонувати в автоматичному режимі та виконувати попередньо визначені оператором підлогомийного робота зони прибирання.

ЗМІСТ

ВСТУП

Стрімкий розвиток алгоритмів штучного інтелекту в останні десятиліття та їхня інтеграція в різні аспекти людського життя призвели до глибоких трансформацій у взаємодії "людина-людина", "людина-комп'ютер" та "комп'ютер-комп'ютер". Оскільки світ стає все більш взаємопов'язаним і залежним від технологій, зростає потреба в автономних системах, здатних виконувати завдання, які традиційно виконувала людина, особливо в таких життєво важливих місцях, як лікарні. На цьому тлі розробка автономної, безпілотної роботизованої системи прибирання для лікарень стає актуальною і важливою сферою для досліджень та інновацій.

Лікарні - це об'єкти з високою прохідністю, які вимагають суворих стандартів чистоти для підтримки безпечного і здорового середовища для пацієнтів, персоналу та відвідувачів. Традиційні методи прибирання часто не відповідають вимогам ефективності, результативності та послідовності. Прибиральники стикаються з такими проблемами, як обмежена доступність, фізична втома та ризик потенційного впливу патогенних мікроорганізмів. Отже, існує потреба в автоматизованих рішеннях, які можуть покращити процес прибирання, одночасно зменшуючи робоче навантаження та ризики для здоров'я, пов'язані з ручною працею. Таким чином, обрана тема дипломної роботи є актуальною.

Метою роботи є розробка підлогомийних роботів, які оснащені сучасними сенсорами, комп'ютерним зором і алгоритмами машинного навчання, що працюють за заздалегідь встановленим графіком, визначеним оператором роботизованої ситеми. Використовуючи технології штучного інтелекту, ці роботи можуть орієнтуватися в лікарняному середовищі, визначати та оцінювати зони, які потребують

прибирання, а також ефективно та якісно виконувати завдання з прибирання.

Для досягнення поставленої мети необхідно виконати наступні завдання:

- провести аналіз архітектурних та функціональних рішень різних роботизованих систем;
- спроектувати роботизовану систему та розробити алгоритм її функціонування;
- обрати необхідне апаратне забезпечення системи;
- зібрати та налаштувати систему роботів;
- провести аналіз отриманих результатів.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

У цьому розділі проводиться розгляд загальних ідей робототехнічних систем. Тут представлено огляд різних типів роботизованих систем, а також розглянуто сучасний аналіз і підходи до розробки систем і концепцій існуючих реалізацій роботів прибиральників. Цей аналіз допомагає визначити найбільш підходящий підхід до розробки роботизованої системи для вирішення завдань, окреслених у проекті. Завдяки ретельному вивченню подібних систем стає можливим встановити конкретні вимоги та отримати чітке розуміння підходу для початкових етапів розробки платформи робота, апаратного забезпечення та інформаційної системи. В кінцевому підсумку, такий підхід скорочує час і витрати на розробку.

1.1 Загальні відомості про робототехнічні системи

Загальне розуміння роботизованої системи передбачає інтеграцію різних технічних дисциплін, включаючи прикладну математику, лінійну алгебру, машинобудування, електротехніку та розробку програмного забезпечення. Ці дисципліни в сукупності дозволяють створити програмовану механічну систему, керовану комп'ютером. Така система здатна сприймати та реагувати на сигнали навколишнього середовища, виконуючи поставлені завдання з певним рівнем автономності.

Вивчаючи доступні джерела інформації про різноманітний спектр роботів, що з'явилися останніми роками [1], можна визначити їх за кількома конструктивно-функціональними характеристиками. Ґрунтуючись на цих характеристиках, виокремлюють дві ключові категорії роботів: роботи-маніпулятори та автономні мобільні роботи.

Роботи-маніпулятори являють собою, як правило, стаціонарні або напівстаціонарні системи, основною функцією яких є виконання маніпуляцій з об'єктами різної складності. Ці роботи часто використовуються в

промисловості для автоматизації різних процесів, як-от складання, зварювання, фарбування та інші. Їхня конструкція та функціональність можуть варіюватися від простих одновісних механізмів до складних багатозадачних систем із високим ступенем автономії та інтегрованими сенсорами.

Автономні мобільні роботи розроблені для автономного пересування в різних середовищах – у повітрі, на суші або на морі. Вони можуть бути оснащені різними сенсорами та інструментами для виконання своїх завдань. Повітряні дрони, наземні роботи на колесах або гусеницях і підводні апарати – це лише деякі приклади з цієї категорії.

Такі роботи, залежно від їхніх конструктивних особливостей і функціонального призначення, можуть або імітувати поведінку людини або різних тварин (наприклад, роботи, що імітують рухи комах або птахів), або являти собою абсолютно нові конструктивні рішення, оптимізовані для виконання специфічних завдань.

Проте, незважаючи на різноманітність функціональних і конструктивних характеристик роботів, у центрі багатьох сучасних робототехнічних систем лежить мехатроніка. Більшість із них оснащені приводами, які поєднують прецизійні механічні компоненти з електричними, утворюючи так звану мехатронну систему [2]. Спектр завдань, які виконують мехатронні системи в робототехніці, дуже широкий, але існують певні характеристики, спільні для більшості цих систем. На рисунку 1.1 наведено схематичне зображення потоку сигналів у типовій мехатронній системі [2]. Команди від цифрових комп'ютерних систем перетворюються в аналогові сигнали за допомогою цифро-аналогового перетворювача, після чого активуються виконавчі механізми. Ці механізми забезпечують взаємодію між електричними та механічними компонентами, які зі свого боку взаємодіють із навколишнім середовищем.

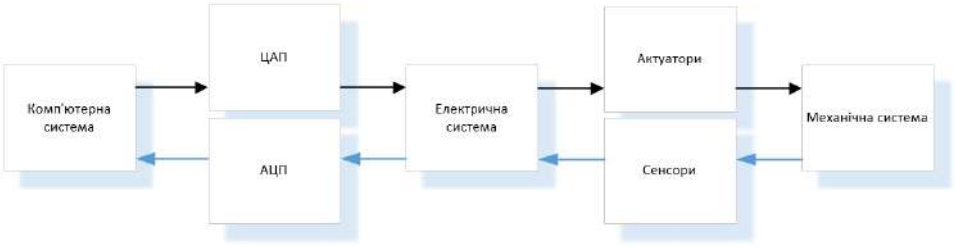


Рисунок 1.1 – Схема потоку сигналу для типової мехатронної системи

Заглиблюючись у вивчення робототехнічних систем, можна звернути увагу на різноманітність їхнього застосування. Виходячи з аналізу доступної літератури [3], робототехнічні системи можна групувати за такими основними напрямками застосування:

- промислові роботи;
- сервісні роботи;
- дослідницькі та розвідувальні роботи.

Промислові роботи – це автоматизовані механізми розроблені спеціально для виробничого середовища, де вони виконують безліч операцій, як-от маніпуляції з об'єктами, зварювання та монтаж. Особливістю промислових роботів є їхня висока точність, швидкість і можливість функціонування без простоїв. Завдяки можливості програмування вони можуть бути легко переналаштовані для виконання різноманітних завдань і часто інтегруються з іншими промисловими системами [4].

Сервісні роботи – це роботи, розроблені для допомоги людям у повсякденному житті, включно з галузями медицини, освіти та домашніх послуг. Ці роботи часто відрізняються багатофункціональністю, здатністю взаємодіяти з людиною та адаптуватися до різних умов. Вони здатні працювати автономно, навчатися нових завдань і адаптуватися до мінливих обставин. Прикладом такої системи є домашній робот-прибиральник.

Дослідницькі та розвідувальні роботи – це такі роботи, які призначені для дослідження місць, які можуть бути недоступні або небезпечні для людини. Ці роботи мають високу автономність, здатність адаптуватися до екстремальних умов і оснащені безліччю сенсорів та інструментів для збору даних. Завдяки дальній взаємодії вони можуть передавати дані на великі відстані, забезпечуючи людству цінну інформацію з найвіддаленіших і незвіданих місць. Приклад – космічна обсерваторія GAIA.

1.2 Аналіз робототехнічних систем та підходів щодо їхньої розробки

Існує кілька методів, які можна використати для прискорення процесу розробки пристрою або продукту. Один із них - аналіз наявних готових систем, які вже мають готове рішення і виконують схожі завдання. Навіть якщо вони віддалено схожі, їхні рішення можна адаптувати або використати за основу. Іншим підходом до прискорення розробки є використання наявних апаратних платформ і програмного забезпечення. Цей метод призводить до створення модульної платформи для кінцевого продукту, що надалі слугуватиме полегшенням його експлуатації та модифікації.

Розглянемо готові автономні роботизовані автономні системи, які розроблялися для прибирання в приміщеннях. Нижче представлений аналіз і візуальне представлення цих роботів взяті з офіційних сайтів компаній [5-7].

Основне призначення таких роботизованих систем - автоматизувати процес прибирання підлоги. Ці машини призначені для автономної роботи й переміщення по визначеній зоні та виконання завдань з миття й сушіння без втручання людини. Яскравий прикладом може послугувати автономний промисловий робот-прибиральник RA660 Navi від компанії Cleanfix. RA660 Navi, який можна побачити на рисунку 1.2. Робот оснащений сенсором позиціювання й алгоритмом внутрішньої навігації, що забезпечує його автономне переміщення за маршрутом, який задають алгоритми очищення роботу.

Рисунок 1.2 – Польовий промисловий робот-прибиральник RA660 Navi

Конструкція, яка відповідає за чищення роботом підлоги, представлена на рисунку 1.3 (зображення взяте із сайту компанії), оснащена обертовими щітками або подушечками, які безпосередньо контактують з поверхнею підлоги. Ці щітки або губки перемішують бруд і плями, розпушуючи їх для ефективного очищення. Тип щіток або подушечок залежить від матеріалу підлоги та інтенсивності очищення. На задній частині конструкції очищення робота можна побачити систему скребка. Система скребків відповідає за збір брудної води та миючого розчину з підлоги після миття. Вона складається з гумового леза, яке притискається до поверхні підлоги, щоб видалити рідину. Зазвичай система скребків регулюється, щоб пристосуватися до різних типів підлоги і забезпечити ефективне збирання води. Система скребків працює разом із вакуумною системою, яка всмоктує брудну воду і сміття, зібране скребком. Ця система гарантує, що після прибирання підлога залишається чистою і сухою.

Рисунок 1.3 – Система чищення робота-прибиральника RA660 Navi

Маршрути, які задають алгоритми очищення роботу, задаються користувачем завдяки простому у використанні додатку для створення звітів для Android і iOS. Додаток RA660 Navi Reporting створює записи про всі інциденти, що сталися в процесі очищення, і зберігає їх. Таким чином, керівництво об'єкта завжди має огляд всіх використовуваних роботів-прибиральників і їх поточний стан. За допомогою push-повідомлень додаток інформує користувачів на їхньому смартфоні або планшеті в режимі реального часу про нові інциденти, що надходять. Для архівування або інших цілей звіти також можна завантажити.

Також до роботу постачаються док-станція (рисунок 1.4), яка знімає такі обмеження, як залежність від кількості води у баку робота і часу автономної роботи батареї, док-станція реалізована на оснівні зарядного пристрою Kuka carla_connect, яка дає змогу заряджати акумулятор і доливати прісну воду повністю автономно.

Рисунок 1.4 – Обслуговування док станцією робота-прибиральника RA660 Navi

Це знімає обмеження застосування RA660 Navi на дуже великі площі. Коли батарея або запас прісної води закінчуються, робот-прибиральник автоматично повертається на док-станцію і перезаряджається. Після успішного заправлення робот повертається в попереднє положення і продовжує процес прибирання. Kuka carla_connect – це зарядний робот для електромобілів від німецької компанії KUKA AG, який можна побачити на рисунку 1.4 або 1.5. Представляє із себе маніпуляційну роботизовану систему.

Рисунок 1.5 – Зарядний робот Kuka carla_connect

Маніпулятор робота зі своєю вилкою автономно переміщується до з'єднувальної муфти за допомогою оптичного виявлення камерою гнізда і заряджає акумулятори. Ключовим компонентом цієї маніпуляційної роботизованої системи є спеціальний захват, який дозволяє не залежати від конкретного виду роз'ємів зарядки електромобілів.

На рисунку 1.4 видно, що компанія Cleanfix використала готовий продукт кампанії KUKA AG для свого робота RA660 Navi, додавши його у систему зарядки відповідний роз’єм, недалеко від якого по трубці подається вода.

Ще одним прикладом, використаним у дослідженні, є робот-прибиральник KIRA B 50 від компанії Kärcher. Kärcher – німецька компанія, що спеціалізується на виробництві обладнання та засобів для прибирання. Компанія широко відома своїми мийками високого тиску, пилососами, пароочисниками, мийками вікон, мийками для підлоги та іншими товарами для прибирання.

Робот KIRA B 50, який можна побачити на рисунку 1.6, також як і RA660 Navi, оснащений інтелектуальною навігаційною системою для

автономної роботи і може самостійно маневрувати на різних поверхнях й ефективно визначаючи зону прибирання та уникаючи перешкод.

Рисунок 1.6 – Робот-прибиральник KIRA B 50

Система чищення робота прибиральника KIRA B 50 дещо відрізняться від робота RA660 Navi. На рисунку 1.7 можна роздивитися, що попередньо робот підмітає завдяки круговій бічній щітці, за документацією її робоча ширина становить 55 см. Далі сміття попадає на роликові щітки, які парно розташовані перпендикулярно вектору руху робота. валики просякнуті водою з хімічним розчином, таким чином збирається сміття і чиститься поверхня, далі працює система скребків разом із вакуумною системою.

Рисунок 1.7 – Робот-прибиральник KIRA B 50 вигляд з низу

Разом із роботом також постачається і спеціально розроблена для нього док станція яку можна побачити на рисунку 1.8. Док-станція забезпечує робота наповненням бака для прісної води, спорожнення і промивання бака для відпрацьованої води й зарядки потужного літій-іонного акумулятора. Щоразу, коли KIRA B 50 потребує чогось із перерахованого вище, він просто підключається до док-станції, де всі його потреби виконуються в автоматичному режимі.

Рисунок 1.8 – Робот-прибиральник KIRA B 50 і док станція

На додаток до можливостей автономного прибирання, робот-прибиральник KIRA B 50 від Kärcher також пропонує ручний режим, що може забезпечити користувачам гнучкість у виконанні рутинних операцій з прибирання. На роботі розташований сенсорний дисплей завдяки якому можна керувати рухами робота та діями з прибирання (рисунок 1.9).

Рисунок 1.9 – Режим ручного роботу KIRA B 50

Було проаналізовано та вивчено ще декілька подібних за реалізацією роботизованих систем від різних компаній, які мають дуже багато спільного з приводу реалізації навігації та розташування сенсорів, але відрізняються дизайном, ходовою й об'ємом стартових ресурсів, що потрібні для прибирання, об'єм баку води, ємність акумулятору тощо.

Особливу увагу привернув компактний і простий робот UV-Guardian від Richtech Robotics, яка презентувала робота у 2019 року і випустила його 2020 року. Робот UV Guardian, представлений на рисунку 1.10, використовує інший підхід. На відміну від розглянутих вище роботів, які використовували механічну і хімічну обробку поверхні, робот UV Guardian використовує технологію ультрафіолетового випромінювання для ефективної дезінфекції повітря і поверхонь.

Рисунок 1.10 – Робот UV Guardian

Оснащений потужними ультрафіолетовими лампами, цей робот може ефективно знищувати шкідливі організми бактерії або віруси, змінюючи

структуру і молекулярні зв'язки їхніх ДНК, ефективно дезінфікуючи поверхні.

Робот здатен вільно переміщуватись у приміщеннях з великою кількістю об’єктів завдяки інтелектуальній технології розпізнавання. Технологія об'єднує данні з камер спереду та ззаду разом із 230-градусним лідаром, що дає змогу роботу точно сприймати оточення й будувати 2D карту навколишнього середовища. Використовуючи цю технологію, UV Guardian може виявляти об'єкти і перешкоди на своєму шляху і динамічно планувати ефективні маршрути для задачі дезінфекції території за якомога менший час.

Робот постачається з невеликою спеціально розробленою док-станцією яка заряджає робота, якщо системі це необхідно. Док-станцію можна побачити на рисунку 1.11, робот під'їжджає і паркується в автоматичному режимі, орієнтуючись на данні з камери. При необхідності людина може використовувати робота у ручному режимі за допомогою розташованого на роботі сенсорного дисплею.

Рисунок 1.11 – Процес паркування роботу UV Guardian до док станції

Ще однією технологією для боротьби з мікроорганізмами в роботах-прибиральниках, яка здебільшого застосовується як доповнення, є

дезінфікуючі розпилювачі. Прикладом може послужити роботизована підлогомийна машина Scrubber 50 від компанії Gaussian Robotics (рисунок 1.12).

Рисунок 1.12 – Зовнішній вигляд робота Gaussian Scrubber 50

За своїм принципом роботи очищення полу робот Scrubber 50 схожий з RA660 Navi та KIRA B 50. Відмінність лише в тому, що він доповнений дезінфекційним розпилювачем (рисунок 1.13) який, під управлінням програмного забезпечення, дозволяє рівномірно розпилювати у повітрі дрібнодисперсний туман дезінфекційного розчину, забезпечуючи ретельне покриття поверхонь.

Рисунок 1.13 – Gausium Scrubber 50, виконує розпилювальну дезінфекцію

Аналізуючи вищенаведені робототехнічні рішення, можна помітити, що компанії у своїх рішеннях створення робота-прибиральника використовують за основу конструктивні рішення підлогомийних машини, які широко використовується в комерційних і промислових приміщеннях для ефективного прибирання та догляду за великими площами підлог. Машина в собі поєднує функції підлогонатирача і пилососа, завдяки чому ефективно видаляє бруд, пил, плями і сміття з різних типів підлогових покриттів. Типова конструкція такої машини наведена на рисунку 1.14. Основні частини, які присутні у всіх типових підлогомийних машинах, в себе включають:

- бак для розчину;
- щітки/пади для чищення;
- система скребків;
- вакуумна система;

- бак для збору води;
- джерело живлення;
- панель управління.

Бак для розчину – це місце, де зберігається миючий розчин або миючий засіб. Зазвичай він виготовляється з міцних і стійких до корозії матеріалів, щоб витримувати хімічні речовини, які використовуються для чищення.

Щітки/пади для чищення – це обертові щітки, які безпосередньо контактують з поверхнею підлоги. Ці щітки або губки перемішують бруд і плями, розпушуючи їх для ефективного очищення. Тип щіток або подушечок залежить від матеріалу підлоги та інтенсивності очищення.

Система скребків – це система, яка відповідає за збір брудної води та миючого розчину з підлоги після миття. Вона складається з гумового леза, яке притискається до поверхні підлоги, щоб видалити рідину.

Вакуумна система – це система підлогомиючої машини, що оснащена потужною вакуумною системою, яка всмоктує брудну воду і сміття, зібране скребком.

Бак для збору води – це резервуар для збирання брудної води та сміття під час прибирання. Бак відокремлений від бака для розчину, щоб підтримувати гігієну та запобігати перехресному забрудненню.

Панель управління зазвичай розташована на ручці. Вона дозволяє оператору керувати різними функціями машини, такими як швидкість обертання щітки, швидкість потоку розчину та сила вакууму. Деякі вдосконалені моделі можуть мати додаткові функції, такі як цифрові дисплеї та програмовані режими прибирання.

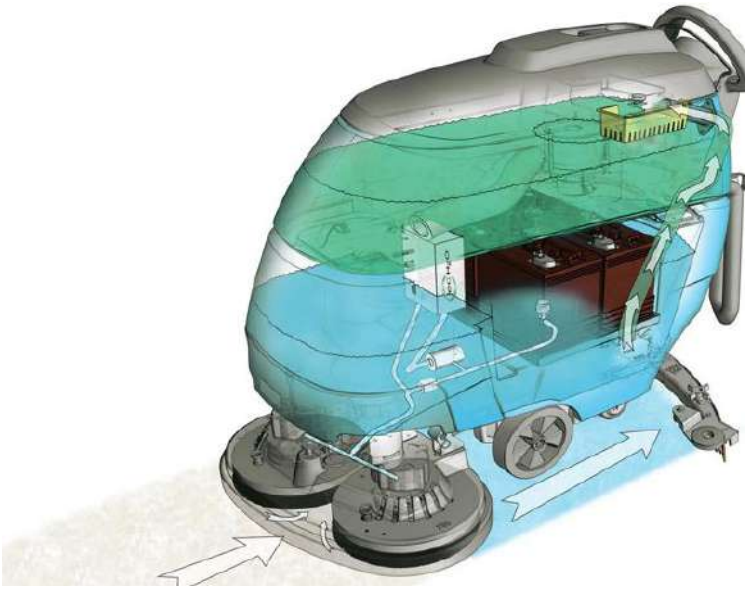


Рисунок 1.14 – Основні частини типової підлогомийноїмашини

1.3 Технічне завдання

Під час предметного аналізу було встановлено, що розглянуті вище автономні системи прибирання, засновані на роботах, передбачають інтеграцію традиційних рішень машин для миття підлоги з передовими технологіями. Поєднуючи сенсори, камери та алгоритми машинного навчання, завдяки чому системи можуть орієнтуватися, прибирати та адаптуватися до різних умов.

Таким чином, необхідно розробити систему для виконання завдання з миття підлоги, яка має наступні характеристики:

- можливість автономної роботи протягом 1 год;
- вартість розробки має складати не більше 20 000 грн;
- можливість системи підтримувати до 3 підлогомийних апаратів;
- можливість системи помити площу 420 км^2;

- можливість простого і гнучкого налаштування поведінки кожного підлого мийного апарата;
- підлогомийні апарати повинні автоматично заряджатись й заправлятись заїжджаючи на спеціально відведене місце;

Для реалізації технічного завдання, було ухвалено рішення створити систему управління, що дасть змогу модифікувати типові підлогомийні машини в автономні роботизовані системи орієнтовані для прибирання у лікарні. Такий підхід дасть змогу уникнути трудомісткого та дорогого процесу розробки власної конструкції з усіма необхідними вузлами, що їх поєднує в собі типова підлогомийна машина. Використовуючи конструкцію готової підлогомийної машини, усувається необхідність розробки власного проекту з нуля, більше зосередившись на розробці уніфікованого ядра управління яке через сенсори буде забезпечувати зворотний зв'язок з навколишнім середовищем, дозволяючи системі виявляти перешкоди, переміщатися навколо них і оптимізувати траєкторію прибирання підлогомийної машини.

2 ПРОЄКТУВАННЯ СИСТЕМИ КЕРУВАННЯ ПІДЛОГОМИЙНОЇ МАШИНИ ТА ДОК-СТАНЦІЇ ДО НЕЇ

У наведеному розділі описується підхід до створення структурної схеми системи управління для підлогомийних машин і док-станцій. Основним завданням під час розроблення системи є створення системи, яка на виході дасть змогу без особливих зусиль інтегрувати себе до будь-якої готової системи підлогомийних машин, наділивши її усією необхідною функціональністю, що потрібна для виконання прибирання в автономному режимі у лікарні.

2.1 Основні вимоги до системи

Для створення системи управління, згідно вимогам технічно завдання в п. 1.3, необхідно спроектувати та зібрати апаратні компоненти, використовуючи сучасні модулі, які доступні на ринку. Система має забезпечувати необхідну продуктивність і підтримку програмного забезпечення штучного інтелекту, алгоритмів комп'ютерного зору та підключення декількох сенсорів.

Сенсори відіграють важливу роль, надаючи дані системі про зовнішнє середовище, які необхідні для виконання системою поставлених завдань. Отже, апаратне забезпечення повинно мати достатню обчислювальну потужність для опрацювання вхідних даних від кількох сенсорів та одночасного виконання алгоритмів, пов'язаних із поведінкою робота (таких як комп'ютерний зір, навчання, реакція та позиціонування).

У процесі проектування системи для початку було поставлено завдання створити загальну структурну схему розроблюваної системи та її функціональних алгоритмів. Це послужить основою для структурованого розроблення, що забезпечує досягнення поставлених у роботі цілей.

2.2 Загальна структурна схема системи

Розроблювана система управління, яка націлена розширити можливості підлогомийних машин до автономного роботизованого пристрою прибирання в лікарні, представляє собою систему, яка інтегрується у систему підлогомийної машини. Загальна структурна схема (рисунок 2.1) роботизованої системи прибирання складається з підлогомийного робота і док-станції. Підлогомийний робот є результатом поєднання двох систем – системи управління, що розробляється і готової системи підлогомийної машини.

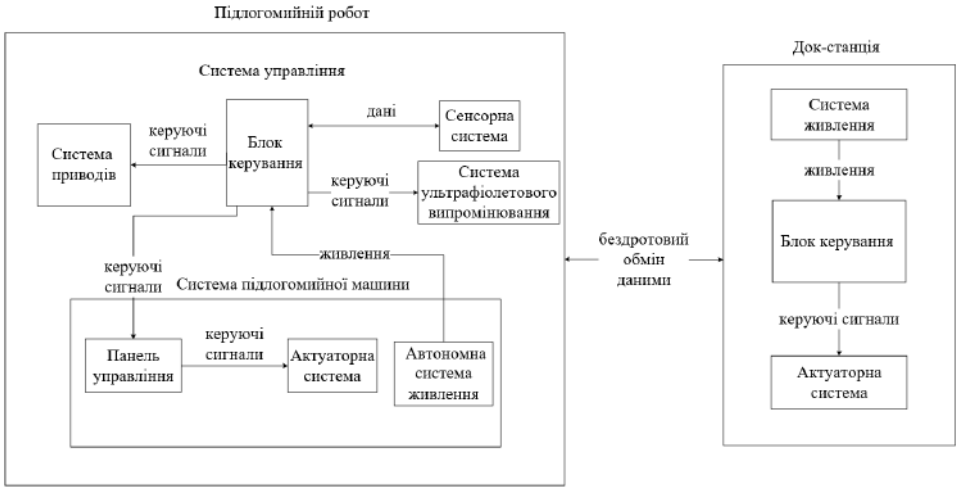


Рисунок 2.1 – Загальна структурна схема розроблювальної роботизованої системи

2.2.1 Система управління

Розроблювальна система управління складається з двох компонентів:

- блоку керування;
- сенсорної системи;
- системи ультрафіолетового випромінювання;

- система приводів.

Блок керування відповідає за обробку та виконання команд. Цей блок призначений для управління і регулювання різних аспектів роботи робота. Він отримує вхідні дані з різних джерел, таких як дані з док-станції, бортові сенсори та телеметричні дані, і перетворює їх на відповідні сигнали управління, які подаються на блок панелі управління системи підлогомийної машини. Для живлення блоку керування він під'єднується до автономного джерела живлення системи підлогомийної машини, що забезпечує автономну та незалежну роботу блоку.

Сенсорна система відповідає за збір даних про навколишнє середовище в режимі реального часу.

Системи ультрафіолетового випромінювання відповідає за дезінфекцію поверхонь.

Система приводів системи управління відповідає за рух робота. Підлогомийні машини не мають у собі систему приводів, оскільки розраховані на ручне керування оператором, який штовхає або направляє машину по підлозі.

2.2.2 Система підлогомийної машини

Беручи до уваги дані, отримані з предметного аналізу, що проводився у першій частині представленої дипломної роботи магістра, було сформовано загальну структурну схему системи типових підлогомийних машин. Система підлогомийних машин складеться з таких компонентів:

- панелі управління;
- актуаторної системи;
- автономної системи живлення.

Панель управління отримує сигнали управління від блоку керування і формує керуючі сигнали що потрапляють до актуаторної системи.

Актуаторна система приймає сигнали з панелі управління і реагує на сигнали здійснюючи необхідну фізичну дію на навколишнє середовище.

Автономна система живлення забезпечує систему управління і систему підлогомийної машини достатньою кількістю енергії протягом певного часу.

2.2.3 Док-станція

Док-станція слугує базою, до якої підлогомийний робот може під'їжджати для підзарядки акумулятора або поповнення запасу води. Док-станція складається з таких компонентів:

- система живлення;
- блок керування;
- актуаторна система.

Блок управління отримує дані від системи управління бездротовим способом через канал обміну даними і надсилає відповідні сигнали на актуаторну систему.

Актуаторна система забезпечує фізичну маніпуляцію з конструкціями для поповнення ресурсів підлогомийного робота.

Блок живлення забезпечує необхідний рівень напруги для належної роботи компонентів док-станції.

2.3 Розробка структурної схеми системи управління

Структурна схема розроблювальної системи управління, яку можна побачити на рисунку 2.2, складається з наступних компонентів:

- плата CPU;
- перетворювачем постійного струму (DC-DC);
- блок елементу розв’язки;
- два контролера;
- два електродвигуна;
- стрічка ультрафіолетових світлодіодів (LEDs UVC Lamp Sterilizer);
- двох гіроскопів;
- лідар;

- відеокамера.

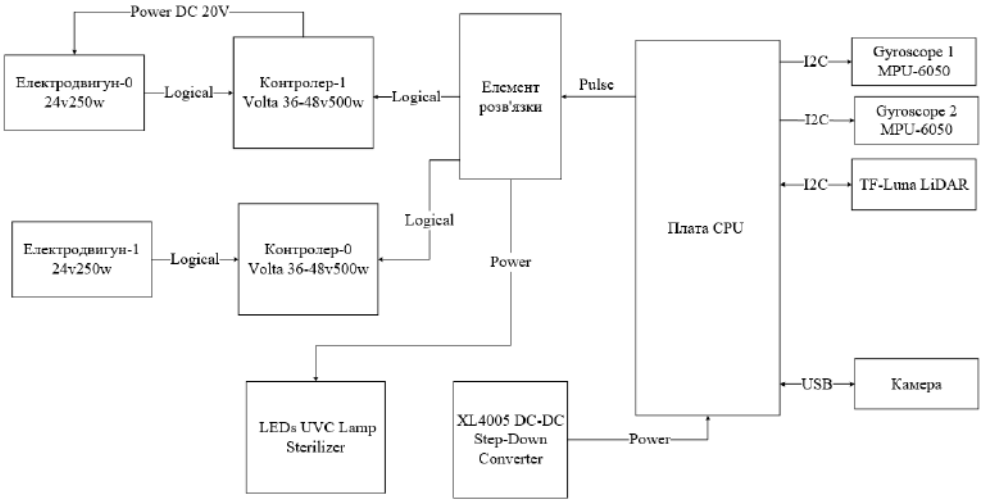


Рисунок 2.2 – Структурна схема системи управління

Центром усіх обчислювальних операцій є плата CPU, вона виконує обробку вхідних сигналів з сенсорів та виконує необхідне обчислення алгоритмів позиціонування, комунікації, реагування і бачення, які реалізують поведінку робота. Обробивши сигнал або виконавши певній алгоритм, плата CPU генерує керуючі сигнали.

Перетворювач постійного струму (DC-DC) – це понижувальний перетворювач постійного струму, який використовується для зниження напруги постійного струму з одного рівня на інший. Він забезпечує перетворення постійної напруги на вході з автономної системи живлення підлогомийної машини в меншу постійну напругу.

Блок елементу розв’язки виконує функцію повторення і посилення сигналів керування, забезпечуючи захист процесорної плати від непередбачених стрибків напруги. Виконуючи роль посередника між сигналами управління і процесорною платою, цей блок забезпечує точне відтворення і посилення сигналів управління, забезпечуючи їхню безперебійну передачу по всій схемі. Крім того, він служить захистом від

різких коливань напруги, ефективно пригнічуючи і пом'якшуючи шкідливий вплив стрибків, які потенційно можуть пошкодити або порушити роботу процесорної плати. Виконуючи ці важливі завдання, блок роз'єднувальних елементів сприяє загальній стабільності, стійкості та довговічності системної плати процесора в системі.

Два блоки контролерів приймають сигнали керування з блоку плати CPU і перетворюють постійний струм від акумулятора підлогомийної машини в сигнали постійного струму, що керують електродвигуном. Блоки контролерів використовують різні алгоритми керування для регулювання напруги, частоти та форми сигналів струму відповідно до бажаної швидкості, крутного моменту та ефективності двигуна. Крім того, контролер забезпечує захист від перевантаження по струму, перегріву та інших електричних несправностей, які потенційно можуть пошкодити двигун або систему.

Два електродвигуна – це потужні елементи мехатронної системи, що реалізують систему приводів системи управління і відповідають за пересування робота.

Стрічка ультрафіолетових світлодіодів (LEDs UVC Lamp Sterilizer) – це світлодіодний стерилізатор, сучасний пристрій для дезінфекції, який використовує технологію світлодіодів (LED) для знищення шкідливих мікроорганізмів з різних поверхонь. Він працює за принципом використання певних довжин хвиль світла для націлювання і знищення бактерій, вірусів та інших патогенних мікроорганізмів. Стрічка містить високоінтенсивні ультрафіолетові світлодіоди, які випромінюють світло, як правило, з довжиною хвилі близько 254 нм.

Лідар, гіроскопи, і камера складають сенсорну систему, вона забезпечує систему управління інтерпретованими даними з навколишнього середовища. Данні пустують до плати CPU. На основі отриманих даних CPU модуль будує 2D карту оточення і розташування об'єктів, завдяки камері відбувається розпізнавання об’єктів.

2.4 Розробка структурної схеми док-станції

На рисунку 2.3 наведена структурна схема док-станції, що складається з наступних компонентів:

- плати CPU;
- блоку живлення;
- перетворювачу постійного струму (DC-DC);
- 4 драйвера StepStick A4988;
- реле помпи.

Процесорна плата CPU є невід'ємною частиною блоку управління всередині док-станції. Її основна функція – виконувати роль сервера, отримуючи бездротовим способом дані від системи управління і виконуючи відповідні дії.

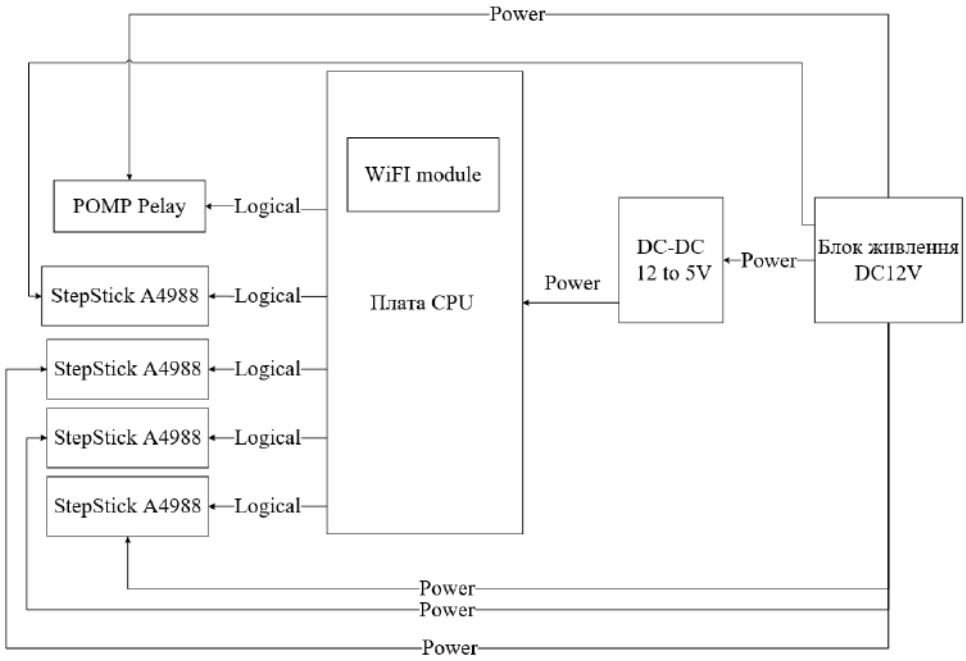


Рисунок 2.3 – Структурна схема док-станції

Блок живлення та DC-DC перетворювач забезпечують необхідного електроживлення. Вони відповідають за забезпечення необхідного рівня постійної напруги. Зокрема, блок живлення забезпечує постійну напругу 12 В для роботи насоса і приводів. Тим часом DC-DC перетворювач знижує напругу до 5 В, забезпечуючи оптимальну роботу системної плати.

Драйвери StepStick A4988, які відповідають за керування кроковим двигуном, реалізують актуаторну систему док-станції.

Реле помпи відіграє вирішальну роль, забезпечуючи контрольовану подачу води до системи підлогомийної машини.

2.5 Функціонування системи

Як вже зазначалось, розроблювана система управління інтегрується в підлогомийну машину, утворюючи автономну систему – підлогомийний робот. Блок керування, що є елементом системи управління, обробляє інформацію від декількох сенсорів і приймає відповідні рішення. Кожен сенсор у системі надає конкретні дані відповідно до свого призначення. Аналізуючи дані, система управління може орієнтуватись у просторі і пересуватись та взаємодіяти з навколишнім середовищем, посилаючи керуючі сигнали на систему приводів своєї системи, а також до панелі управління підлогомийної машини.

Підлогомийні машини володіють мінімальним інтерфейсом управління для користувачів, який представляє собою кнопки, тумблери, різного виду перемикачі, які зазвичай розташовані на рукоятці підлогомийної машини, що з технічного погляду представляють собою ключі електричного ланцюга. Ці ключі дають змогу управляти актуаторною системою підлогомийної машини, що являє собою набір електромеханічних пристроїв різної потужності. Блок розв’язки системи управління замикає ключі панелі управління підлогомийної машини, таким чином відбувається керування системами, що належать підлогомийній машині.

Як відомо, в лікарнях є режим, якого дотримуються медперсонал і пацієнти. Робот-прибиральник враховує цей режим. Коли приходить час, робот використовує інформацію щодо плану поверху лікарні, яку завантажив оператор підлогомийного робота, і подорожує до місця прибирання палати. Якщо блок управління робота прогнозує недостатню кількість води в баку для наступного прибирання або низький рівень енергопостачання, він повертається до док-станції для поповнення своїх ресурсів.

Після успішного паркування на стикувальному вузлі робот бездротовим зв'язком передає сигнал запиту. Отримавши сигнал, док-станція використовує свою систему приводів, щоб забезпечити механічні пристрої для поповнення запасів води або зарядки системи живлення робота. Після завершення док-станція посилає керуючі сигнали для втягування механічних пристроїв, які використовувалися для поповнення запасів, та інформує робота про успішне поповнення запасів. Після отримання цієї інформації безпілотник відновлює виконання поставлених завдань відповідно до вказівок блоку управління.

2.5.1 Алгоритм функціювання системи

Вся система працює завдяки взаємодії двох алгоритмів, один для робота, а інший для док-станції. Функціональність алгоритму для робота зображена на рисунку 2.4. Спочатку робот намагається виявити найближчу док-станцію. Якщо обмін даними між роботом і док-станцією відбувається успішно, алгоритм переходить до наступного етапу, який передбачає перевірку роботи системи сенсорів і виконавчих механізмів. У разі збою на цьому етапі система робота записує звіт про помилку і припиняє свою роботу. Однак, якщо всі системи пройшли перевірку успішно, алгоритм робота входить у нескінченний цикл.

У цьому циклі робот безперервно обробляє дані, отримані від системи сенсорів. На основі цих даних алгоритм поведінки всередині робот аналізує і структурує інформацію, роблячи її придатною для виконання алгоритмів

нечіткої логіки і різних методів прийняття рішень. Ці методи дозволяють системі правильно інтерпретувати інформацію і реагувати, надсилаючи керуючі сигнали до виконавчої системи. Алгоритм поведінки робота – це складний багатопотоковий процес, покликаний сприяти ефективній обробці даних і прийняттю рішень.

Загальний алгоритм роботи док-станції проілюстровано на рисунку 2.5. Після активації док-станції вона переходить у стан очікування на отримання даних від робота. Після отримання даних алгоритм входить у нескінченний цикл, де виконується поведінковий процес. Цей процес передбачає аналіз отриманих даних від робота та надсилання відповідних керуючих сигналів до виконавчої системи док-станції. Якщо алгоритм поведінки не генерує жодного керуючого сигналу, відбувається вихід з нескінченного циклу. Вихід із циклу супроводжується надсиланням даних до роботу, що свідчить про успішне завершення процесу обслуговування та перехід у стан прослуховування даних.

Загалом, алгоритми системи забезпечують комплексну основу для ефективної взаємодії робота та док-станції. Завдяки безперервному обміну даними, обробці та прийняттю рішень система забезпечує надійну та ефективну роботу, підтримуючи необхідний зв'язок та координацію між двома компонентами.

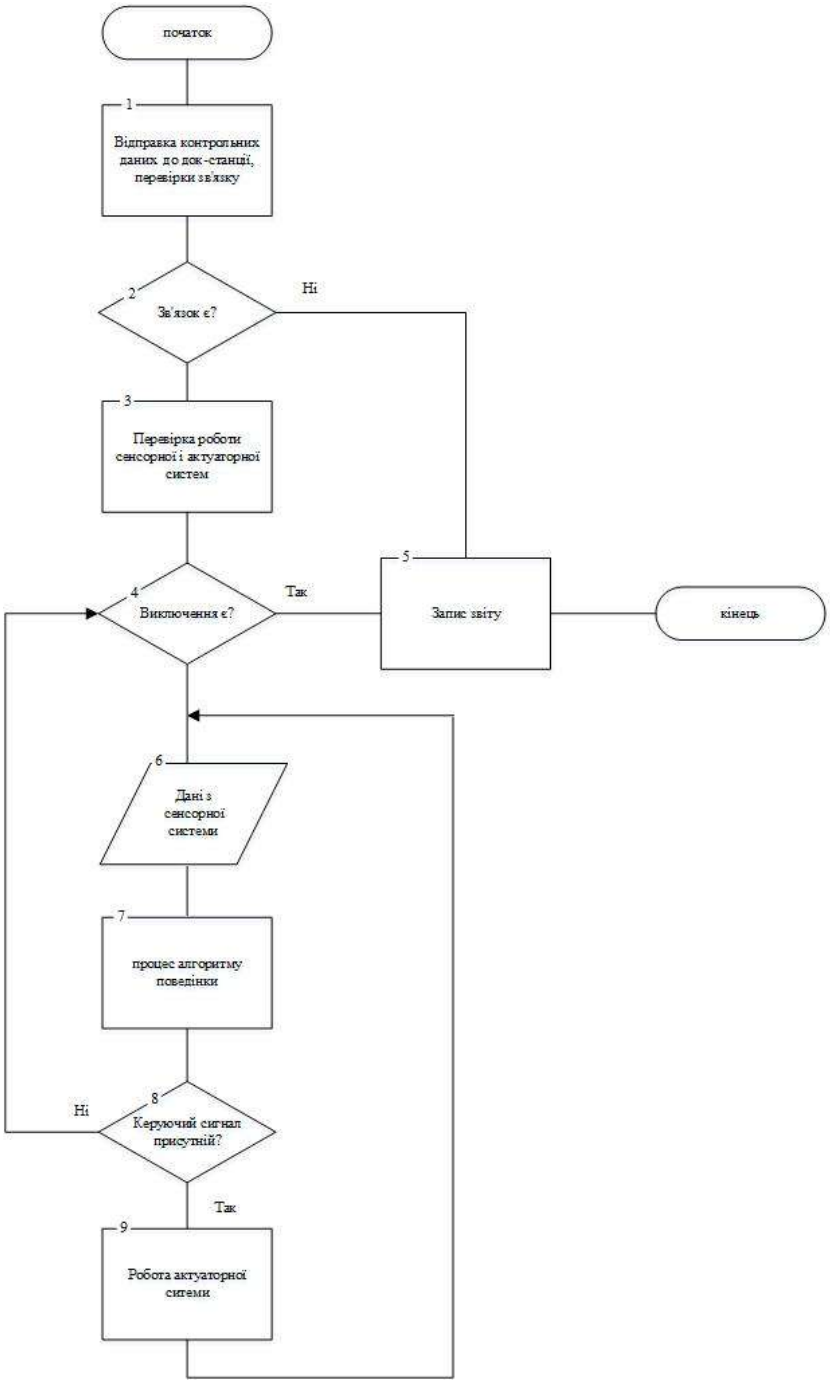


Рисунок 2.4 – Загальний алгоритм функціювання підлогомийного робота

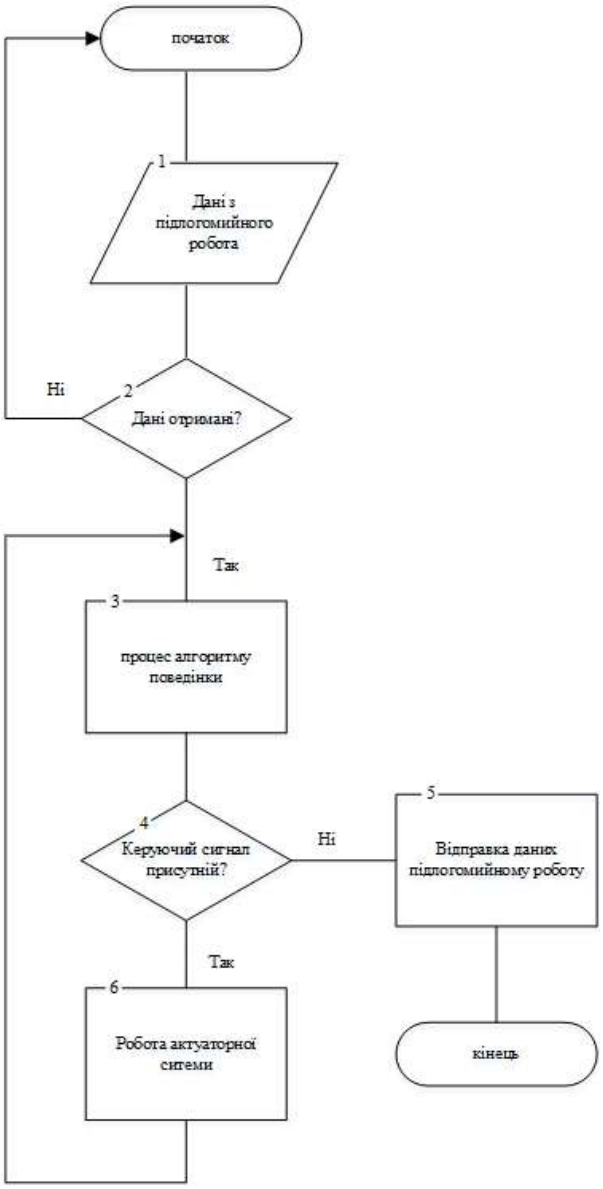


Рисунок 2.5 – Загальний алгоритм функціювання док-станції

3 АПАРАТНЕ ЗАБЕЗПЕЧЕННЯ РОЗРОБЛЮВАНОЇ СИСТЕМИ

У приведеному розділі описано стратегію розробки електричної схеми для роботизованої системи прибирання. Ця схема охоплює обчислювальну, сенсорну та виконавчу системи. Вибрані модулі були обрані на основі їхніх конкурентних переваг. Протягом усього процесу розробки було також розглянуто інтегрування існуючих рішень та ідеї з пристроїв, розглянутих у першому розділі. Ця інтеграція мала на меті прискорити загальний графік розробки. Основною метою цього розділу є створення системи, здатної одночасно отримувати сигнали від декількох сенсорів, використовуючи при цьому алгоритми позиціонування і комп'ютерного зору для прийняття обґрунтованих рішень.

Щоб розпочати процес розробки, ретельна увага була приділена вибору відповідних модулів для електричної схеми. Ці модулі відіграють життєво важливу роль у забезпеченні бажаної функціональності роботизованої платформи. Впроваджуючи готові рішення і використовуючи існуючі технології та концепції, на виході спрощується обслуговування готового пристрою, а також заощаджується цінний час і ресурси на розробку роботизованої системи.

3.1 Вибір стеку технологій

Для реалізації технічного завдання дипломної роботи магістра було ухвалено рішення розроблювати систему з використанням готових технічних рішень – модулів. Ці модульні компоненти можуть включати механічні частини, сенсори, виконавчі механізми, програмні модулі тощо.

Однією з ключових переваг модульного підходу є підвищена гнучкість та універсальність. Завдяки використанню взаємозамінних модулів, роботизовані системи можуть бути швидко реконфігуровані або модернізовані для задоволення вимог, що змінюються, або виконання нових

завдань без необхідності значного перероблення або складання. Така модульність дає змогу швидко створювати прототипи та впроваджувати роботизовані системи, заощаджуючи час і ресурси.

Ще однією перевагою є поліпшення технічного обслуговування та масштабованості. Завдяки модульним компонентам несправні або зношені деталі можна легко виявити і замінити, що зводить до мінімуму час простою і знижує витрати на обслуговування. Крім того, модульні робототехнічні системи можна збільшити або зменшити шляхом додавання або видалення модулів, що дає змогу легко налаштовувати й адаптувати їх до різних застосунків або робочих просторів.

Такої практики дотримуються не тільки стартапи, а й інші організації в галузі робототехніки. Дотримуючись цих принципів, розробники можуть досягти інтероперабельності та гнучкості своїх мобільних робототехнічних систем, забезпечуючи безперешкодну інтеграцію та ефективну роботу.

3.2 Блок керування системи управління

Для блоку керування було розглянуто три пристрої, це мікрокомп’ютери:

- BeagleBone Black Rev C 4GB ASUS4 (рисунок 3.1);
- Raspberry Pi 4 Model B (рисунок 3.2);
- Tinker Board: ASUS Tinker Board (рисунок 3.3).

Рисунок 3.1 – Плата BeagleBone Black Rev C 4GB ASUS4

BeagleBone Black Rev C 4GB – це потужний мікрокомп'ютер, призначений для вбудованих систем і робототехніки [8]. Він оснащений процесором ARM Cortex-A8 AM335x 1 ГГц у поєднанні з 512 МБ оперативної пам'яті DDR3. Мікрокомп'ютер пропонує безліч варіантів підключення, включно з портами USB 2.0, Ethernet, виходом HDMI і накопичувачем microSD. Він також має 65 контактів GPIO (General Purpose Input/Output), що дає змогу легко взаємодіяти із зовнішніми пристроями та сенсорами. BeagleBone Black відомий своєю гнучкістю і розширюваністю, завдяки розробці з відкритим вихідним кодом і сумісності з різними дистрибутивами Linux. Це дає розробникам свободу в налаштуванні та адаптації системи під свої конкретні потреби. Однак одним із головних недоліків BeagleBone Black є відносно обмежений об'єм оперативної пам'яті, що може обмежити продуктивність у ресурсномістких додатках. Крім того, відсутність вбудованого бездротового зв'язку, спонукає до покупки WI-FI SUB модулю.

Рисунок 3.2 – Плата Raspberry Pi 4 Model B

Raspberry Pi 4 Model B - це передостанній мікрокомп’ютер з серії одноплатних мікрокомп'ютерів Raspberry Pi, розроблених Raspberry Pi

Foundation. Вона була розроблена для сприяння викладанню базової інформатики та навичок програмування в школах, але набула широкої популярності і зараз використовується в різних проектах і додатках.

Плати Raspberry Pi відомі своєю доступністю, універсальністю та простотою використання. Вони мають невеликий форм-фактор і оснащені досить потужними апаратними компонентами. Передостанні моделі, такі як Raspberry Pi 4 Model B та Raspberry Pi 4 Model B+, пропонують значне покращення продуктивності порівняно з попередніми версіями.

Raspberry Pi 4 Model B оснащений чотирьохядерним процесором ARM Cortex-A72 з тактовою частотою 1,5 ГГц і 2 ГБ, 4 ГБ або 8 ГБ (B+) оперативної пам'яті LPDDR4, залежно від моделі. Він має кілька портів USB, включаючи USB 3.0, вихід HDMI, Ethernet, Wi-Fi, Bluetooth і слот для карт пам'яті microSD для зберігання даних. Він також має роз'єми GPIO для взаємодії з зовнішніми пристроями та сенсорами.

Raspberry Pi підтримує різні операційні системи, включаючи дистрибутиви Linux, такі як Raspbian (тепер називається Raspberry Pi OS), Ubuntu та інші. Це дозволяє користувачам запускати широкий спектр програмних додатків, від веб-серверів і медіацентрів до систем домашньої автоматизації та робототехнічних проектів [9]. Raspberry Pi також має жваву та активну спільноту, яка надає вичерпну документацію, навчальні посібники та величезне сховище проектів і бібліотек програмного забезпечення.

Незважаючи на свої численні переваги, Raspberry Pi має деякі обмеження. Одним з головних недоліків є обмежені можливості графічного процесора, що може вплинути на роботу графічно інтенсивних додатків. Крім того, відсутність вбудованої пам'яті означає, що користувачам доводиться покладатися на зовнішні пристрої зберігання даних, як правило, на карти пам'яті microSD. Хоча Raspberry Pi з роками покращив можливості підключення, йому все ще не вистачає деяких функцій, які зазвичай зустрічаються в інших мікрокомп'ютерах, наприклад, вбудованої підтримки SATA або більшої кількості портів USB 3.0.

Тим не менш, Raspberry Pi став популярним вибором для аматорів, викладачів та професіоналів завдяки своїй доступності, універсальності та широкій підтримці спільноти. Він дав можливість окремим особам та організаціям досліджувати та створювати інноваційні проекти в різних сферах, від освіти та домашньої автоматизації до IoT (Інтернету речей) і не тільки.

Рисунок 3.3 – плата Tinker Board: ASUS Tinker Board

ASUS Tinker Board: ASUS Tinker Board – це багатофункціональний мікрокомп'ютер, призначений для високопродуктивних обчислень і мультимедійних можливостей. Він оснащений чотирьохядерним процесором ARM Cortex-A17 з тактовою частотою 1,8 ГГц і 2 ГБ оперативної пам'яті LPDDR3. Мікрокомп'ютер оснащений графічним процесором Mali-T764, що забезпечує підтримку відтворення відео 4K і прискорення графіки OpenGL ES 3.0. Він пропонує безліч варіантів підключення, включно з виходом HDMI, чотирма портами USB 2.0, Ethernet, Wi-Fi і Bluetooth. ASUS Tinker Board. Мікрокомп’ютер підтримує різні операційні системи, включно з дистрибутивами Linux, такими як Debian і Android [10]. Однак одним із головних недоліків ASUS Tinker Board є відносно обмежена підтримка спільноти та сумісність програмного забезпечення порівняно з більш популярними платформами, такими як Raspberry Pi. Це може призвести до

меншої екосистеми проєктів і ресурсів, доступних для користувачів. Крім того, доступність аксесуарів і доповнень для ASUS Tinker Board може бути більш обмеженою порівняно з Raspberry Pi, завдяки меншої популярності, що може вплинути на можливості розширення та налаштування.

Завдяки своєї популярності і доступності на вибір блоку керування було обрано мікрокомп'ютер Raspberry Pi 4 Model B.

3.3 Сенсорна система системи управління

Для реалізації блоку сенсорної системи, яка була описана у підрозділі 2.2, були обрані такі готові апаратні пристрої, які спроможні наділити роботизовану систему прибирання можливістю взаємодії з навколишнім середовищем. Сенсорна система складеться заступних пристроїв:

- далекоміру TF Luna LiDAR;
- камери Raspberry Pi Camera Module v2;
- двох гіроскопів MPU6050.

3.3.1 Модуль виявлення відстані

Сенсор TF Luna LiDAR - це компактний і недорогий LiDAR-сенсор, розроблений компанією Benewake. LiDAR розшифровується як Light Detection and Ranging – це технологія дистанційного зондування, яка використовує лазерне світло для вимірювання відстаней і створення детальних 3D-карт навколишнього середовища. Сенсор TF Luna LiDAR спеціально розроблений для автономної навігації та уникнення перешкод в роботизованих системах, дронах та інших додатках, які потребують точних вимірювань відстані. Він має невеликий форм-фактор, низьке енергоспоживання і дальність виявлення до 8 метрів. Сенсор має невеликі розміри і малу вагу, що дозволяє легко інтегрувати його в різні пристрої і системи, не займаючи багато місця. Похибка вимірювання складає ±3 см, що дає змогу точно розраховувати відстань у приміщеннях [11].

Сенсор підтримує три інтерфейси передачі даних UART, I2C і PWM, що дозволяє гнучко інтегрувати до різних до різних пристроїв рішень або платформ. Зовнішній вигляд модуля представлено на рисунку 3.4.

Рисунок 3.4 – Зовнішній вигляд модулю TF-Luna

3.3.1.1 Доопрацювання модулю виявлення відстані

Модуль TF Luna LiDAR - це невеликий і компактний LiDAR-модуль. Він призначений в першу чергу для вимірювання відстані, і не реалізує в собі конструкцію, що надає змогу отримати 360-градусну хмару точок або в іншому діапазоні, як це мають деякі інші рішення готових LiDAR-систем.

Для того щоб лідар мав можливість сканування в 180°, для нього була розроблена конструкція корпусу, яка в подальшому друкується на 3Д-принтері. Пристрій містить двигун постійного струму, драйвер і мікроконтролер Tiny13. Вигляд конструкції можна побачити на рисунку 3.5. Платформа, на якій закріплений лідар, приводиться в періодичний рух двигуном під керуванням мікроконтролера з постійною швидкістю, коли подається напруга від системи керування. Система керування дроном не керує двигуном пристрою.

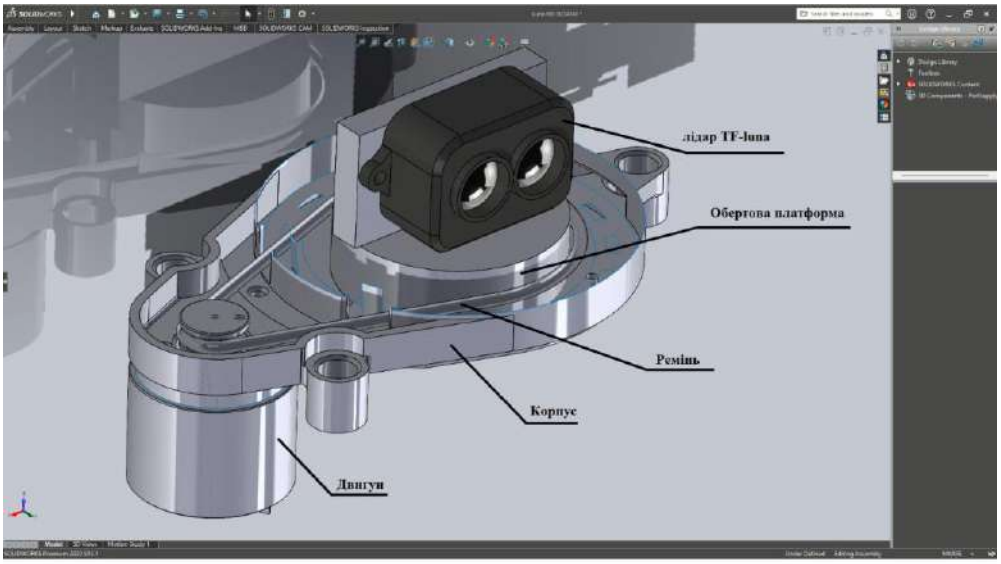


Рисунок 3.5 – Зовнішній вигляд пристрою для TF-Luna

3.4Система ультрафіолетового випромінювання

Для реалізації блоку системи ультрафіолетового випромінювання, було обрано пристрій LEDs UVC Lamp Sterilizer (рисунок 3.6), які спроможні наділити роботизовану систему прибирання можливістю виконувати очистку да дезінфекцію важкодоступних поверхонь. LEDs UVC випромінюють ультрафіолетове випромінювання на значну відстань, забезпечуючи широке покриття для ефективної дезінфекції.

Рисунок 3.6 – Модуль LEDs UVC Lamp Sterilizer

3.3.2 Камера

Raspberry Pi Camera Module v2, також відомий як Raspi Cam v2, - це офіційний модуль камери, розроблений спеціально для одноплатних комп'ютерів Raspberry Pi, зовнішній вигляд модулю можна побачити на рисунку 3.7. Він оснащений 8-мегапіксельним сенсором зображення Sony IMX219, що забезпечує високу роздільну здатність фотографій і відеозйомку.

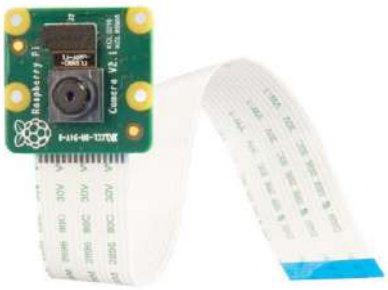


Рисунок 3.7 – Зовнішній вигляд модулю Raspi Cam v2

Максимальна роздільна здатність модуль камери 3280 x 2464 пікселів дозволяє робити деталізовані фотографії. Він підтримує різні роздільні здатності відео, зокрема 1080p зі швидкістю 30 кадрів на секунду, 720p зі швидкістю 60 кадрів на секунду і VGA зі швидкістю 90 кадрів на секунду, що забезпечує плавний відеозапис [12].

Модуль камери підключається до Raspberry Pi через порт CSI (Camera Serial Interface) за допомогою стрічкового кабелю. Цей інтерфейс забезпечує високошвидкісну передачу даних між модулем камери і Raspberry Pi.

Камера оснащена об'єктивом з фіксованим фокусом, Raspi Cam v2 пропонує поле зору (FOV) приблизно 62,2 градуси по діагоналі. Об'єктив оптимізовано для фото- та відеозйомки загального призначення.

Модуль має компактний розмір, високу якість зображення та легко інтегрується з платою Raspberry Pi .

3.3.3 Пристрій відстеження руху

MPU6050 - це популярна і універсальна інтегральна схема (IC), яка поєднує 3-осьовий акселерометр і 3-осьовий гіроскоп в одній мікросхемі, модуль представлено на рисунку 3.8. Вона призначена для відстеження руху та визначення орієнтації в різних електронних пристроях. Мікросхема використовує технологію мікроелектромеханічних систем (MEMS) для забезпечення точного та надійного зондування руху.

Рисунок 3.8 – Зовнішній вигляд модулю MPU6050

Акселерометр в MPU6050 вимірює лінійне прискорення по осях X, Y і Z, дозволяючи виявляти зміни швидкості або нахилу. Він має програмований повномасштабний діапазон, що дозволяє користувачам налаштовувати чутливість відповідно до конкретних вимог програми.

Гіроскоп MPU6050 вимірює кутову швидкість по осях X, Y і Z. Він надає точні дані про обертання і може бути використаний для визначення змін в орієнтації та обертального руху [13]. Подібно до акселерометра, гіроскоп також пропонує програмований повномасштабний діапазон для адаптації до різних рівнів кутової швидкості.

MPU6050 містить вбудований цифровий процесор руху (DMP), який виконує складні обчислення та алгоритми об'єднання сенсорів. Ця функція дозволяє мікросхемі забезпечувати об'єднаний вихід даних акселерометра і гіроскопа, спрощуючи завдання отримання точної інформації про рух.

Зв'язок з MPU6050 здійснюється через послідовний інтерфейс I2C (Inter-Integrated Circuit), що робить його сумісним з широким спектром мікроконтролерів і систем.

3.5 Система приводів системи управління

Для того, щоб роботизована система прибирання, що базується на підлогомийній машині, мала можливість пересуватися, було обрано такі пристрої, які реалізують систему приводів системи управління робота:

- два контролера Volta K24/250-DC;
- два електродвигуна 24v 250w BLDC.

3.4.1 Контролери

Контролер Volta K24/250-DC - це пристрій, призначений для точного регулювання швидкості обертання валу колекторних двигунів постійного струму. Цей контролер розроблений для двигунів постійного струму з номінальною напругою 24 вольта. Модуль пропонує зручний інтерфейс

управління, включаючи тригер дросельної заслінки або педаль акселератора. Зовнішній вигляд пристрою можна побачити на рисунку 3.9.

Рисунок 3.9 – Зовнішній вигляд контролера Volta K24/250-DC

3.4.2 Електродвигуни

Колекторний двигун постійного струму D24/250DC-D (рисунок 3.10), він використовується переважно в різних електрифікованих іграшках з електричним приводом в електричних скутерах і електричних квадроциклах, призначених для дітей у віці від 4 до 8 років. Для досягнення оптимального зчеплення з дорогою вимагає зменшення обертів коліс, підключених до цього двигуна, як правило, в 5-10 разів відносно частоти обертання валу двигуна, за допомогою системи передач.

Рисунок 3.10 – Зовнішній вигляд електродвигуна D24/250DC-D

Технічні характеристики електродвигуна D24/250DC-D наступні: він працює при номінальній напрузі 24В, з номінальною потужністю 250 Вт і максимальною вихідною потужністю 500 Вт. Номінальний струм становить 13,7 А, а максимальний струм досягає піку в 21 А при короткочасних навантаженнях. У режимі холостого ходу вал двигуна досягає максимальної швидкості обертання 2650 об/хв, яка знижується приблизно на 25-30% під дією навантаження. Цей двигун дозволяє двонаправлене обертання валу при використанні контролера, здатного змінювати напрямок обертання, або механізму зворотного ходу PR-2. Він досягає максимального ККД 80% і використовує пасивне охолодження за рахунок циркуляції повітря, що сприяє відведенню тепла від корпусу. Розміри двигуна включають довжину 107 мм (включаючи вал) і діаметр корпусу 100 мм.

3.2 Живлення системи управління

Для забезпечення живлення блоку управління системи управління, та роботи світлодіодного стерилізатора робота використовується автономна система живлення підлогомийної машини, проте її акумулятори видають близько 24 В. Для зниження напруги використовується два конвертера XL4005 DC-DC Step-Down Converter 4.-38V to 1.25.-32. Він є простим і надійним налаштовуючим понижувальним перетворювачем постійного струму на мікросхемі імпульсного перетворювача XL4005.

Рисунок 3.11 – Зовнішній вигляд конвертеру XL4005 DC-DC

3.6 Блок керування системи док станції

Для блоку керування док станції було обрано пристрій Raspberry Pi Pico W, зовнішній вигляд якого представлено на рисунку 3.11.

Pi Pico W - це компактна мікроконтролерна плата, розроблена також Raspberry Pi Foundation. Це вдосконалена версія Raspberry Pi Pico, що включає в себе можливості бездротового підключення.

Pico W побудований на базі мікроконтролера RP2040 з двоядерним процесором ARM Cortex-M0+, що працює на частоті 133 МГц. Цей чіп забезпечує необхідну обчислювальну потужність для широкого спектру вбудованих додатків. Крім того, він пропонує 264 КБ оперативної пам'яті і підтримує зовнішню флеш-пам'ять об'ємом до 16 МБ.

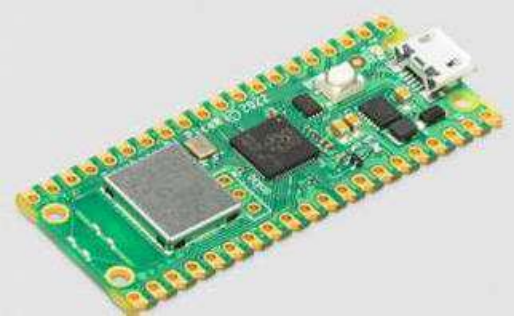


Рисунок 3.11 – Зовнішній вигляд Raspberry Pi Pico W

Бездротовий зв'язок Pico W забезпечується за допомогою вбудованої мікросхеми ESP32. Ця мікросхема забезпечує функції Wi-Fi та Bluetooth, що дозволяє Pico W спілкуватися з іншими пристроями та підключатися до Інтернету бездротовим способом. Pico W підтримує Wi-Fi 802.11b/g/n і Bluetooth 4.2.

Що стосується можливостей вводу/виводу, Pico W має 26 багатофункціональних GPIO виводів, які можна використовувати для цифрового вводу/виводу, ШІМ, I2C, SPI і UART-зв'язку [14]. Він також має

порт micro-USB для живлення і передачі даних, а також програмований світлодіод і кнопку користувача.

3.7 Актуаторна система док станції

Актуаторна система док станції утворена з чотирьох крокових двигунів NEMA 17, зовнішній вигляд якого представлено на рисунку 3.12.

Рисунок 3.12 – Зовнішній вигляд крокового двигуна NEMA 17

NEMA 17 - це широко використовувана модель крокового двигуна зі стандартним розміром корпусу 42 мм × 42 мм. Цей крокový двигун зазвичай працює при номінальній напрузі 12-24 В і має кут нахилу кроку 1,8°на крок, що дає 200 кроків за один оберт. Двигуни NEMA 17 забезпечують хороший крутний момент і точне позиціонування, що робить їх придатними для різних застосувань, включаючи 3D-принтери, верстати з ЧПК, робототехніку і системи автоматизації.

3.8 Розробка електричної схеми систем

3.8.1 Розробка електричної схеми системи управління

На рисунку 3.13 показана електрична схема системи управління, в якій представлені ключові компоненти та їхні взаємозв'язки. В основі системи лежить мікрокомп'ютер Raspberry Pi 4B, який слугує блоком керування. Він

встановлює пряме з'єднання з трьома пристроями сенсорної системи через шину I2C. Лідар TF-luna і два гіроскопи MPU-6050 складають цю сенсорну систему, що дозволяє збирати важливі дані про навколишнє середовище у приміщені лікарні для нормальної роботи підлогомийної системи. Шина I^2C, що відповідає галузевому стандарту двопровідного апаратного інтерфейсу, забезпечує передачу даних зі швидкістю до 1000 Мбіт/с завдяки підтримці стандартних тактових частот.

У схемі мікрокомп'ютер Raspberry Pi 4B виступає в ролі ведучого на шині I2C, відповідаючи за весь обмін даними і встановлюючи синхронізацію для вхідних пристроїв. Гнучкість ролі ведучого дозволяє мікрокомп'ютеру динамічно переініціалізувати шину, регулюючи час опитування сенсорів за необхідності для ефективного отримання потрібних даних від лідара і гіроскопів. Ця можливість забезпечує гнучкість та адаптивність робота до різних сценаріїв.

Для візуальної інформації робот використовує камеру Raspi Cam v2, яка взаємодіє з Raspberry Pi через порт CSI. CSI не тільки забезпечує живлення камери, але й сприяє високошвидкісній передачі даних, що дозволяє здійснювати потокове відео в реальному часі та знімати зображення.

Для того, щоб забезпечити належну розв'язку та ізоляцію сигналу, в системі управління використовується мікросхема HCPL-0600 та мікроконтролер ATtiny24 у своїй конструкції. Мікросхема HCPL-0600 слугує логічним блоком з оптичним зв'язком, що забезпечує високошвидкісний логічний інтерфейс, забезпечуючи при цьому повну ізоляцію мікрокомп'ютера Raspberry Pi 4B. Мікроконтролер ATtiny24 з'єднується з Raspberry Pi за допомогою одного дроту, підключеного до роз'єднувального елемента HCPL-060. Сконфігурований як "slave", мікроконтролер обробляє отримані пакети даних, визначаючи відповідні сигнали управління 5В для подачі на модулі, що відповідають за управління силовим ланцюгом системи приводів.

Керування двома двигунами в системі приводів здійснюється за допомогою контролерів Volta K24/250-DC. Ці контролери, спеціально розроблені для управління окремими двигунами, взаємодіють з мікроконтролером ATtiny24 через спеціалізований інтерфейс, заснований на рівнях напруги. Використовуючи цю конфігурацію, мікроконтролер точно керує роботою двигунів, забезпечуючи точне і швидке управління.

Для керування процесом заряджання мікроконтролер використовує реле SSR-25 DA. Керуючи перемиканням зарядного модуля СМ в колі, мікроконтролер контролює стан зарядки. Ця інформація отримується через сусідні контакти (PA3, PA4) між мікроконтролером і зарядним модулем, що забезпечує можливість визначити, чи зарядка триває або була успішно завершена.

Для керування роботою світлодіодного стерилізатора мікроконтролер використовує реле MS-DD6010.

3.8.2 Розробка електричної схеми док-станції

На рисунку 3.14 зображено електричнусхему док-станції, яка демонструє ключові компоненти її апаратної системи. В основі цієї системи лежить мікроконтролер Raspberry Pi Pico W, яка містить інтегрований на плату модуль Wi-Fi. Вся док-станція покладається на блок живлення, який складається з понижувального трансформатора, з'єднаного з випрямлячем. Ця комбінація ефективно перетворює вхідну енергію на напругу постійного струму 12 В, яка слугує основним джерелом живлення для електромагніту насоса.

Для забезпечення безперебійної роботи Raspberry Pi Pico W використовується модуль перетворювача L7805cv. Цей модуль знижує напругу до стабільних 5 В, гарантуючи належне функціонування мікрокомп’ютеру Raspberry Pi Pico W. Крім того, інтегральна мікросхема HCPL-0600 забезпечує оптичну розв'язку та підсилення сигналу. Вона

полегшує ізоляцію керуючих сигналів і підсилює їхню силу, забезпечуючи надійний і точний зв'язок всередині док-станції.

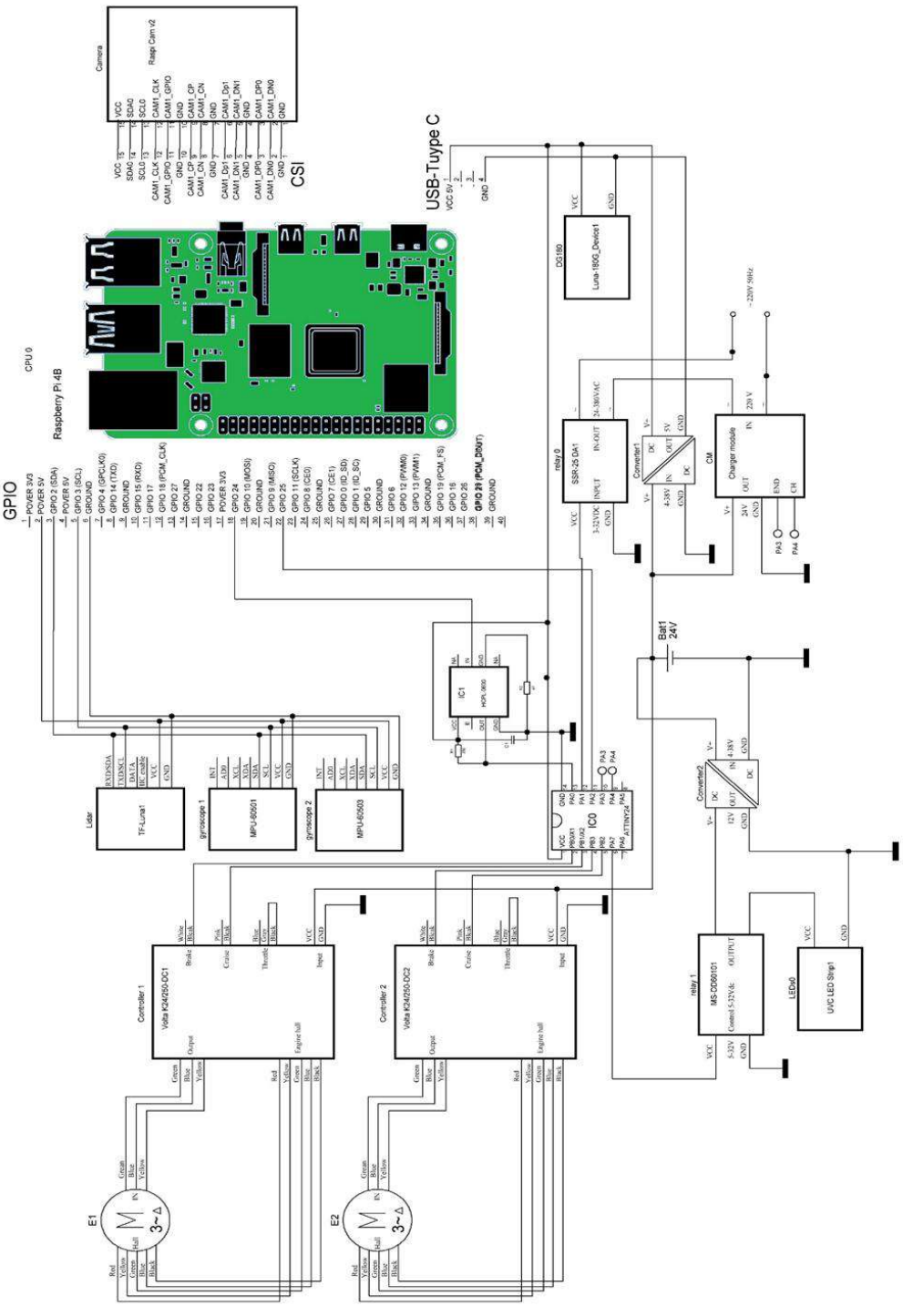


Рисунок 3.13 Схема електрична розроблюваної ситеми управління дону

Отформатировано: Украинский (Украина)

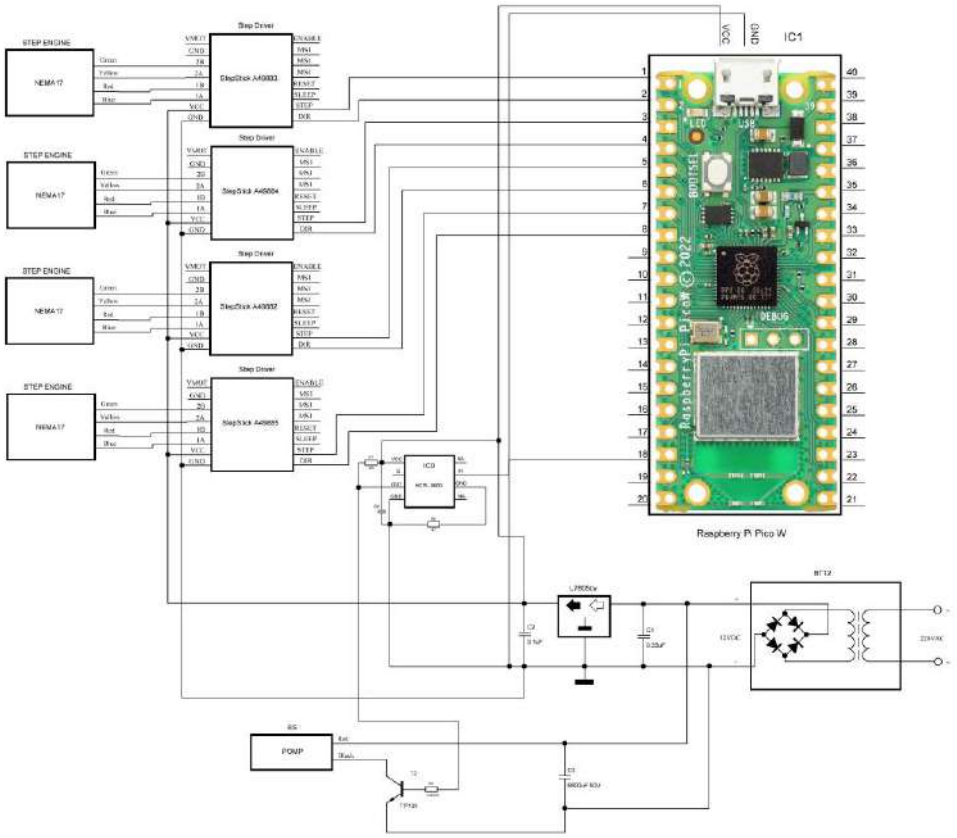


Рисунок 3.14 – Електрична схема док-станціїї

4 НАЛАШТУВАННЯ РОБОТИ СИТЕМИ ПІДЛОГОМИЙНОГО РОБОТА

4.1 Вибір стеку технологій

Як вже обговорювалось у третьому розділі дипломної роботи магістра, центральною обчислювальною системою буде виступати одноплатний комп'ютер Raspberry Pi Model B. Плата відповідає необхідним вимогам до технологій передачі даних і відповідає бажаним технічним характеристикам за обчислювальною потужністю та енергоспоживанням. Крім того, вона має компактні розміри та зручна у налаштуванні, конфігуруванні та встановленні програмних компонентів. На Rasberry Pi будуть створюватися та компілюватися бібліотеки розробки та програмне забезпечення для програмного налаштування поведінки робота. Для забезпечення зручності розробки та налаштування операційної системи спочатку буде налаштовано мережевий протокол SSH для віддаленого доступу до Rasberry Pi. Це полегшить розробку і тестування. Віддалена розробка буде виконуватись у Visual Studio Code.

4.2 Встановлення Raspberry Pi OS Lite

На сьогоднішній день для Raspberry Pi доступний широкий спектр операційних систем, кожна з яких призначена для певних цілей. При встановленні цих операційних систем розробники часто включають різноманітні програмні інструменти, компоненти, інтерпретатори та додатки, сервіси яких працювати у фоновому режимі і ніколи не використовуються, використовуючи частину ресурсів процесора. Щоб вирішити цю проблему, багато користувачів обирають Raspberry OS Lite 64-bit під час встановлення.

Raspberry OS Lite 64-bit - це мінімальна версія операційної системи, яка включає лише основні компоненти. Вона постачається з попередньо

встановленими інструментами для налаштування апаратних інтерфейсів Raspberry Pi та готовим до використання компілятором GNU GCC з інтерпретатором Python.

Для встановлення операційної системи користувачі використовують крос-платформне програмне забезпечення під назвою Raspberry Pi Imager (рисунок4.1). Цей інструмент встановлюється на інший комп'ютер і полегшує процес запису образу системи на карту microSD [15]. Він забезпечує зручний, покроковий підхід до процесу встановлення. Для оптимальної продуктивності рекомендується використовувати SD-карту з класом не нижче 10.

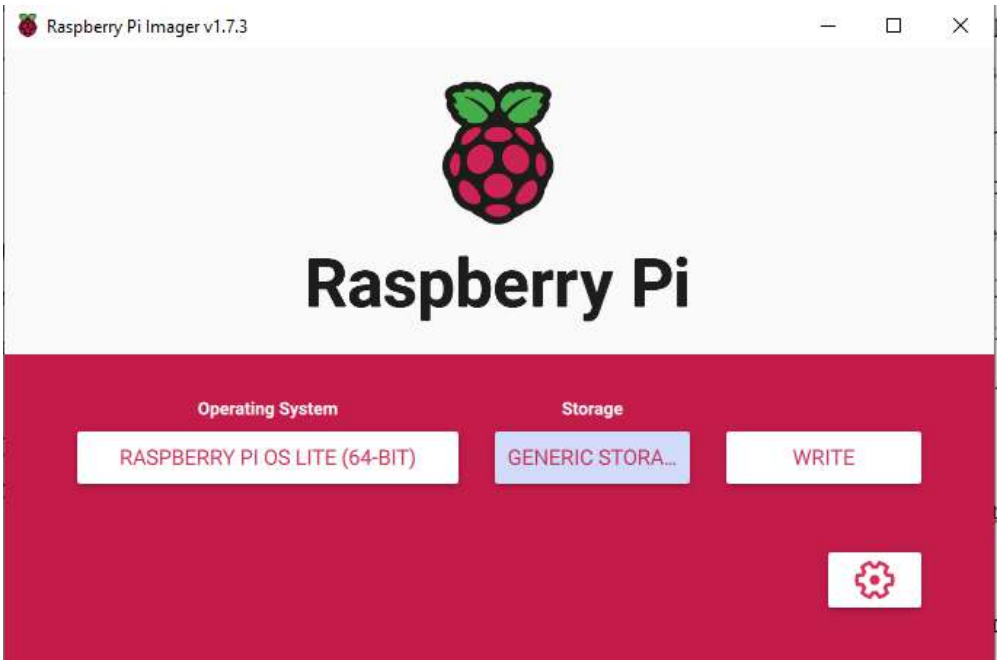


Рисунок 4.1 – Інсталятор Raspberry Pi Imager

Після успішного завантаження та встановлення програми Raspberry Pi Imager до своєї операційної системи, треба відкрити відкрито програму, щоб отримати доступ до її інтерфейсу. Далі треба натиснути кнопку "CHOOSE OS" і обрано операційну систему Raspberry Pi OS Lite (64-bit) з доступних

варіантів (рисунок 4.2). Після вибору відповідної операційної системи треба підключити SD-карту до комп'ютера через адаптер. Потім здійснити натискання на кнопку "CHOOSE STORAGE" для вибору необхідного накопичувача, після чого треба активувати кнопку "WRITE" для початку процесу інсталяції. Після завершення інсталяції SD-карту було вставлено в слот для SD-карт у Raspberry Pi 4 B.

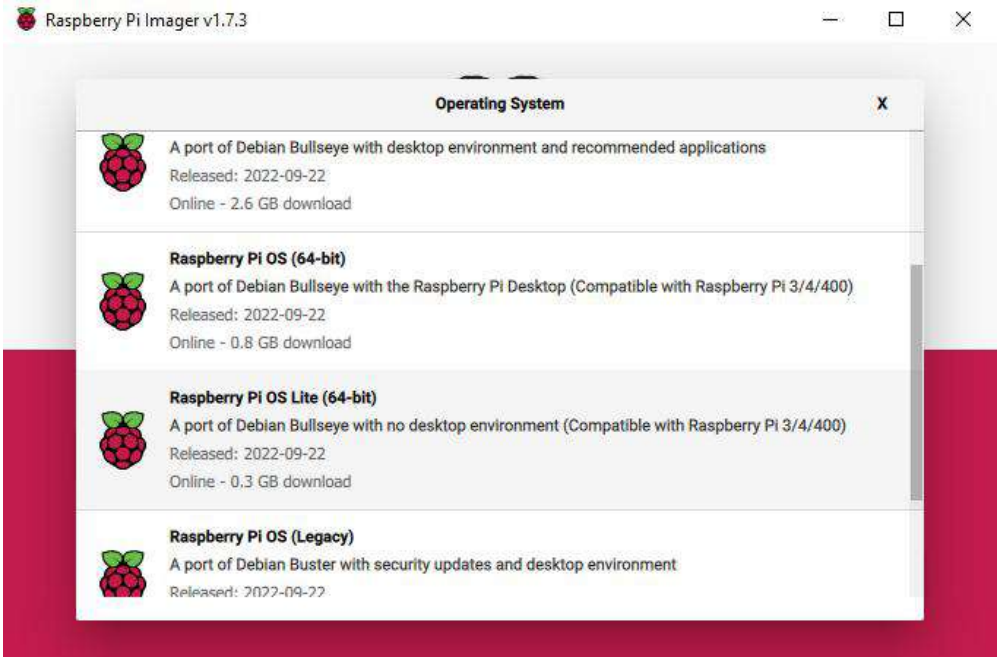


Рисунок 4.2 – Вибір Raspberry Pi OS Lite (64-bit)

У процесі початкового налаштування Raspberry Pi OS Lite необхідне під'єднання миші, клавіатури та монітора до плати для конфігурації системи. Під час першого запуску операційної системи вона запросить введення логіна і пароля для створення облікового запису користувача.

Далі для налаштування SSH серверу й підключення до бездротової мережі Wi-Fi треба виконати наступну команду:

```
$> sudo raspi-config
```

Після відкриття програми конфігурації з'являється інтерфейс, показаний на рисунку 4.3. Для налаштування з'єднання Wi-Fi, треба перейти до розділу системних параметрів, після обирати опцію бездротової локальної мережі (Wireless LAN). У цьому розділі було обрано свій регіон та назва SSID мережі і пароль до неї. Для запуску SSH сервера у головному меню було обрано параметри інтерфейсу (Intrface options), й SSH розділ. Після переходу було встановлено значення YES й виконано вихід з програми конфігурації. Для того щоб визначити IP-адресу, яку призначив DHCP маршрутизатор, треба використати наступну команду в терміналі:

```
$> hostname -I
```

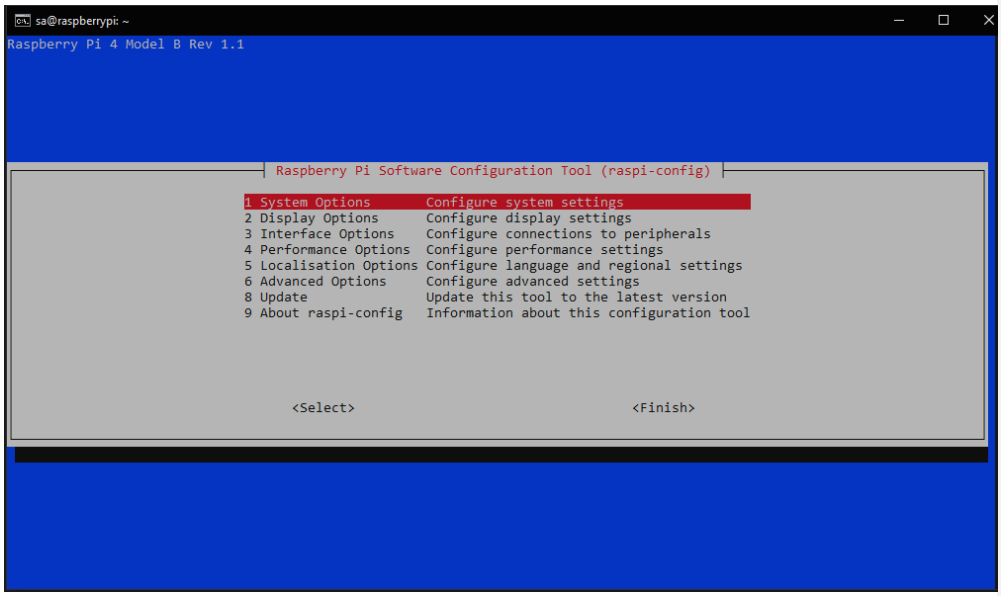


Рисунок 4.3 – Вигляд головного меню Configuration tool

Після того, коли було отримано, яка IP у пристрою Raspberry Pi у мережі, для впевнення, що робочий комп’ютер має доступ до плати, робочий комп’ютер працює на операційній системі Windows, тому було вирішено

використовувати cmd.exe (рисунок 4.4), щоб перевірити з'єднання й підключитись до Raspberry Pi, треба виконати наступни:

```
C:\> ping 192.168.0.109
C:\> ssh sa@192.168.0.109
```

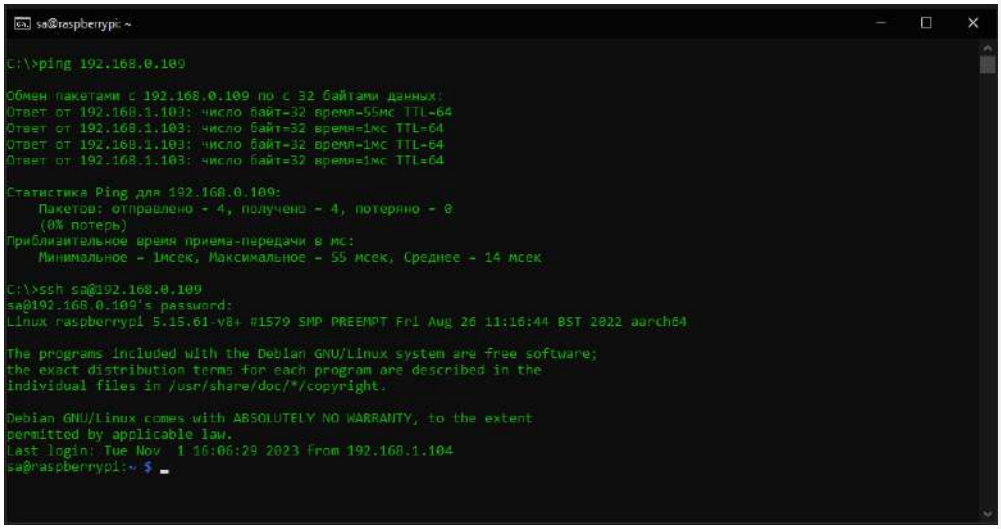


Рисунок 4.4 – Відпрацювання команд ping та ssh

4.3Встановлення додатка TensorFlow Lite Object Detection Models

TensorFlow Lite - це бібліотека програмного забезпечення з відкритим вихідним кодом, розроблена Google, яка дозволяє ефективно розгортати моделі машинного навчання на різних платформах, зокрема на пристроях з обмеженими ресурсами, таких як мобільні телефони, вбудовані системи та мікроконтролери. Вона розроблена, щоб забезпечити швидкий висновок, малий обсяг пам'яті та оптимізовану продуктивність для додатків машинного навчання на пристроях.

В основі TensorFlow Lite лежить використання стисненого формату файлів під назвою FlatBuffers для представлення моделей машинного

навчання, що зменшує обсяг пам'яті і дозволяє ефективно зберігати і передавати дані. Цей формат дозволяє швидко завантажувати та ефективно використовувати моделі на пристроях з обмеженими обчислювальними ресурсами.

TensorFlow Lite використовує декілька оптимізацій для прискорення виконання моделей машинного навчання. Він використовує апаратне прискорення за допомогою спеціалізованих бібліотек та API, таких як Android Neural Networks API (NNAPI) або Apple Core ML, щоб використовувати основні апаратні можливості пристрою, такі як графічні або нейронні процесори (NPU). Це забезпечує ефективні паралельні обчислення та зменшує затримку виведення [16].

Для подальшого підвищення продуктивності TensorFlow Lite підтримує квантування моделі – техніку, яка знижує точність ваг і активацій моделі з плаваючою точкою до форматів з меншою точністю, таких як представлення з фіксованою точкою або цілими числами. Це зменшує споживання пам'яті та обчислювальні вимоги, зберігаючи прийнятний рівень точності. Крім того, оператори оптимізовані для різних платформ за допомогою злиття ядер, коли кілька операцій об'єднуються в одне оптимізоване ядро, мінімізуючи передачу пам'яті і підвищуючи швидкість виконання.

TensorFlow Lite підтримує широкий спектр моделей машинного навчання, в тому числі побудованих за допомогою TensorFlow, популярного фреймворку для глибокого навчання. Він надає набір API для завантаження моделей, попередньої обробки вхідних даних та постобробки вихідних даних, що дозволяє легко інтегрувати моделі машинного навчання в існуючі програми [17].

Таким чином, TensorFlow Lite – це потужна програмна бібліотека, яка дозволяє ефективно розгортати моделі машинного навчання на пристроях з обмеженими ресурсами. Її оптимізації, такі як апаратне прискорення, квантування моделі та злиття ядер, дозволяють швидко та ефективно

використовувати пам'ять, що робить її цінним інструментом для додатків машинного навчання на пристроях таких яка Raspberry Pi.

Для того щоб встановити програмне забезпечення необхідно виконати наступні дії:

- оновити пакети Raspberry Pi OS;
- скачати репозиторій програми TensorFlow Lite Object Detection;
- встановити TensorFlow і OpenCV;
- налаштувати модель виявлення TensorFlow Lite.

4.3.1 Встановлення необхідних компонентів

Для роботи з платою Raspberry Pi було вирішено використовувати термінал, у якому попередньо налагоджено SSH-з'єднання, описане в підрозділі 4.2. Для оновлення програмного забезпечення та системних компонентів треба застосовати команду в командному рядку, спрямовану на отримання останніх оновлень.

```
$> sudo apt-get update
$> sudo apt-get dist-upgrade
```

Далі треба виконати встановлення допоміжних програм: GIT, призначений для контролю версій, CMAKE - інструмент для збирання проєктів, а також програми для архівації та WGET. Для цього було використано таку команду, зазначену в джерелі [18]:

```
$> sudo apt install -y cmake wget unzip
```

4.3.2 Завантаження додатку TensorFlow Lite Object Detection

Для інсталяції програми TensorFlow Lite Object Detection на операційну систему Raspberry Pi OS потрібне завантаження її початкового коду з

репозиторію на платформі GitHub. Цей процес здійснюється шляхом виконання певної команди в командному рядку:

```
$> git clone https://github.com/EdjeElectronics/TensorFlow-Lite-Object-Detection-on-Android-and-Raspberry-Pi.git
```

Після завершення завантаження даних, згідно з інструкцією [19], було створено директорію під назвою tflite1. Цю директорію потім буде переміщено до вказаного каталогу, деталі якого представлено на рисунку 4.5.

Рисунок 4.5 – створення і переміщення до папки проекта з GIT hub

Далі була виконана процедура створення віртуального середовища для програми. Цей крок зроблено з метою запобігання можливих майбутніх конфліктів між версіями бібліотечних пакетів, встановлених на Raspberry Pi. Особливу увагу приділено встановленню TensorFlow у дане середовище, що забезпечує ізоляцію і запобігання конфліктам версій.

```
$> sudo pip3 install virtualenv
$> python3 -m venv tflite1-env
```

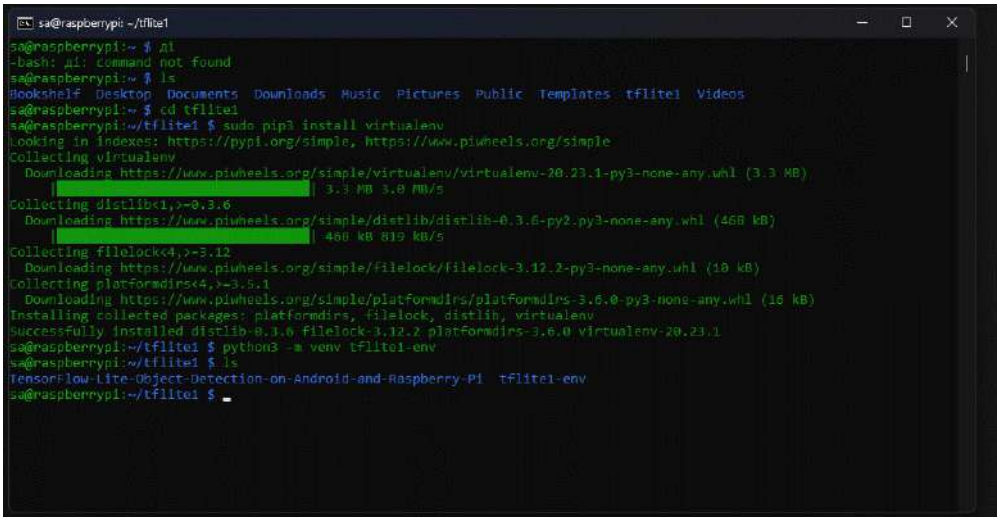


Рисунок 4.7 – Результат виконання команд створення середовища

Результатом виконання команд є створенням папки з ім'ям tflite1-env всередині каталогу tflite1, рисунок 4.7. У папці tflite1-env зберігатимуться всі бібліотеки пакетів для цього середовища. Активація середовища, виконується за допомогою команди:

```
$> source tflite1-env/bin/activate
```

Наступним кроком було здійснено встановлення залежностей TensorFlow Lite і OpenCV. Для цього використовувався скрипт get_pi_requirements.sh, запуск якого виконувався шляхом виконання відповідної команди:

```
$> bash get_pi_requirements.sh
```

4.3.3 Налаштування моделі виявлення TensorFlow Lite

З моделлю виявлення пов'язані два файли: файл detect.tflite (який є самою моделлю) і файл labelmap.txt (який надає карту міток для моделі).

У дипломній роботі магістра буде використано моделі виявлення від Google. Google надає зразок квантованої моделі виявлення об’єктів SSDLite-MobileNet-v2, яка навчена на наборі даних MSCOCO і перетворена для роботи в TensorFlow Lite. Він може виявляти та ідентифікувати 80 різних звичайних об’єктів, таких як люди, автомобілі, чашки тощо. Завантажимо зразок моделі, використовуючи команду:

```
$> bash  https://storage.googleapis.com/
download.tensorflow.org/models/tflite/coco_ssd_mobilenet_v1_
1.0_quant_2018_06_29.zip
```

Після завантаження розархівуємо її в робочу папку наступною командою:

```
$> unzip coco_ssd_mobilenet_v1_1.0_quant_2018_06_29.zip -d
google model
```

Запустити сценарій виявлення веб-камери в реальному часі, можна ввівши таку команду з каталогу /home/pi/tflite1, включивши при цьому папку з моделлю навчання google_model

```
$> python3 TFLite_detection_webcam.py --modeldir=goole_model
```

5 РОЗРОБКА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

У цьому розділі описуються процес розробки програмного забезпечення, за допомогою якого користувач зможе налаштувати бажану поведінку підлогомийних роботів. Користувач задає графік роботи та територію прибирання для кожного робота окремо.

5.1 Встановлення інструментів для розробки ПЗ

Автономна система прибирання в лікарні базується на одному або декількох роботах. Для налаштування та коригування виконання їхньої автономної роботи буде використовується розроблене програмне забезпечення, яке компілюється на Windows і Linux та побудоване за моделлю архітектури клієнт-сервер. Серверний компонент виступає як центральний вузол для налаштування поведінки парком роботів-прибиральників. Налаштування поведінки роботів має інтуїтивно зрозумілий користувацький інтерфейс для адміністраторів та операторів лікарні для взаємодії з системою.

Для написання програмного забезпечення використовувався кросплатформенний C++ фреймворк wxWidgets.

wxWidgets - це популярний C++ фреймворк з відкритим вихідним кодом, призначений для розробки крос-платформних графічних інтерфейсів користувача (GUI). Спочатку відомий як wxWindows, він був створений у 1992 р. Джуліаном Смартом та Робертом Роблінгом [20]. За допомогою wxWidgets розробники можуть написати код один раз і компілювати його на різних операційних системах, включаючи Windows, macOS, Linux та Unix-подібні системи.

Однією з ключових переваг wxWidgets є його здатність забезпечувати нативний вигляд на кожній підтримуваній платформі. Це означає, що додатки, створені з використанням wxWidgets, легко поєднуються з базовою

операційною системою, забезпечуючи послідовний користувацький досвід. Використовуючи нативні елементи керування, такі як кнопки, меню та діалогові вікна, фреймворк гарантує, що елементи графічного інтерфейсу виглядатимуть так, як очікується на кожній платформі.

Фреймворк дотримується моделі програмування, керованої подіями. Взаємодії з графічним інтерфейсом, такі як натискання кнопок або вибір меню, генерують події, які можуть бути оброблені програмою. Розробники можуть визначити обробники подій, які реагуватимуть на ці події, оновлюючи користувацький інтерфейс за потреби.

wxWidgets пропонує велику бібліотеку віджетів, що містить широкий спектр компонентів інтерфейсу, таких як кнопки, текстові елементи керування, списки, дерева та сітки. Ці віджети можна компонувати і налаштовувати для створення складних і багатофункціональних інтерфейсів.

Фреймворк також підтримує інтернаціоналізацію та локалізацію, дозволяючи розробникам створювати додатки, які можна перекладати різними мовами та адаптувати до різних місцевостей.

Завдяки вичерпній документації, навчальним посібникам, зразкам та активній спільноті, wxWidgets надає широкі можливості для розробників. Його розширюваність дозволяє створювати кастомні віджети та елементи керування для задоволення специфічних вимог додатків.

Для розробки серверної та клієнтської частини буде використовуватись бібліотека Boost.

Бібліотека Boost - це набір бібліотек C++, які забезпечують підтримку різних завдань, таких як лінійна алгебра, обробка зображень та багатопотоковість. Ці бібліотеки відомі своєю високою якістю, ефективністю та сумісністю зі стандартною бібліотекою шаблонів (STL).

Серед бібліотек Boost, BOOST.ASIO має особливе значення для розроблювальної системи, оскільки BOOST.ASIO пропонує масштабовані та ефективні асинхронні операції вводу/виводу. Асинхронний ввід/вивід є критично важливим при розробці високопродуктивних мережевих додатків,

які можуть обробляти численні одночасні з'єднання, не витрачаючи ресурси на холості потоки. BOOST.ASIO надає як високорівневий об'єктно-орієнтований інтерфейс, так і низькорівневий доступ до сервісів вводу/виводу, що дозволяє програмістам писати мережеві додатки з різним рівнем абстракції.

BOOST.ASIO підтримує різні об'єкти вводу/виводу, такі як сокети, таймери та послідовні порти, що робить його універсальним вибором для широкого спектру додатків. Однією з його основних переваг є можливість написання незалежного від платформи коду, який може працювати на різних операційних системах без модифікацій. Для встановлення цих бібліотек знадобиться компілятор C++ [21].

5.1.1 Встановлення компілятору C++ MinGW-w64

Для встановлення компілятора MinGW на систему Windows було виконано наступний алгоритм дій: спочатку виконано завантаження інсталятора MinGW з веб-сайту https://winlibs.com/, з вибором відповідної версії GCC. Процес встановлення полягає в розпакуванні архіву в зазначену директорію, в даному випадку C:\Mingw64. Після завершення інсталяці, для коректної роботи компілятора, необхідно налаштувати змінні оточення. Це робиться через меню "Пуск": клікнути правою кнопкою миші на "Комп'ютер" або "Цей ПК", вибрати "Властивості", а потім у вікні "Властивості системи" перейти за посиланням "Додаткові параметри системи". У діалоговому вікні "Властивості системи" слід натиснути кнопку "Змінні середовища", як показано на малюнку 5.1, для додавання необхідних параметрів [22-24].

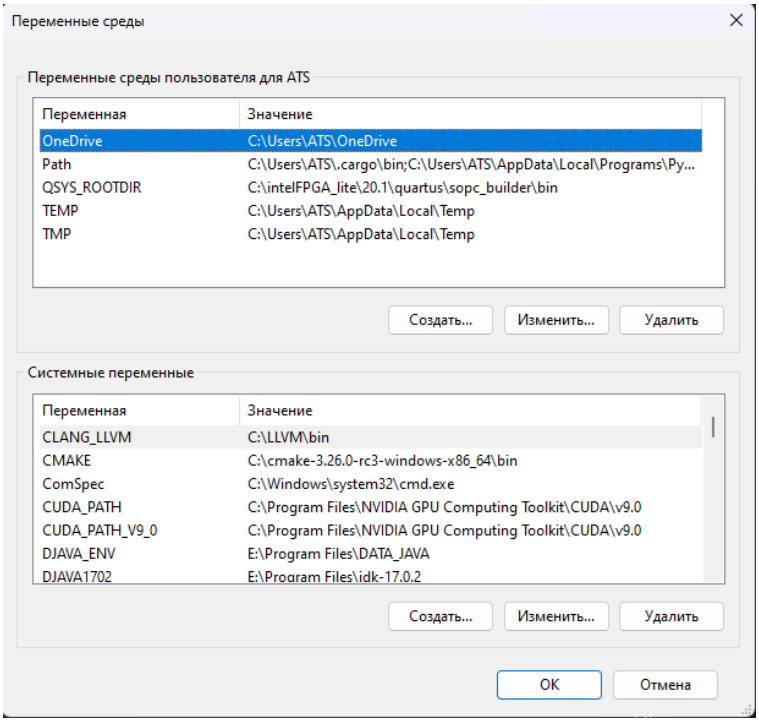


Рисунок 5.1 – Вигляд вікна налаштувань змінних середовища

Далі було проведено процес редагування системних змінних. У вікні "Змінні середовища", у розділі "Системні змінні", необхідно обрати змінну "Path" і натиснути кнопку "Змінити". До списку шляхів було додано шлях до каталогу MinGW bin, який під час розроблення було вказано як C:\Mingw64\bin. Після додавання шляху важливо натиснути "ОК", щоб зберегти внесені зміни. Для перевірки коректності виконаних дій рекомендується використовувати спеціальну команду в cmd.exe.

```
C:\> g++ -v
```

Для того що впевнитись, що все зроблено правильно, команда, в результаті виконання поверне версію компілятора c++, як показано на рисунку 5.2.

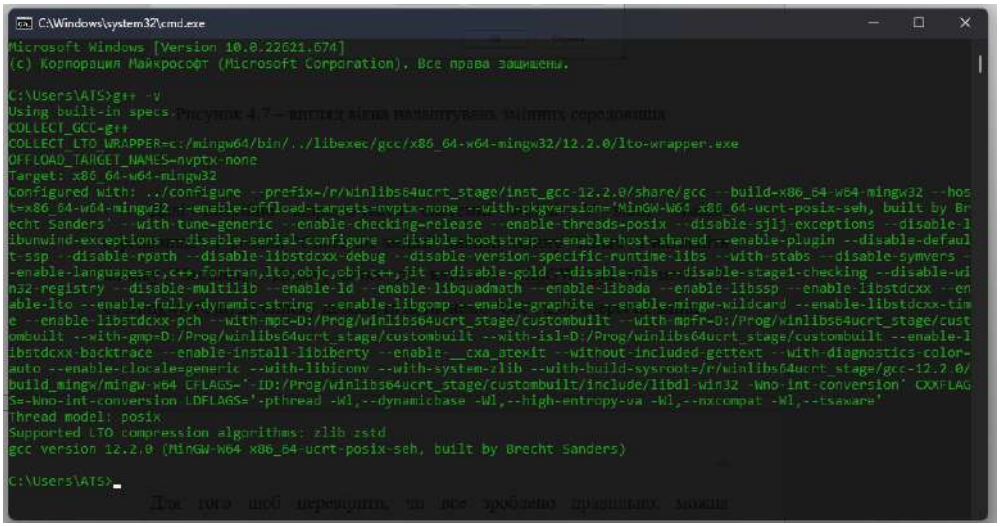


Рисунок 5.2 – Результат виконання команди `g++ -v`

5.1.2 Встановлення фреймворку wxWidgets

Щоб встановити фреймворк wxWidgets у Windows, потрібно завантажити початковий код фреймворку wxWidgets з офіційного сайту https://www.wxwidgets.org.

Після завершення процесу завантаження було проведено етап розпакування архіву в той каталог, який обрано для компіляції фреймворка. Потім було відкрито командний рядок у директорії, де розпаковано проект wxWidgets. Для переходу до відповідного каталогу збірки, призначеного для Windows, у командному рядку було виконано таку команду:

```
C:\> cd build\msw
```

Далі виконані команди, які виконають процес збирання відладочної версії wxWidgets за допомогою утиліти компілятора MinGW.

```
C:\> mingw32-make -j 8 SHARED=1 BUILD=debug setup_h
C:\> mingw32-make -j 8 SHARED=1 BUILD=debug
```

По завершенню збірки, було скопійовано вміст папки wxWidgets-3.2.2.1 lib\gcc_lib у заздалегідь створену папку проекту до папки залежностей (E:\P\TEST_xxWidgeth\Dependencies\xxWidgeth\lib\MinGW\Debug).

Також, скопійовано dll у каталог, де компілятор буде поміщати виконувані файли налагодження (E:\P\TEST_xxWidgeth\bin\Debug).

Після того, як бібліотеки були скопійовані, було виділено вихідну папку з зібраними бібліотеками і збирано знову, але цього разу для релізу:

```
C:\> mingw32-make -j 8 SHARED=0 RUNTIME_LIBS=static BUILD=release
```

Після успішного завершення процесу в робочу папку (TEST_xxWidgeth) проекту було скопійовано заголовки з папки збірки фремворку. Для того, щоб зібрати свій проект, у корені робочої папки проекту було створено make фали з назвою makefile.Win32Debug і makefile.Win32Release, дерево файлів проекту має такий вигляд рисунок 5.3.

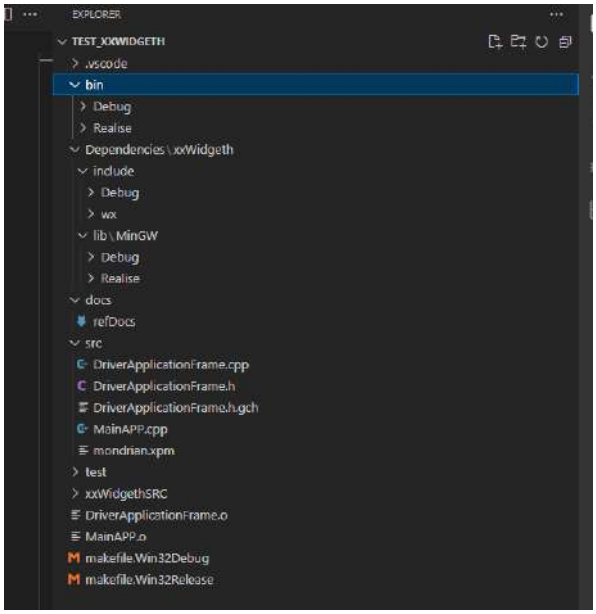


Рисунок 5.3 – Дерево файлів проекту налаштування поведінки роботів

У процесі написання make фалу необхідно вказати відповідні бібліотеки в збірціі, включно з /Dependencies/xxWidgeth/include/Debug/mswu (рисунок) 5.4. Запустити процес складання в режимі налагодження відбувається за допомогою такої команди:

```
C:\> make -j16 -f makefile.Win32Debug APP
```

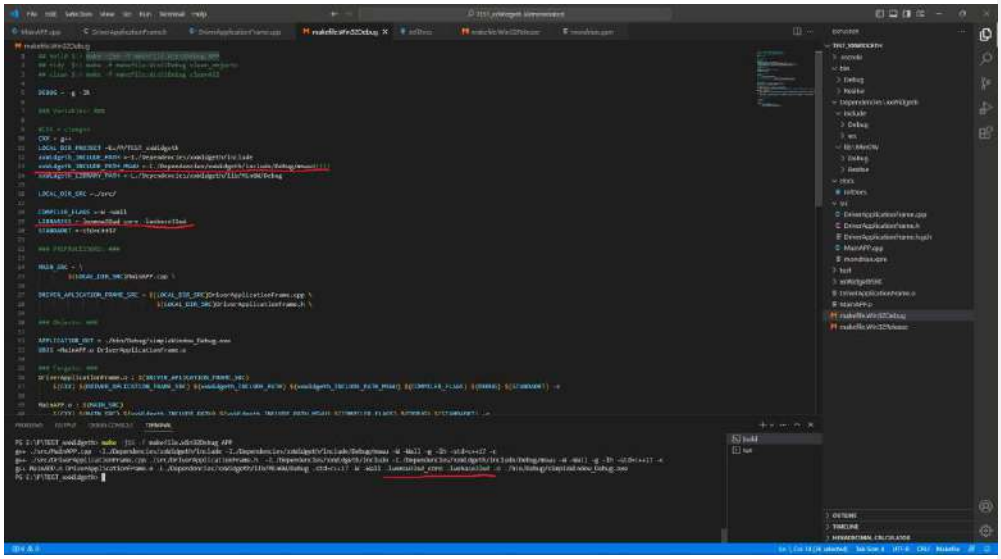


Рисунок 5.4 – Важливі компоненти підключення для збору проекту

5.1.3 Встановлення бібліотеки Boost для Window

Для отримання копії бібліотеки Boost було виконано перехід на офіційний веб-сайт https://www.boost.org/users/download/. У розділі Downloads було обрано останню стабільну версію релізу. Після переходу на відповідну вкладку релізу, було представлено варіанти завантаження архіву вихідного коду для платформ Unix і Windows, як показано на рисунку 5.5. Було завантажено архів boost_1_82_0.zip для подальшого використання.

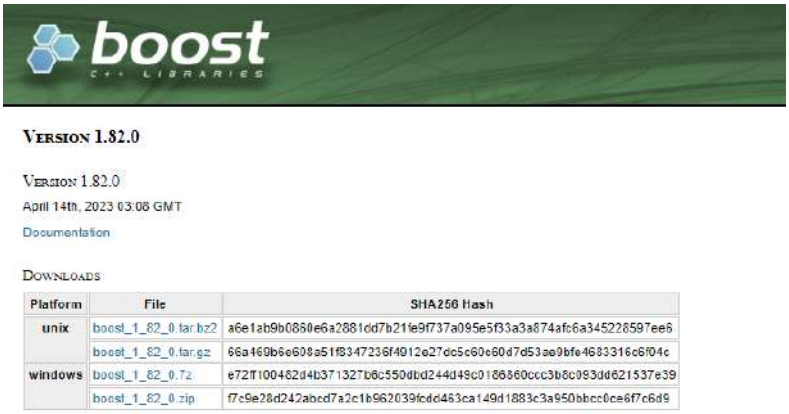


Рисунок 5.5 – Архіви початкового коду бібліотеки Boost

Після завершення процесу завантаження був проведений етап розпакування архіву, за яким послідувало відкриття папки boost_1_82_0 і запуск файлу bootstrap.bat. Виконання сценарію цього файлу призвело до складання програми b2.exe. Для налаштування збірки бібліотеки Boost під конкретний компілятор, необхідно було внести зміни до файлу project-config.jam, вказавши там назву використовуваного компілятора (наприклад, using gcc або using msvc).

Після вибору відповідного компілятора, було розпочато етап складання бібліотеки з використанням програми b2.exe. Процес роботи програми b2.exe і його результати продемонстровані на рисунку 5.6.

5.1.3 Встановлення бібліотеки Boost для Raspberry Pi OS Lite

Для того щоб розробити клієнтську частину програми для робота, було вирішено встановити бібліотеку Boost на Raspberry Pi OS Lite. Оскільки у підрозділі 4.2 було налаштовано віддалене з’єднання через SSH, програму можна зібрати для робота та про тестувати у Visual Studio Code.

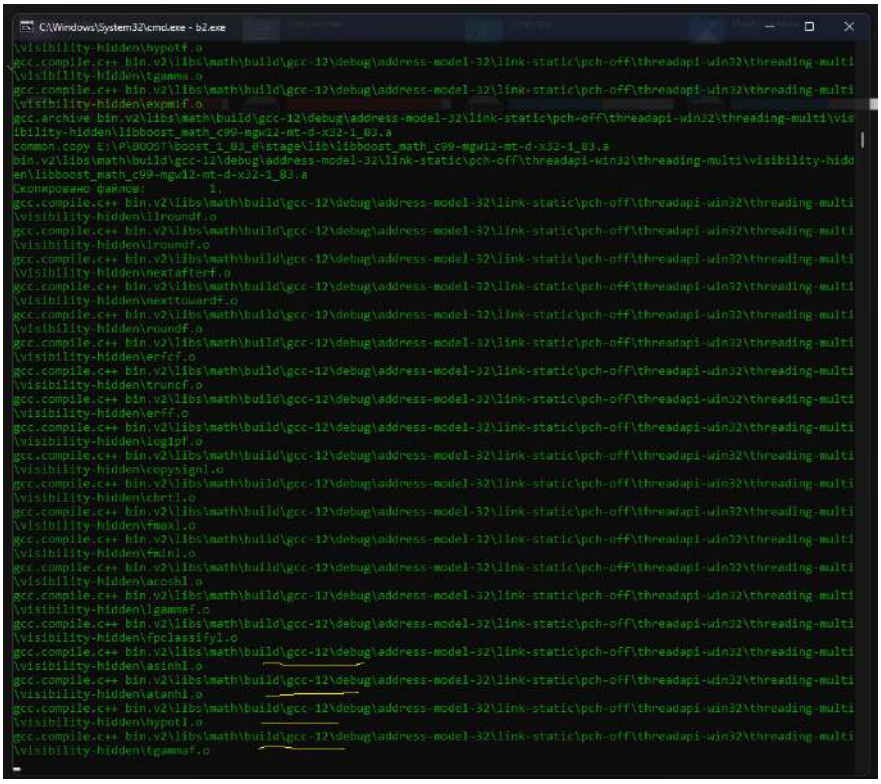


Рисунок 5.6 – Процес збірки бібліотеки Boost для GCC

У терміналі, встановлюючи SSH-з'єднання з Raspberry Pi, було виконано процес завантаження аналогічного архіву вихідного коду бібліотеки Boost, який було завантажено раніше. Для цього використовувалася така команда:

```
$>                                                    wget
https://boostorg.jfrog.io/artifactory/main/release/1.82.0/source
/boost_1_82_0.zip
```

Після завантаження розпакувавши архів було здійснено перехід у папку boost_1_82_0 та запущено сценарій bootstrap.sh, наступними командами:

```
$> unzip boost_1_82_0.zip
```

```
$> cd boost_1_82_0
$> sudo ./bootstrap.sh
```

Результатом виконання команд особливо останньої буде створення виконавчого файлу b2, генеровані файли можна побачити на рисунку 5.7.

Рисунок 5.7 – Результат роботи сценарію bootstrap.sh

Коли сценарій bootstrap.sh відпрацює без помилок, з’явиться програма b2, яку потрібно запустити, щоб почати компілювати бібліотеку, це можна зробити наступною командою:

```
$> sudo ./b2
```

5.2 Розробка програми Drone behavior administrator

Drone behavior administrator призначена для управління поведінкою роботів у режимі реального часу з використанням графічного інтерфейсу. Програма має надавати оператору можливість відповідати за розмітку зон прибирання на 2D мапі сканування поверху, яку передає робот, а також встановлення графіку роботи. Ця важлива інформація є основою для роботи розробленої системи прибирання в лікарні на основі підлогомийних роботів, встановлені дані керують маршрутами та завданнями очищення роботів.

5.2.1 Графічний інтерфейс

Графічний інтерфейс реалізовано за допомогою бібліотеки wxwidgets. Це дало можливість створити інтуїтивно зрозумілий і простий у використанні інтерфейс. Головне вікно програми, представлене на рисунку

5.8, було поділено на 8 розділів: 3 інформаційних і 5 інтерактивних, за допомогою об’єкту wxStaticLine бібліотеки wxwidgets. Кнопки, слайдери та перемикачі реалізовані за допомогою класів wxStaticText, wxButton, wxTextCtrl, wxChoice wxListBox та wxCheckBox у конструкторі DriverApplicationFrame, який наслідує реалізацію класу wxFrame (рисунок 5.8).

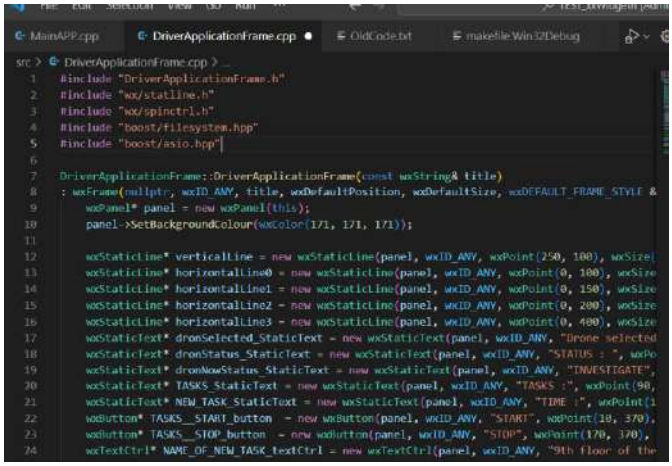


Рисунок 5.8 – Розміщення елементів інтерфейсу програми

Двомірна мапа малюється за допомогою класу wxPaintDC попіксельно на основі даних, які приймаються від робота з лідару.

Рисунок 5.8 – Графічний інтерфейс програми

5.2.2 Серверна частина

Drone behavior administrator запускає сервер, який реалізований за допомогою бібліотеки boost.asio. Кожен робот функціонує як клієнт, який може передавати дані про свій поточний стан і отримувати команди від сервера. Схему з'єднання представлено рисунку 5.9.

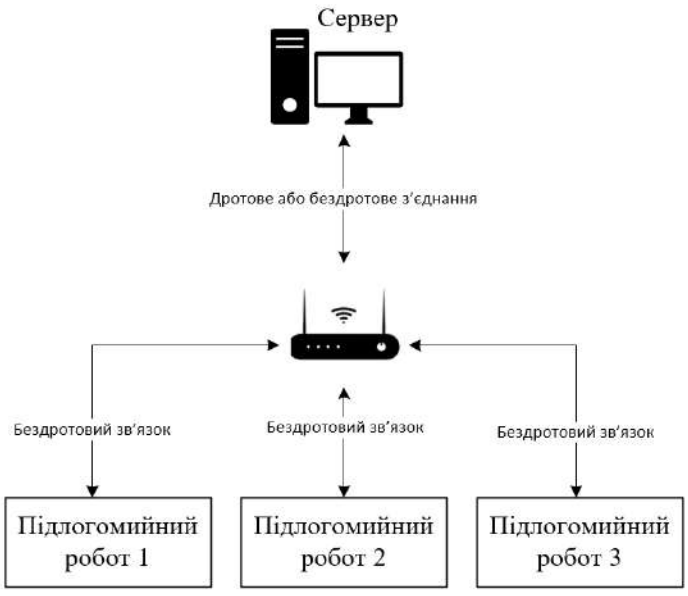


Рисунок 5.9 – Схема підключення роботів до серверу

Повний код представлено у додатку В. У представленому коді реалізовано асинхронний сервер із використанням популярної бібліотеки Boost.Asio, що слугує для взаємодії з різноманітними клієнтами, які у дипломному проекті є підлогомийний роботами.

Спочатку в коді встановлюється специфікація для сумісності з Windows, забезпечуючи коректну роботу на операційних системах починаючи з Windows 10. Це важливо для забезпечення універсальності коду та його коректної роботи в різних середовищах.

Далі, включається низка бібліотек. Boost.Asio надає комплексну функціональність для асинхронного програмування, що підходить для реалізації серверів, здатних обробляти безліч з'єднань одночасно без блокування основного потоку виконання. boost::date_time використовується для отримання поточної дати в зручному форматі.

У коді також визначено глобальну змінну session_id_generator, яка служить для створення унікальних ідентифікаторів сесії для кожного під'єднаного клієнта або робота. Це ключовий елемент для відстеження та ідентифікації даних, отриманих від різних джерел.

Основний робочий клас Session являє собою абстракцію одного з'єднання з роботом. Усередині нього використовується асинхронне читання і запис даних, що дає змогу серверу ефективно обробляти безліч з'єднань без затримок. Уся отримана інформація логірується із зазначенням дати та унікального номера сесії, що забезпечує прозорість і можливість подальшого аналізу даних.

Клас Server призначений для управління вхідними з'єднаннями. При появі нового з'єднання створюється новий екземпляр Session, який бере на себе управління цим з'єднанням.

У основній функції main ініціалізується контекст виконання, відбувається створення екземпляра серверу, це починає обробляти вхідні з'єднання на порту 12345. Потім запускається головний цикл обробки подій, що обробляє всі асинхронні завдання та операції.

5.2.3 2D карта

На рисунку 5.8 у відміченій 8 секції малюється мапа за допомогою класу wxPaintDC, який надає змогу малювати піксель, код представлений у додатку А DriverApplicationFrame.h функція void render(wxDC& dc) малює точки відстані, які надав робот після проходження поверху або певної зони, він передає інформацію на сервер один раз. Сервер зі свого боку відображає цю інформацію на 2D мапі в графічному інтерфейсі. Ця карта дає змогу користувачеві бачити пройдені та непройдені зони, а також встановлювати зони прибирання для кожного робота індивідуально.

5.3 Тестування роботи системи

5.3.1 Тестування серверу

У цьому підрозділі описано результати тестування сервера, створеного з використанням бібліотеки boost.asio. Метою тестування було перевірити функціональність сервера, а також його здатність приймати та обробляти повідомлення від клієнтів.

Сервер був скомпільований і запущений на операційній системі Windows. В якості клієнтів використовувалися машина з операційною системою Linux, яка були розгорнута на віртуальній машині VMware [25]. Важливо зазначити, що налаштування інтернет-адаптера для віртуальних машини було виконано в режимі "bridget", що дало змогу клієнтам і серверу взаємодіяти в одній локальній мережі. Результат можна побачити на рисунку 5.10.

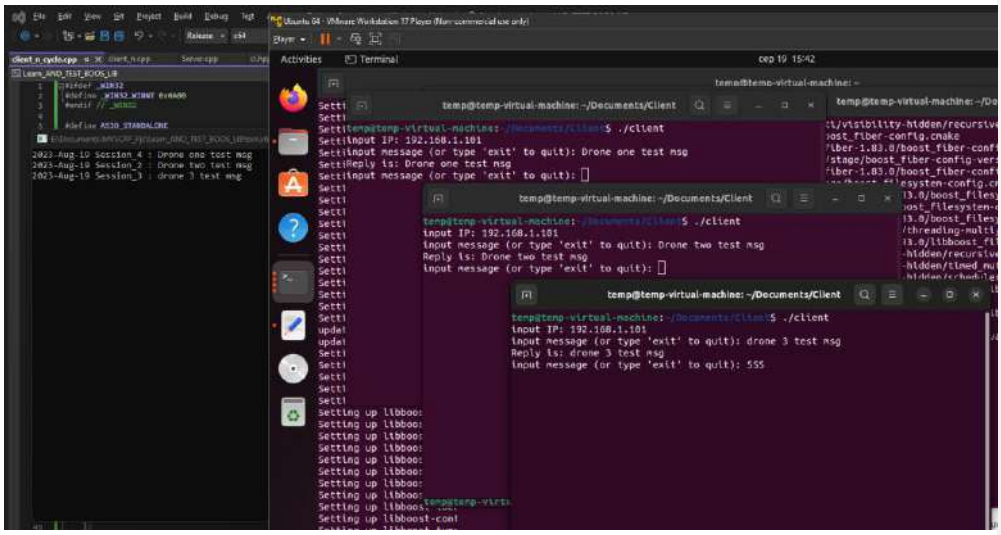


Рисунок 5.10 – Результат тестування роботи серверу та клієнтів

У процесі тестування було запущено три клієнти. Кожен з них успішно підключився до сервера і відправив повідомлення. У відповідь сервер

повторив прийняте повідомлення і відправив його назад клієнту, що було підтверджено на боці клієнтського додатка. Крім того, на консолі сервера відображалися всі прийняті повідомлення разом з ідентифікатором сесії, що дає змогу відстежувати активність кожного підключеного клієнта.

Код сервера заснований на асинхронних операціях, що робить його продуктивним і здатним обробляти безліч одночасних підключень. Кожен новий клієнт, який підключився до сервера, створює нову сесію, яка ідентифікується унікальним номером. У процесі взаємодії з клієнтом сервер спочатку асинхронно зчитує дані, що надійшли від клієнта, а потім асинхронно відправляє їх назад.

Код клієнта дає змогу користувачеві вводити IP-адресу сервера для підключення та надсилати повідомлення на сервер. Після надсилання кожного повідомлення клієнт також отримує відповідь від сервера і відображає її.

Проведені тести показали стабільну роботу сервера. Усі три клієнти успішно під'єдналися і взаємодіяли із сервером, не було помічено помилок або збоїв у роботі серверного застосунку. Таким чином, можна зробити висновок про коректну роботу реалізованого сервера і його готовність до подальшого використання і інтегрування до системи керування роботами, а клієнтську частину – до робота.

5.3.2 Тестування TensorFlow Lite Object Detection Models

Цей підроділ присвячено тестуванню моделі, яку було встановлено у розділі 4.3 де описано процес налаштування моделі виявлення об'єктів на базі TensorFlow Lite.

Модель, заснована на TensorFlow Lite, здатна виявляти та ідентифікувати до 80 різних типів об'єктів. Цей великий спектр об'єктів робить модель потенційно універсальною для різноманітних завдань.

Метою розробки підлогомийного робота є його функціонування в умовах лікарні. З цією метою проводилося тестування моделі на зображенні

інтер'єру лікарні. Вихідне зображення (рисунок 5.11), містило елементи інтер'єру та людину, що дало змогу оцінити ефективність і точність детектування та класифікації об'єктів в умовах лікарні.

Рисунок 5.11 – Вихідне зображення TensorFlow Lite Object Detection

За результатами обробки цього зображення, модель успішно ідентифікувала чотирьох людей, стілець і ліжко. Це свідчить про здатність моделі коректно інтерпретувати об'єкти в контексті лікарняного інтер'єру.

```
class, confidence, bx, by, bw, bh
person, 1.00, 0.41, 0.50, 0.03, 0.12
person, 0.99, 0.40, 0.49, 0.02, 0.11
person, 1.00, 0.39, 0.47, 0.02, 0.11
person, 0.97, 0.10, 0.48, 0.07, 0.13
boat, 0.33, 0.72, 0.75, 0.59, 0.54
chair, 0.51, 0.52, 0.57, 0.09, 0.07
bed, 0.34, 0.72, 0.79, 0.52, 0.39
```

Рисунок 5.11 – Результати обробки зображення

ВИСНОВКИ

В дипломній роботі магістра автоматизовано процес прибирання приміщень у лікарні шляхом інтеграції розробленої системи управління до підлогомийних машин. При цьому було розглянуто і проаналізовано підходи щодо створення схожих роботизованих систем прибирання та їх апаратної частини.

Розроблено систему управління, що дозволяє себе інтегрувати до багатьох систем підлогомийних машини, після чого такі машини отримують змогу автоматичному режимі виконувати прибирання виділеної території, які заздалегідь вказані оператором. Система адаптується під навколишнє середовище й відповідно реагує на зміну перешкод на встановленому шляху.

Проведено тестування Серверної частини й програмного забезпечення TensorFlow Lite Object Detection Models.

При виконанні дипломної роботи магістра була розроблена система, що об'єднує загальнодоступні модулі в єдину комплексну систему, що забезпечує автономну систему прибирання в лікарні. Завдяки цьому обслуговування та заміна комплектуючих роботу проводиться з мінімальними зусиллями.

Наукова новизна дипломної роботи полягає у розробці системи управління, яка здатна розширити можливості звичайних підлогомийних машин, перетворюючи їх на роботизовану систему прибирання. Ця система керується за допомогою розробленого програмного забезпечення, що дає змогу вибрати територію в лікарні та встановити графік роботи.

Практична цінність розробленої системи управління для підлогомийних машин є у можливості розвантаження персоналу лікарень. Враховуючи високі вимоги до чистоти в медичних закладах, система може значно покращити процес прибирання, зменшуючи фізичне навантаження та ризики для здоров'я працівників. Зокрема, використання роботизованих систем прибирання в лікарнях може бути ефективним у боротьбі з

поширенням інфекційних захворювань. Завдяки автоматизації процесу прибирання знижується ризик контакту персоналу з патогенами, особливо у випадках, коли пацієнти хворі на заразні захворювання.

Таким чином, розроблена система не лише сприяє підвищенню ефективності прибирання, але й може відігравати важливу роль у підтримці безпечного середовища в лікарнях та зниженні рівня захворюваності.

Дипломна робота магістра пройшла апробацію на Всеукраїнській науково-практичної конференції молодих учених "Телекомунікації, автоматизація, комп’ютерно-інтегровані та інформаційні технології"[26].

Таким чином, отримано продукт, що має низьку вартість й високий потенціал для майбутнього розвитку.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. J A. Dynamics and control of robotic systems : підручник. Hoboken, New Jersey, USA : John Wiley & Sons, Inc., 2020. 516 p. URL: https://nibmehub.com/opac-service/pdf/read/Dynamics%20and%20Control%20of%20Robotic%20Systems.pdf (date of access: 10.08.2023).

2. Kurdila A. J., Ben-Tzvi P. Dynamics and control of robotic systems. Wiley & Sons, Limited, John, 2019. 520 p.

3. Association I. R. M. Robotic systems: concepts, methodologies, tools, and applications : підручник. IGI Global, 2020.

4. McKerrow P. J. Introduction to robotics (electronic systems engineering series) : підручник. Addison-Wesley, 1991. 800 p.

5. RA660 navi XL – the future of autonomous industrial cleaning. Cleanfix RA660 Navi XL. URL: https://www.ra660navi.com/ (date of access: 12.10.2023).

6. Scrubber drier KIRA B 50. Kärcher International. URL: https://www.kaercher.com/int/professional/floor-scrubbers-scrubber-driers/autonomous-scrubber-driers/kira-b-50-15330010.html (date of access: 12.10.2023).

7. UV guardian. Richtech AI. URL: https://www.richtech-ai.com/uv-guardian (date of access: 12.10.2023).

8. BeagleBone black rev C. MiniBoard.com.ua. URL: https://miniboard.com.ua/boards/415-beaglebone-black-rev-c.html (date of access: 11.10.2023).

9. Waller J. S. Raspberry pi 3 model B for beginners: explore what raspberry pi 3 model B can do. Independently Published, 2019.

10. Clark L. What is the ASUS tinker board?. Practical tinker board. Berkeley, CA, 2018. P. 3–11. URL: https://doi.org/10.1007/978-1-4842-3826-4_1 (date of access: 13.09.2023).

11. TF-Luna lidar range sensor. Waveshare Wiki. URL: https://www.waveshare.com/wiki/TF-Luna_LiDAR_Range_Sensor (date of access: 22.10.2023).

12. Camera. Raspberry Pi Documentation. URL: https://www.raspberrypi.com/documentation/accessories/camera.html (date of access: 19.10.2023).

13. Interface MPU6050 accelerometer and gyroscope sensor with arduino. last minute engineers. URL: https://lastminuteengineers.com/mpu6050-accel-gyro-arduino-tutorial/ (дата звернення: 24.10.2023).

14. Bell C. Introducing the raspberry pi pico. Beginning MicroPython with the Raspberry Pi Pico. Berkeley, CA, 2022. P. 1–42. URL: https://doi.org/10.1007/978-1-4842-8135-2_1 (date of access: 09.11.2023).

15. Getting started with your Raspberry Pi. Raspberry Pi. URL: https://www.raspberrypi.com/documentation/computers/getting-started.html (date of access: 25.10.2023).

16. Object detection using tensorflow lite / A. K. Kashyap et al. International journal of research publication and reviews. 2023. Vol. 4, no. 5. P. 3093–3097. URL: https://doi.org/10.55248/gengpi.4.523.40694 (date of access: 26.12.2023).

17. Situnayake Pete Warden D. O'Reilly.TinyML. Wykorzystanie TensorFlow Lite do uczenia maszynowego na Arduino i innych mikrokontrolerach : підручник. Helion, 2022. 432 p.

18. Huamán A. Installation in linux. OpenCV Documentation. URL: https://docs.opencv.org/4.x/d7/d9f/tutorial_linux_install.html (date of access: 27.10.2023).

19. Venv – Creation of virtual environments. Python documentation. URL: https://docs.python.org/3/library/venv.html (дата звернення: 29.10.2023).

20. Mukherjee A. Learning boost C++ libraries. Packt Publishing, Limited, 2015.

21. Anggoro W., Torjo J. Boost. asio C++ network programming - second edition. Packt Publishing, Limited, 2015.

22. C programming language. Prentice Hall, 1988. 274 p.

23. O'Dwyer A. Mastering the C++17 STL: Make full use of the standard library components in C++17. Packt Publishing, 2017. 384 p.

24. Palakollu S. M. Practical systems programming with C: pragmatic example applications in linux and unix-based operating systems. Apress L. P., 2020.

25. Zimmer D. VMware Server and VMware Player. The way forward for Virtualization. BoD, 2006. 416 p.

26. Архів – Всеукраїнський науково-практичний форум "ТАК". Всеукраїнський науково-практичний форум "ТАК". URL: https://tak.donntu.edu.ua/arxiv/ (дата звернення: 20.12.2023).

ДОДАТОК А
ПРОГРАМА DRONE BEHAVIOR ADMINISTRATOR

Лістинг А – MainAPP.cpp

```
class Main : public wxApp {
   public:
   virtual bool OnInit() override;
};

wxIMPLEMENT_APP(Main);
DECLARE_APP(Main)

bool Main::OnInit() {
   DriverApplicationFrame* MainDriverApplicationFrame = new DriverApplicationFrame("Hospital drone
behavior administrator [ Ver 0.01 betta ]");
   MainDriverApplicationFrame->SetClientSize(1080, 800);
   MainDriverApplicationFrame->Center();
   MainDriverApplicationFrame->Show(true);

   return true;
}
```

Лістинг А – DriverApplicationFrame.h

```
#ifndef DRIVER_APLICATION_FRAME
#define DRIVER_APLICATION_FRAME

#include "wx/wxprec.h"
#ifndef WX_PRECOMP
   #include <wx/wx.h>
#endif

#include "wx/frame.h"

class DriverApplicationFrame : public wxFrame {
public:
   DriverApplicationFrame(const wxString& title);
   void OnPaint(wxPaintEvent& evt);
   void render(wxDC& dc);
   void radDatafromSession()
   void AllEventGet();

private:
  void MY_OnButtonClicked(wxCommandEvent &evt);

private:
   wxPoint lineStart;
   wxPoint lineEnd;

};

#endif //DRIVER_APLICATION_FRAME
```

Лістинг А – AsinchServer.cpp

```
#define ASIO_STANDALONE

#include "boost/asio.hpp"
#include "boost/date_time/posix_time/posix_time.hpp"
#include <iostream>
#include <memory>
#include <chrono>
#include <iomanip>
#include <sstream>

using boost::asio::ip::tcp;

int session_id_generator = 0;

std::string getCurrentDate() {
    const boost::posix_time::ptime now = boost::posix_time::second_clock::local_time();
    return to_simple_string(now.date());
}

class Session : public std::enable_shared_from_this<Session> {
private:
    tcp::socket socket_;
    std::array<char, 128> buffer_;
    int session_Drone_id_;

    void read() {
        auto self(shared_from_this());
        socket_.async_read_some(boost::asio::buffer(buffer_),
            [this, self](boost::system::error_code ec, std::size_t length) {
                if (!ec) {
                    boost::asio::async_write(socket_, boost::asio::buffer(buffer_, length),
                        [this, self](boost::system::error_code ec, std::size_t /*length*/) {
                            if (!ec) {
                                read();
                            }
                        });

                    std::cout << getCurrentDate() << " Session_" << session_Drone_id_ << " : ";
                    for (int i = 0; i <= length - 1; i++) {
                        std::cout << buffer_[i];
                    }
                    std::cout << std::endl;
                }
            });
    }

public:
    Session(tcp::socket socket) : socket_(std::move(socket)), session_Drone_id_(++session_id_generator) {}

    void start() {
        read();
    }
};

class Server {
private:
    tcp::acceptor acceptor_;
```

```
    tcp::socket socket_;

    void accept() {
        acceptor_.async_accept(socket_,
            [this](boost::system::error_code ec) {
                if (!ec) {
                    std::make_shared<Session>(std::move(socket_))->start();
                }
                accept();
            });
    }

public:
    Server(boost::asio::io_context& io_context, short port)
        : acceptor_(io_context, tcp::endpoint(tcp::v4(), port)), socket_(io_context) {
        accept();
    }
};

int main() {
    try {
        boost::asio::io_context io_context;
        Server server(io_context, 12345);
        io_context.run();
    }
    catch (std::exception& e) {
        std::cerr << e.what() << std::endl;
    }

    return 0;
}
```

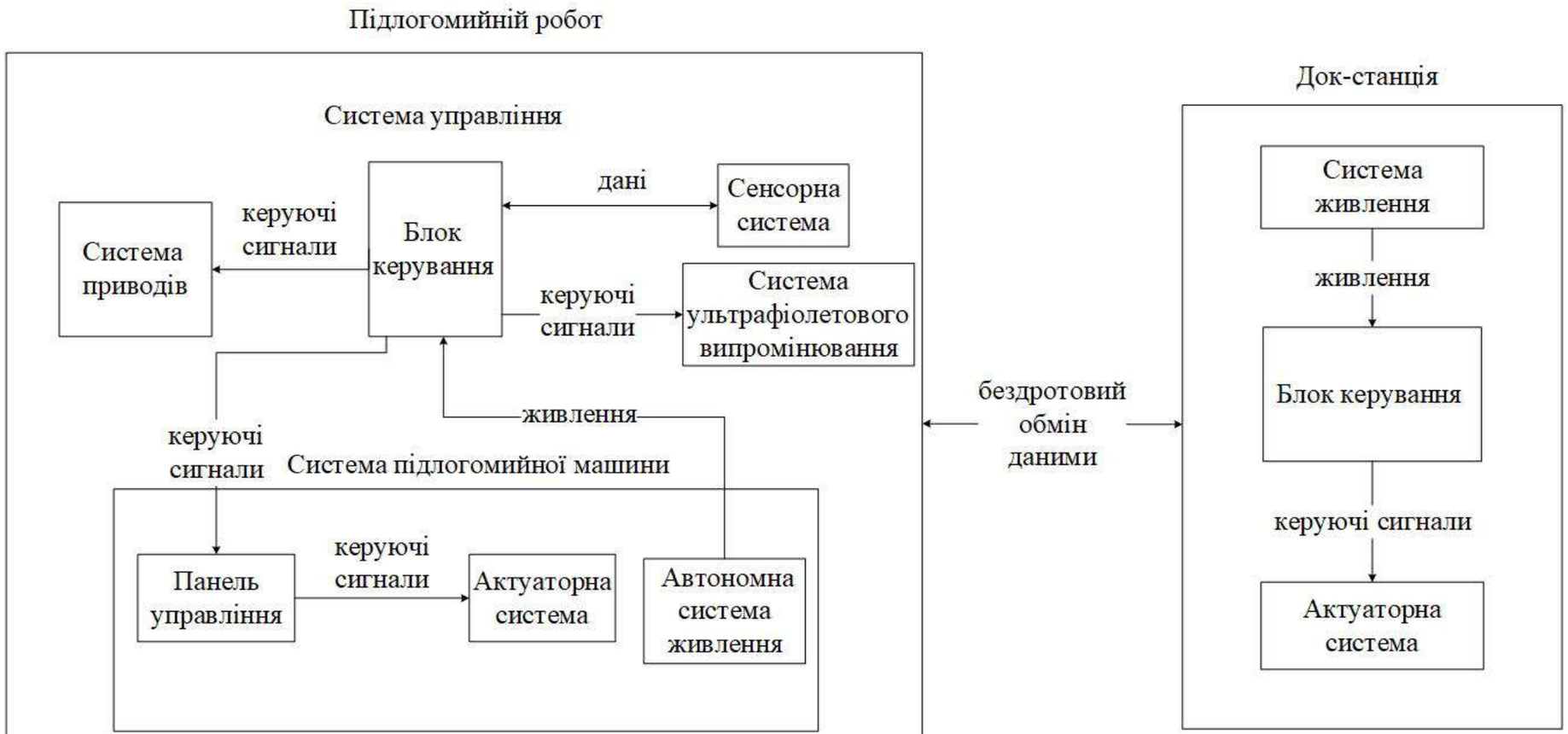
ДОДАТОК Б
ПРОГРАМА CLIENT ДЛЯ РОБОТІВ

Лістинг В – Client.cpp

```
#include "boost/asio.hpp"
#include <iostream>

using boost::asio::ip::tcp;

int main() {
   try {
      boost::asio::io_context io_context;

      std::string server_ip;
      std::cout << "input IP: ";
      std::cin >> server_ip;

      tcp::resolver resolver(io_context);
      auto endpoints = resolver.resolve(server_ip, "12345");
      tcp::socket socket(io_context);
      boost::asio::connect(socket, endpoints);

      std::cout << "input message: ";
      std::string request;
      std::cin.ignore(); // Ignore the newline character left in the stream after previous input
      std::getline(std::cin, request);
      size_t request_length = request.length();
      boost::asio::write(socket, boost::asio::buffer(request, request_length));

      char reply[128];
      size_t reply_length = boost::asio::read(socket, boost::asio::buffer(reply, request_length));
      std::cout << "Reply is: ";
      std::cout.write(reply, reply_length);
      std::cout << "\n";
   }
   catch (std::exception& e) {
      std::cerr << e.what() << "\n";
   }

   return 0;
}
```

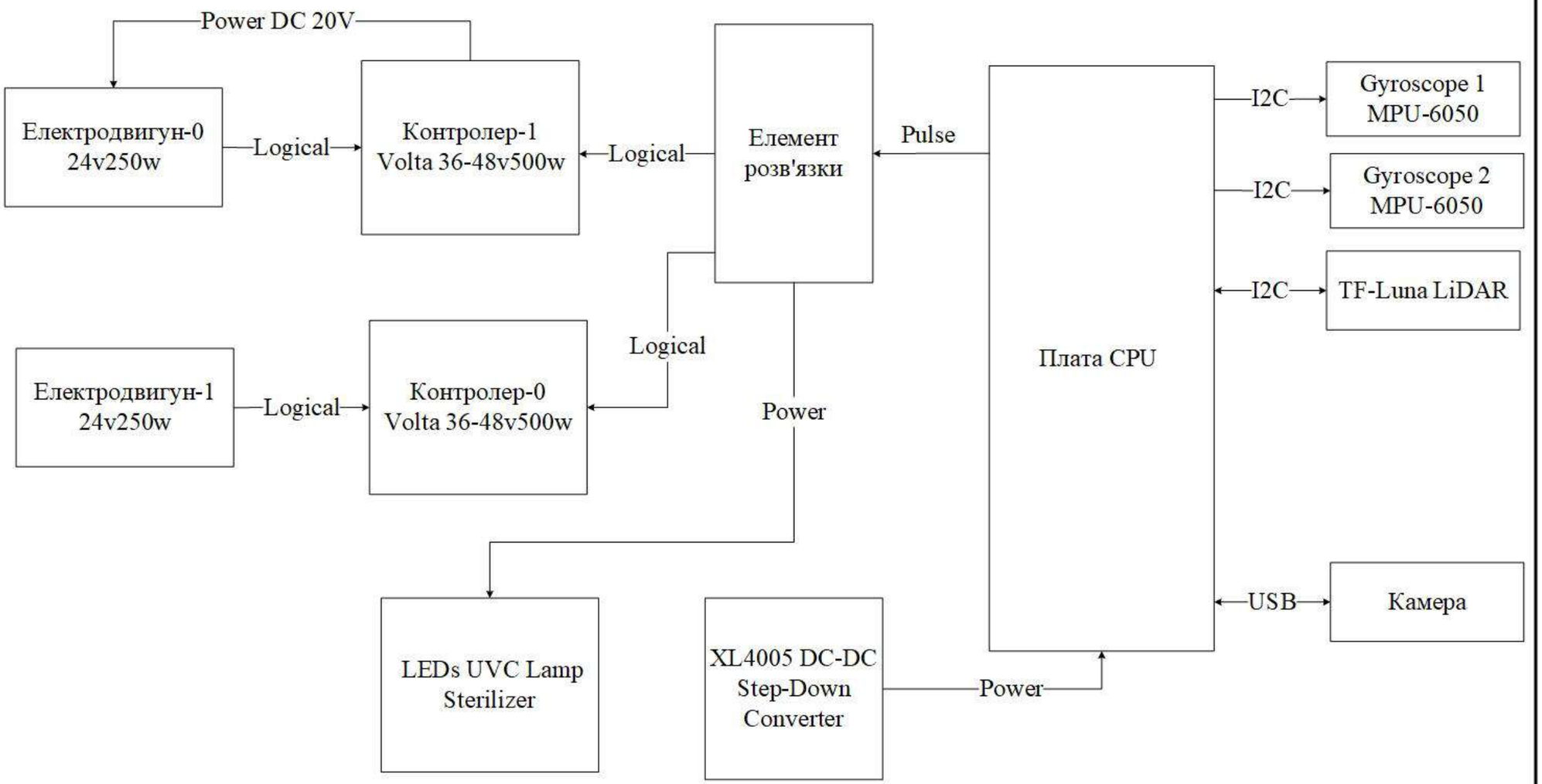
ДОДАТОК В

Загальна структурна схема роботизованої ситеми



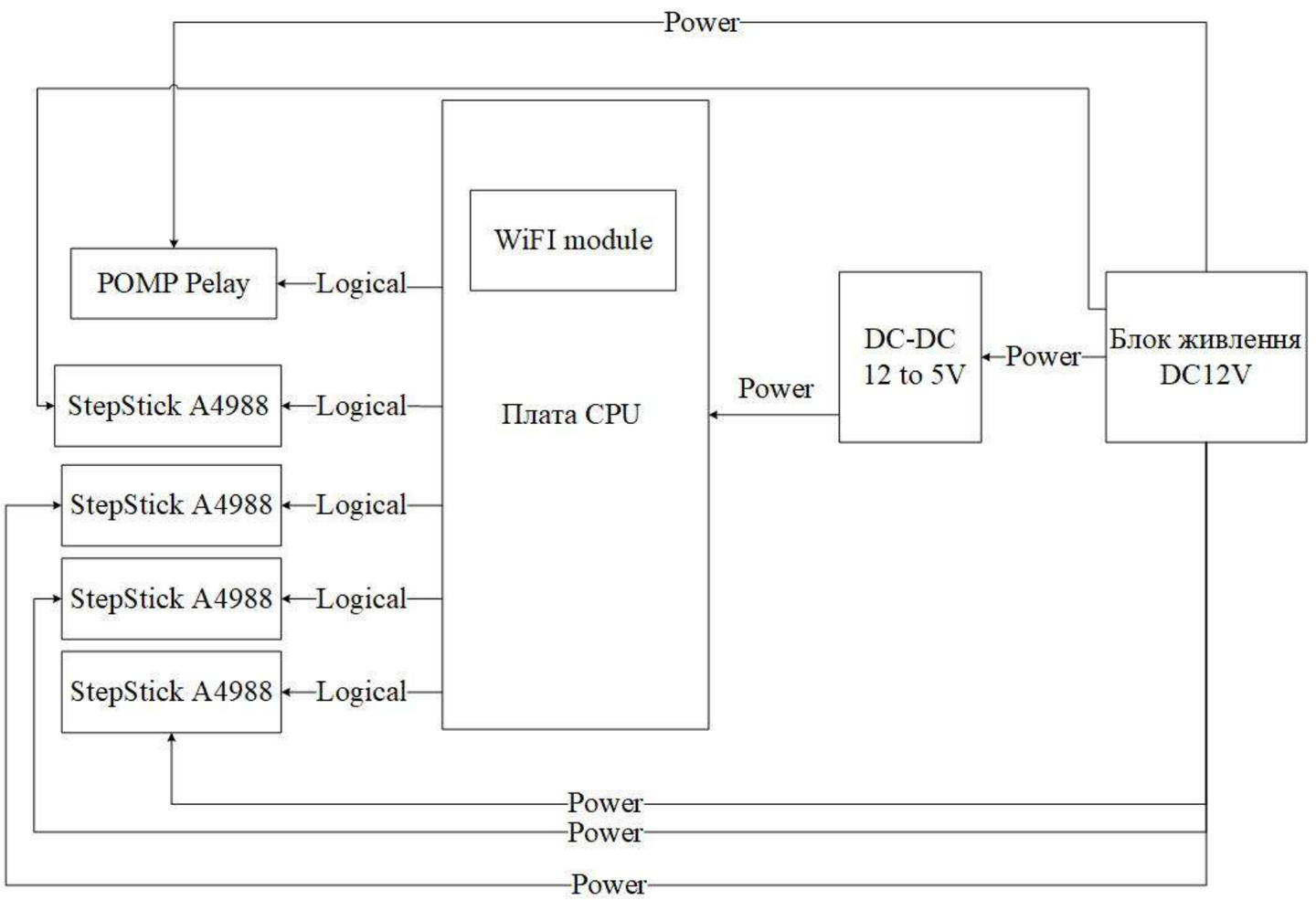
					13.02070849.00014 ПЛ1	
Зам.	Лист.	№ докум.	Підп.	Дата		Лит. / Маса / Масшт.
Розроб.		Зорін І.В.			Загальна структурна схема роботизованої системи	
Перев.		Фурманова Н.І.				Лист 1 / Листів 1
Т.контр.						
Н.контр.		Поспєєва І.Є.				НУ «Запорізька політехніка» БК-512м
Затв.		Малий О.Ю.				

Структурна схема системи управління



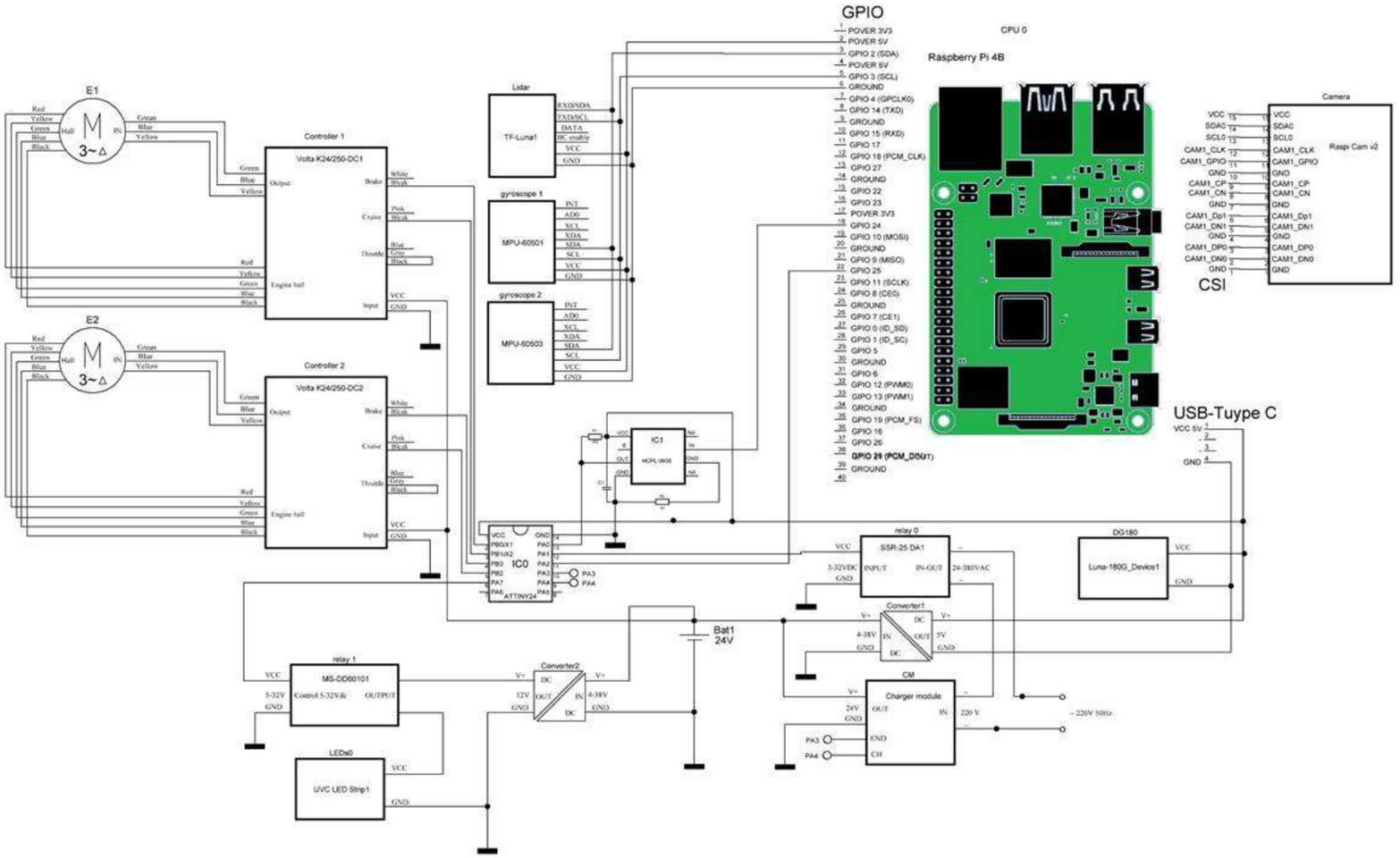
					13.02070849.00014 ПЛ2	
Зам.	Лист.	№ докум.	Підп.	Дата		
Розроб.		Зорін І.В.			Структурна схема системи управління	Літ. / Маса / Масшт.
Перев.		Фурманова Н.І.				Лист 2 / Листів 1
Т.контр.						
Н.контр.		Поспєєва І.Є.				НУ «Запорізька політехніка» БК-512м
Затв.		Малий О.Ю.				

Структурна схема док-станції



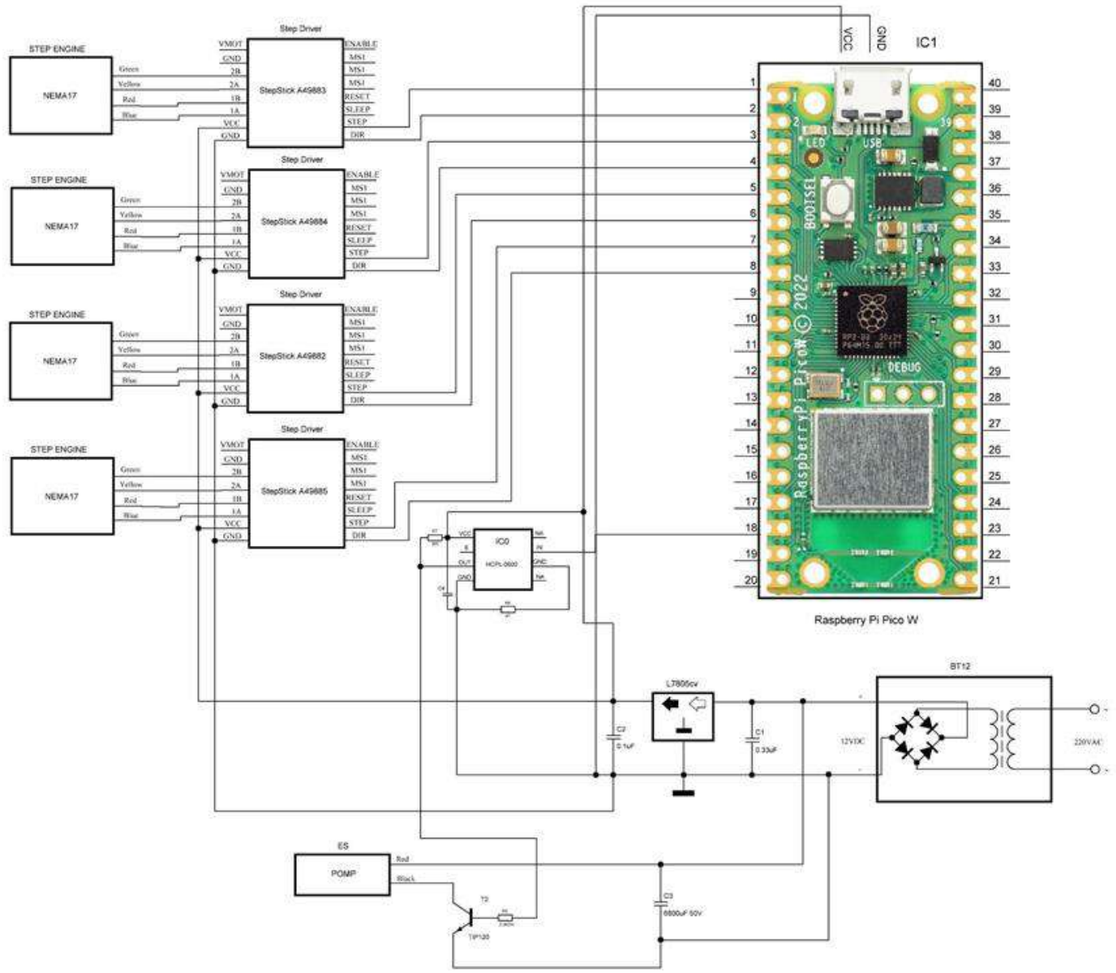
					13.02070849.00014 ПЛ3		
Зам.	Лист.	№ докум.	Підп.	Дата	Структурна схема док-станції	Лит.	Маса	Масшт.
Розроб.		Зорін І.В.						
Перев.		Фурманова Н.І.				Лист 3	Листів 1	
Т.контр.								
Н.контр.		Поспєєва І.Є.				НУ «Запорізька політехніка» БК-512м		
Затв.		Малий О.Ю.						

Електрична схема системи управління

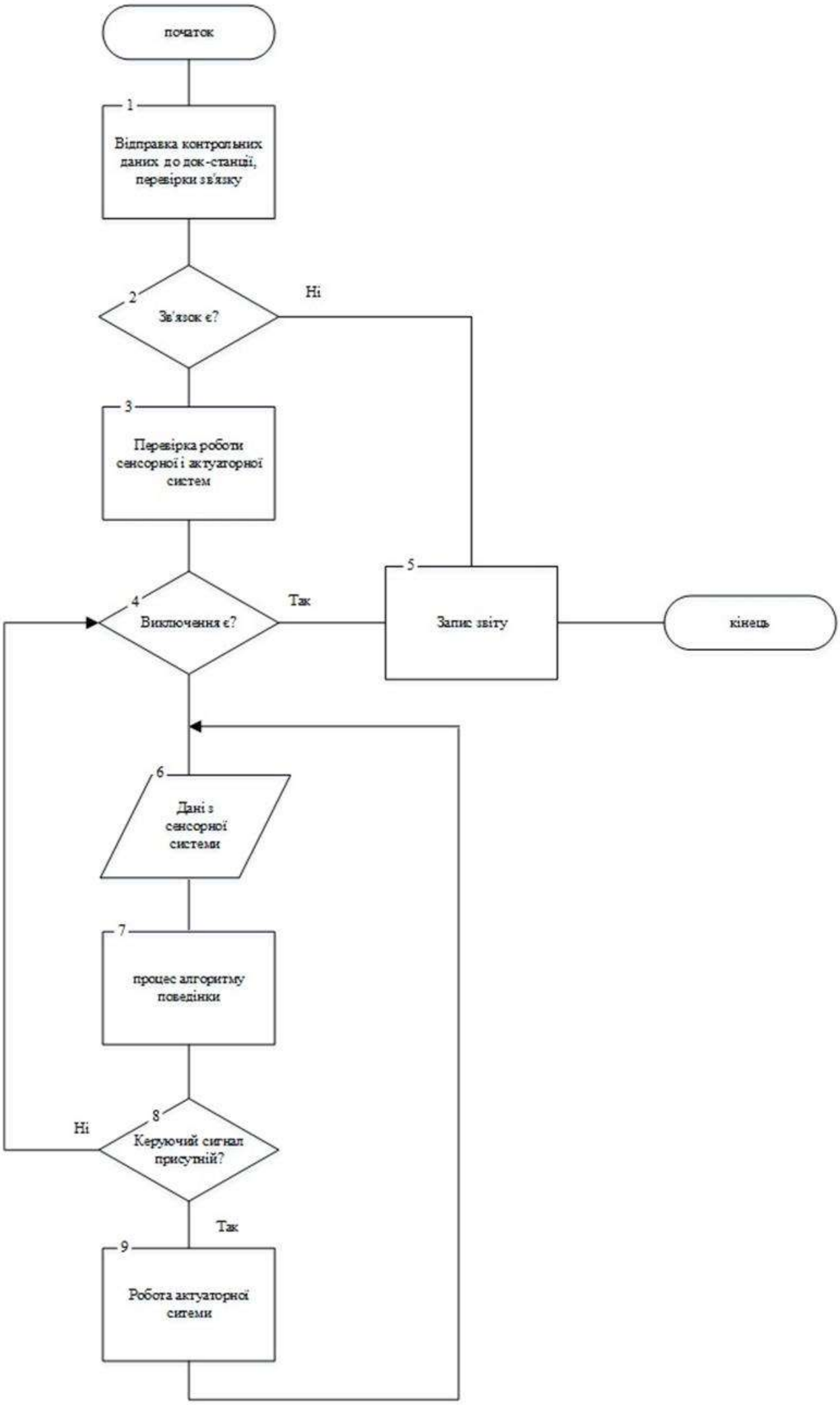


						13.02070849.00014 ПЛ4
Зам.	Лист.	№ докум.	Підп.	Дата	Електрична схема системи управління	Літ. / Маса / Масшт.
Розроб.		Зорін І.В.				
Перев.		Фурманова Н.І.				Лист 4 / Листів 1
Т.контр.						НУ «Запорізька політехніка» БК-512М
Н.контр.		Поспєєва І.Є.				
Затв.		Малий О.Ю.				

Електрична схема док-станції



						13.02070849.00014 ПЛ5
Зам.	Лист.	№ докум.	Підп.	Дата	Електрична схема док-станції	Літ. / Маса / Масшт.
Розроб.		Зорін І.В.				
Перев.		Фурманова Н.І.				Лист 3 / Листів 1
Т.контр.						
Н.контр.		Поспєєва І.Є.				НУ «Запорізька політехніка» БК-512М
Затв.		Малий О.Ю.				



					13.02070849.00014 А1	
Зам.	Лист.	№ докум.	Підп.	Дата		
Розроб.		Зорін І.В.			Загальний алгоритм функціювання підлогомийного робота	Лит. / Маса / Масшт.
Перев.		Фурманова Н.І.				Лист 6 / Листів 1
Т.контр.						
Н.контр.		Поспєєва І.Є.				НУ «Запорізька політехніка» БК-512м
Затв.		Малий О.Ю.				