

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»

Інститут інформатики та радіoeлектроніки,
Факультет комп'ютерних наук і технологій

(повне найменування інституту, назва факультету)

Кафедра програмних засобів

(повне найменування кафедри)

Пояснювальна записка

до дипломного проєкту (роботи)

магістр

(ступінь вищої освіти)

на тему ДОСЛІДЖЕННЯ ТА ПРОГРАМНА РЕАЛІЗАЦІЯ ПІДСИСТЕМИ
КОМПЛЕКСУ АВТОМАТИЧНОГО КОНТРОЛЮ ОСВІТЛЕННЯ У
ПРИМІЩЕННІ
RESEARCH AND SOFTWARE IMPLEMENTATION OF THE AUTOMATIC
CONTROL SUBSYSTEM FOR ROOM LIGHTING CONTROL

Виконав: студент(ка) 2 курсу, групи КНТ-120м
Спеціальності 121 Інженерія програмного
забезпечення

(код і найменування спеціальності)

Освітня програма (спеціалізація)

Інженерія програмного забезпечення

Зіненко Д.І.

(прізвище та ініціали)

Керівник Зеленьова І.Я.

(прізвище та ініціали)

Рецензент Терещенко Е.В.

(прізвище та ініціали)

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

Інститут, факультет ІРЕ, ФКНТ

Кафедра програмних засобів

Ступінь вищої освіти магістр

Спеціальність 121 Інженерія програмного забезпечення

(код і найменування)

Освітня програма (спеціалізація) Інженерія програмного забезпечення

(назва освітньої програми (спеціалізації))

ЗАТВЕРДЖУЮ

Завідувач кафедри ПЗ, д.т.н, проф.

С.О. Субботін

“ ” 20 року

З А В Д А Н Н Я

НА ДИПЛОМНИЙ ПРОЄКТ (РОБОТУ) СТУДЕНТА(КИ)

Зіненка Данила Ігоровича

(прізвище, ім'я, по батькові)

1. Тема проєкту (роботи) Дослідження та програмна реалізація підсистеми
комплексу автоматичного контролю освітлення у приміщенні

Research and Software Implementation of the Automatic Control Subsystem for
Room Lighting Control

керівник проєкту (роботи) Зеленьова Ірина Яківна, к.т.н., доцент

(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом закладу вищої освіти від “05” листопада 2021 року № 435

2. Строк подання студентом проєкту (роботи) 1 грудня 2021 року

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

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити) 1. Аналіз предметної області. 2. Розробка макету підсистеми
автоматичного контролю освітлення у приміщенні. 3. Розробка програмної
реалізації підсистеми. 4. Тестування підсистеми та керівництво користувача

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

6. Консультанти розділів проєкту (роботи)

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	прийняв виконане завдання
1-4 Основна частина	Зеленьова І.Я., к.т.н. доцент		
Нормоконтроль	Бєлова А.В., асистент		

7. Дата видачі завдання “04” вересня 2021 року.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломного проєкту (роботи)	Строк виконання етапів проєкту (роботи)	Примітка
1	Постановка завдання роботи	1 тиждень	Завдання
2	Аналіз предметної області	2 тиждень	Розділ 1
3	Розробка принципової схеми	3 тиждень	Розділ 2
4	Розробка макету за допомогою САПР	4 тиждень	Розділ 2
5	Розробка робочого макету	5-6 тиждень	Розділ 2
6	Розробка архітектури програми	7 тиждень	Розділ 3
7	Розробка програми	8-9 тижні	Розділ 3
8	Тестування та відлагодження програмного забезпечення, розробка керівництва користувача	10 тиждень	Розділ 4
9	Оформлення пояснювальної записки та документів до неї.	11 тиждень	Додатки
10	Нормоконтроль та рецензування	12 тиждень	
11	Захист роботи	13 тиждень	

Студент(ка)

Зіненко Д.І.
(підпис) (прізвище та ініціали)

Керівник проєкту (роботи)

Зеленьова І.Я.
(підпис) (прізвище та ініціали)

РЕФЕРАТ

Пояснювальна записка до кваліфікаційної роботи магістра: 86 с., 7 табл., 43 рис., 2 дод., 32 джерел.

ПРОТОТИПУВАННЯ, МАКЕТУВАННЯ, РОБОТОТЕХНІКА,
НАЛАГОДЖУВАЛЬНІ ПЛАТФОРМИ, C++, ARDUINO.

Об'єкт дослідження – апаратні та програмні рішення підсистем автоматичного контролю.

Мета роботи – поліпшення якості процесу контролю освітлення в приміщенні за рахунок апаратно-програмної підсистеми автоматизації контролю.

Матеріали, методи та технічні засоби: структурне та об'єктно-орієнтоване програмування, налагоджувальна платформа Arduino NANO, мови програмування C++, персональний комп'ютер з процесором AMD Ryzen 7 під управлінням операційної системи Microsoft Windows 10.

Результати. Створено проєкт підсистеми автоматичного контролю освітлення у приміщенні, розроблено інструкції для виготовлення макетів підсистеми.

Висновки. За результатами роботи було розроблено детальне керівництво користувача, створено віртуальний та робочий макет підсистеми, та протестовано програмне забезпечення.

Галузь використання – робототехніка та програмування мікроконтролерів.

ABSTRACT

Explanatory note to the diploma qualifying work of the master: 86 pages, 7 tables, 43 figures, 2 appendixes, 32 sources.

PROTOTYP, LAYOT, ROBOTICS, ADJUSTMENT PLATFORMS, C ++, ARDUINO.

The object of research is software and hardware for the implementation of the automatic control subsystem for room lighting control.

The purpose of the research is to implement the of the automatic control subsystem for room lighting control, to develop software, hardware solutions and instructions for the use of technical components.

Materials, methods and technical means: structural and object-oriented programming, Arduino NANO debug board, C ++ programming languages, personal computer with AMD Ryzen 7 processor running Microsoft Windows 10 operating system.

Results. The project of the subsystem of automatic control subsystem for room lighting control is created, the instructions for production of models of the subsystem are developed.

Conclusions. Based on the results of the work, a detailed user manual was developed, a virtual and working layout of the subsystem was created, and the written software was tested.

Field of application - robotics and programming of microcontrollers.

ЗМІСТ

Перелік скорочень та умовних позначок	7
Вступ.....	8
1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ.....	10
1.1 Огляд різновидів автоматизації	10
1.2 Автоматика та робототехніка	14
1.3 Огляд існуючих рішень та стандартів освітлення в приміщеннях	20
1.4 Аналіз компонентів	24
1.5 Аналіз та обґрунтування вибору програмних засобів.....	31
1.6 Постановка задачі розробки та дослідження.....	34
2 РОЗРОБКА МАКЕТУ ПІДСИСТЕМИ АВТОМАТИЧНОГО КОНТРОЛЮ ОСВІТЛЕННЯ У ПРИМІЩЕННІ	35
2.1 Схема електрична принципова	35
2.2 Прототипування на платформі Tinkercad	37
2.3 Інструкція створення макету на платформі Tinkercad	39
2.4 Інструкція створення робочого макету підсистеми.....	43
2.5 Висновки до розділу	46
3 РОЗРОБКА ПРОГРАМНОЇ РЕАЛІЗАЦІЇ ПІДСИСТЕМИ.....	47
3.1 Структура програми.....	47
3.2 Алгоритм роботи програми.....	48
3.3 Розробка коду на платформіTinkercad	49
3.4 Розробка програми за допомогою ArduinoIDE	56
3.5 Висновки до розділу	61
4 ТЕСТУВАННЯ ПІДСИСТЕМИ ТА КЕРІВНИЦТВО КОРИСТУВАЧА	62
4.1 Тестування системи	62
4.2 Керівництво користувача	64
Висновки	66
Перелік джерел посилання	67
Додаток А Текст програми.....	70
Додаток Б Слайди презентації	79

ПЕРЕЛІК СКОРОЧЕНЬ ТА УМОВНИХ ПОЗНАК

ARM –	Advanced RISC Machines (продвинуті RISC машини);
GPIO –	General-purpose input/output (Інтерфейс введення/виведення загального призначення);
HTML –	Hyper Text Markup Language (мова розмітки гіпертексту);
IP –	Internet Protocol (інтернет протокол або міжмережевий протокол);
IDE –	Integrated Development Environment (інтегроване середовище розробки);
MEMS –	мікроелектромеханічні системи;
SPI –	Serial Peripheral Interface (послідовний периферійний інтерфейс);
RAD –	Rapid application development (швидка розробка застосунків);
RISC –	Reduced Instruction Set Computing (обчислення зі скороченим набором команд);
АСП –	автоматизована система планування;
АСНД –	автоматизована система наукових досліджень;
АСУТП –	автоматизована система управління технологічним процесом;
АСКТП –	автоматизована система керування технологічним процесом;
АЕК –	автоматизований експериментальний комплекс;
ГАП –	гнучке автоматизоване виробництво;
САК –	систем автоматичного керування;
САПР –	система автоматизованого проєктування.

ВСТУП

Стрімкий розвиток робототехніки сприяє збільшенню кількості мікроконтролерів та налагоджувальних платформ на базі цих мікроконтролерів. Доступність та велика популярність таких платформ як Arduino дає можливість розробки різних систем. Провівши аналіз різновидів автоматизації було виявлено, що велику кількість задач можливо виконувати без втручання людини, що дає можливість не відволікавшись виконувати першеступеневе завдання. Тому роботу було присвячено вирішенню актуального завдання, а саме розробці підсистеми автоматичного контролю освітлення у приміщенні на базі Arduino NANO. Розробка подібної підсистеми виводить робототехніку на новий рівень та дає безліч можливостей.

Об'єкт дослідження – апаратні та програмні рішення підсистем автоматичного контролю.

Мета роботи – поліпшення якості процесу контролю освітлення в приміщенні за рахунок апаратно-програмної підсистеми автоматизації контролю.

Завдання, які було поставлено для досягнення мети:

- аналіз мікроконтролерів;
- аналіз ринку деталей;
- аналіз готових рішень;
- аналіз мови програмування;
- аналіз середовища розробки;
- розробка готових рішень;
- розробка інструкцій.

У першому розділі проведено аналіз предметної області, аналіз необхідної інформації для побудови обраної підсистеми.

У другому розділі створено макети підсистеми, спочатку у віртуальній середі розробки, потім за допомогою реальних рішень створено робочий макет підсистеми.

У третьому розділі розроблено програмну реалізацію підсистеми, спочатку на платформі для віртуального налагоджування, потім для реального макету.

У четвертому розділі проведено тестування системи, визначено переваги та недоліки створеної підсистеми, та розроблено керівництво користувача.

У результаті створено підсистему автоматичного контролю освітлення у приміщенні, протестовано на робочому макеті, зроблено висновки та виділено недоліки для доопрацювання у майбутньому. Розроблено керівництво користувача, та докладно описано як користуватись системою.

Наукова новизна отриманих результатів полягає у тому, що розроблено програмне та апаратне забезпечення, що слугує подальшому розвитку в напрямку автоматизації.

Практична значимість полягає в розробленій підсистемі автоматичного контролю освітлення, яке дозволяє скоригувати рівень освітлення. Оригінальне рішення автора є у взаємодії механічних ролетів з датчиками освітленості, які отримують дані з робочого місця та порівнюються з ідеальним значенням, що було отримано з проаналізованих стандартів освітлення в Україні. Проведені експерименти підтвердили працездатність розробленого програмного забезпечення. Дану систему можливо використовувати у різних приміщеннях, від лікарень до робочих приміщень заводу з доопрацюваннями, наприклад, розроблену версію програми доцільно використовувати у приміщеннях вищих навчальних закладів, а також школах, гімназіях, ліцеях.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Огляд різновидів автоматизації

Автоматизація є одним з напрямів науково-технічного прогресу, який спрямовано на застосування саморегульованих технічних засобів, економіко-математичних методів і систем керування, що звільняють людину від участі у процесах отримання, перетворення, передавання і використання енергії, матеріалів чи інформації, істотно зменшують міру цієї участі чи трудомісткість виконуваних операцій [1]. Автоматизація дозволяє підвищити продуктивність праці, поліпшити якість продукції, оптимізувати процеси управління, відсторонити людину від виробництв, небезпечних для здоров'я. Автоматизація, за винятком найпростіших випадків, вимагає комплексного, системного підходу до вирішення завдання. Застосовувані методи обчислень іноді копіюють нервові і розумові функції людини [1].

Основними перевагами автоматизації є:

- збільшення пропускної здатності або продуктивності;
- підвищення якості та передбачуваності якості;
- підвищена надійність, процесів або продуктів;
- підвищення узгодженості продукції;
- скорочення прямих людських витрат на робочу силу та видатків.

Основними недоліками автоматизації є:

- загрози безпеці/уразливості: автоматизована система може мати обмежений рівень інтелекту, отже, більш сприйнятлива до скоєння помилок за межами свого безпосереднього набору знань;
- непередбачувані / надмірні витрати на розробку: вартість наукових досліджень та розробки процесу автоматизації, може перевищити вартість заощаджень від самої автоматизації;
- висока ціна: автоматизація нового продукту, як правило, вимагає дуже великих початкових інвестицій порівняно з питомою вартістю продукту, отже видатки на автоматизацію, може бути закладено у вартість багатьох

продуктів протягом тривалого часу [2].

Мета автоматизації — підвищення продуктивності праці, поліпшення якості продукції, покращення керування, усунення людини від виробництв, небезпечних для здоров'я [2].

Автоматизація, за винятком простих випадків, вимагає всебічного, системного підходу до рішення завдання, тому окремі вирішення завдань засобами автоматизації, зазвичай називаються системами, наприклад:

- автоматична система керування;
- автоматизована система керування;
- система автоматизації проєктних робіт(САПР);
- автоматизована система керування технологічним процесом (АСК ТП).

Система автоматизації — інформаційно об'єднана сукупність програмованих пристроїв автоматизованого та автоматичного контролю, регулювання та керування [3].

Галузі, де автоматизація є важливою: виробництво сталі, гірнича справа, машинобудування, хімічне виробництво, харчова промисловість, енергетика, керування автомобільним, залізничним і повітряним транспортом.

Складовими елементами системи автоматизації є підсистеми:

- технологічної та аварійної сигналізації;
- автоматичного блокування і технологічного захисту;
- аварійного керування;
- реєстрації стану керованого процесу і дій оператора;
- обмін даними між пристроями системи та зі зовнішніми інформаційними системами [3].

Сучасні системи автоматизації можуть бути досить складними. До складу систем автоматизації входять датчики (сенсори), пристрої введення, пристрої керування (контролери), виконавчі пристрої, пристрої виведення, комп'ютери, сервери, робочі станції.

Основна тенденція розвитку систем автоматизації йде в напрямку створення автоматичних систем, які здатні виконувати задані функції або процедури без участі людини. Роль людини полягає в підготовці вихідних даних, вибір алгоритму (методу рішення) і аналізі отриманих результатів. Також в подібних системах передбачається поступово нарощувана захист від нестандартних подій (аварій) або способи їх обходу (з точки зору науки катастроф це не одне і те ж) [4].

Основними видами систем автоматизації є:

- автоматизована система планування (АСП);
- автоматизована система наукових досліджень (АСНД);
- система автоматизованого проєктування (САПР);
- автоматизований експериментальний комплекс (АЕК);
- гнучке автоматизоване виробництво (ГАП) і автоматизована система управління технологічним процесом (АСУТП);
- автоматизована система управління експлуатацією (АСУ) і система автоматичного управління (САУ).

Система автоматизованого проєктування (САП або САПР) або автоматизована система проєктування (АСП) — автоматизована система, призначена для автоматизації технологічного процесу проєктування виробу, результатом якого є комплект проєктно-конструкторської документації, достатньої для виготовлення та подальшої експлуатації об'єкта проєктування. Реалізується на базі спеціального програмного забезпечення, автоматизованих банків даних, широкого набору периферійних пристроїв [4].

САПР виконує такі функції:

- конструкторська частина — розробка повного комплексу конструкторської документації;
- технологічна частина — розрахунок і проєктування технологічних схем, технологічного оснащення, транспорту;
- архітектурно-будівельна частина — розрахунок і проєктування металевих і залізобетонних конструкцій;

- санітарно-технічні системи — проєктування теплопостачання, опалення і вентиляції виробничих і адміністративних корпусів, а також водопостачання і каналізації;
- електротехнічні системи — розрахунок і проєктування електропостачання, електросилового устаткування, світлотехнічної частини проєктів, телемеханізації електропостачання;
- гідротехнічні спорудження — розрахунок і проєктування напірного і безнапірного гідротранспорту відвальних хвостів, стійкості укосів хвостосховищ [5];
- системи автоматизації — розробка схем зовнішніх з'єднань, електричних і трубних проводок щитів автоматики;
- кошторисна частина — складання локальних і зведених кошторисів, відомостей матеріалів, специфікацій, комплектація обладнання.

Робота з САПР полягає у створенні геометричної моделі виробу (двовимірної чи тривимірної, твердотілої), генерацію на основі цієї моделі конструкторської документації (креслень виробу, специфікацій тощо) і його наступний супровід [5].

Слід зазначити, що термін «САПР» стосовно промислових систем має ширше тлумачення, ніж CAD — він включає CAD, CAM і CAE.

Автоматизована система керування технологічним процесом (АСК ТП) — автоматизована система у вигляді комплексу програмних і технічних засобів, призначена для вироблення та реалізації керувальної дії на технологічний об'єкт керування згідно з прийнятими критеріями керування.

Під АСК ТП зазвичай розуміється комплексне рішення, що забезпечує автоматизацію основних технологічних операцій на виробництві в цілому або якійсь його ділянці, що випускає відносно завершений продукт.

Автоматизована система, призначена для оптимізації керування технологічними процесами виробництва [5].

АСК ТП — це людино-машинна система, що забезпечує автоматизований збір інформації з вимірювальних перетворювачів сигналів і її

первинну обробку (фільтрування сигналів, лінеаризація характеристик перетворювачів, «офізичення» сигналів, тобто перетворення та візуалізації сигналів у значеннях параметрів у фізичних одиницях вимірювання: °С, Па, В та ін.) для розрахунку, видачі та реалізації керувальних впливів на технологічне обладнання відповідно до прийнятих критеріїв керування [6].

АСК ТП здійснює реалізацію впливів на об'єкт керування в темпі перебігу технологічного процесу, тобто в реальному часі, при цьому забезпечує керування об'єктом в цілому, а її технічні засоби беруть участь у виробленні рішень з керування. Важливо зробити акцент на слові «автоматизована» [6]. Під цим мається на увазі, що система керування аж ніяк не повністю автономна (самостійна), і потрібна участь людини (оператора) для реалізації певних завдань. Зазначеними обставинами АСК ТП якісно відрізняється від традиційних систем автоматичного керування (САК), які представляють технічні засоби для автоматизації дій людини на окремих ділянках технологічного процесу і призначені для роботи без будь-якого контролю з боку людини та повністю автономні. На відміну від цього в АСК ТП реалізується автоматизований процес прийняття рішень з керування технологічним процесом як єдиним цілим, для чого в ній застосовують різне «інтелектуальне» автоматичне обладнання обробки інформації, в першу чергу сучасні багатофункціональні, високопродуктивні промислові комп'ютери [6].

1.2 Автоматика та робототехніка

Завданнями автоматизації виробництва є автоматичний контроль (включаючи автоматичну сигналізацію та автоматичний захист обладнання), автоматичне регулювання та автоматичне керування. Ці завдання вирішуються шляхом створення систем автоматизації як інформаційно об'єднаної сукупності програмованих пристроїв автоматизованого та автоматичного контролю, регулювання та управління. Системи автоматизації будуються з урахуванням пристроїв автоматики [7].

Автоматика — галузь науки і техніки, яка розробляє технічні засоби та методи для здійснення технологічних процесів без безпосередньої участі людини.

Робототехніка — прикладна наука, що займається розробкою автоматизованих технічних систем і є найважливішою технічною основою розвитку виробництва [8].

Робототехніка спирається такі дисципліни, як електроніка, механіка, кібернетика, телемеханіка, мехатроніка, інформатика, і навіть радіотехніка і електротехніка. Виділяють будівельну, промислову, побутову, медичну, авіаційну та екстремальну (військову, космічну, підводну) робототехніку.

Робототехніка орієнтована на створення роботів і робототехнічних систем, призначених для автоматизації складних технологічних процесів і операцій, у тому числі таких, що виконуються в недетермінованих умовах, для заміни людини під час виконання важких, утомливих і небезпечних робіт [8].

Під керівництвом роботом розуміється рішення комплексу завдань, що з адаптацією робота до кола вирішуваних їм завдань, програмуванням рухів, синтезом системи управління та її програмного забезпечення.

За типом управління робототехнічні системи поділяються на наступні.

Біотехнічні:

- командні (кнопкове та важільне управління окремими ланками робота);
- копіюючі (повтор руху людини, можлива реалізація зворотного зв'язку, що передає зусилля, екзоскелети);
- напіваавтоматичні (управління одним командним органом, наприклад, рукояткою всієї кінематичної схемою робота);

Автоматичні:

- програмні (функціонують по заздалегідь заданій програмі, переважно призначені на вирішення одноманітних завдань у постійних умовах оточення);
- адаптивні (вирішують типові завдання, але адаптуються під умови функціонування);

- інтелектуальні (найрозвиненіші автоматичні системи);

Інтерактивні:

- автоматизовані (можливе чергування автоматичних та біотехнічних режимів);
- супервізорні (автоматичні системи, в яких людина виконує лише цільовказівні функції);
- діалогові (робот бере участь у діалозі з людиною на вибір стратегії поведінки, при цьому зазвичай робот оснащується експертною системою, здатною прогнозувати результати маніпуляцій і дає поради щодо вибору мети) [8].

Прототипування – швидка «чорнова» реалізація базової функціональності для аналізу роботи системи в цілому. Під час прототипування видно більш детальна картина пристрою системи. Використовується в машино- і приладобудуванні, програмуванні і в багатьох інших областях техніки. Прототипування є важливим етапом розробки, тому що дозволяє зрозуміти як буде працювати система, передбачити та виправити помилки. Після етапу прототипування обов'язково слідують етапи перегляду архітектури системи, розробки, реалізації та тестування кінцевого продукту.

Прототипування не обов'язково виконується в рамках тих же технологій, що й розроблювана система. Як правило, прототип стає додатком до технічного завдання.

Якості, якими повинен володіти ефективний прототип:

- етап створення прототипу не повинен бути затяжним;
- ефективні прототипи є сфокусованими, це означає що слід звертати увагу на складні частини при створенні прототипів. Необхідно знайти взаємодії, які давно відомі в теорії;
- необхідно звертати увагу на елементи взаємодії, які принесуть користь вашому продукту [5].

Іноді тестування алгоритмів управління на реальному об'єкті управління може виявитися досить ресурсомістким процесом. Якщо об'єкт управління

знаходиться в безпосередній доступності, то може використовуватися техніка швидкого прототипування (Rapid Prototyping) на напівнатурному стенді реального часу. В його основі лежить машина реального часу – комп'ютер, що працює в режимі жорсткого реального часу, і забезпечений фізичними інтерфейсами для зв'язку з обладнанням. Швидке прототипування називається техніка, при якій реальний об'єкт підключений до машини реального часу, на якому запущений алгоритм управління. Ключове зручність швидкого прототипування полягає в тому, що, по-перше, не потрібно спеціально програмувати машини реального часу, на них працюють звичайні моделі з мінімальними доробками, а, по-друге, можна налаштовувати параметри системи управління прямо в процесі роботи. Це значно прискорює процес тестування на реальній машині. Тому таке прототипування і називається швидким.

У разі, якщо машина реального часу відсутня, є можливість зімітувати реальний час на робочому комп'ютері за допомогою різних програм (Proteus Design Suite, Altium Designer, TkGate та інші). За допомогою цих програм можна побачити чи буде працювати створена система. Дозволяє передбачити помилки, але не фізичні.

При швидкому прототипування (англ. Rapid prototyping або throwaway prototyping) передбачається, що створюється макет, який на якомусь етапі буде залишений («викинутий») і не стане частиною готової системи [5].

Основна перевага такого підходу - в швидкості: у відповідь на свої вимоги замовник майже відразу отримує прототип інтерфейсу, і відразу може уточнити вимоги, до того як розпочато написання робочого коду системи. Вартість зміни вимог на цьому етапі дуже низька, оскільки немає коду, який потрібно було б переписувати.

Дуже важливо, щоб таке прототипування було виконано в найкоротші терміни, оскільки в даному випадку витрачаються час і ресурси на код, який не буде в подальшому використаний.

Швидке прототипування не обов'язково виконується в рамках тієї ж платформи і тих же технологій, що й розробляється система. Для прототипу

графічного інтерфейсу користувача (GUI) можуть використовуватися як стандартні HTML-сторінки, або прототип може готуватися в програмі, спеціально призначеної для створення макетів.

Прототипування з використанням налагоджувальних платформ використовується для керування електронними пристроями. По суті, це – однокристальний комп'ютер, здатний виконувати прості завдання. Використання однієї мікросхеми значно знижує розміри, енергоспоживання і вартість пристроїв, побудованих на базі мікроконтролерів [5].

Як, найбільш поширеними сімействами мікроконтролерів є:

- AVR;
- ARM cortex;
- MSP.

AVR – сімейство восьмибітних мікроконтролерів, що раніше випускалися фірмою Atmel, потім Microchip. Мікроконтролери AVR мають гарвардську архітектуру (програма і дані знаходяться в різних адресних просторах) і систему команд, близьку до ідеології RISC. Процесор AVR має 32 8-бітових регістра загального призначення, об'єднаних в регістровий файл. На відміну від «ідеального» RISC, регістри не абсолютно ортогональні [9].

ARM (первісно Advanced RISC Machine – поліпшена RISC машина, попередник Acorn RISC Machine) – 32-бітна RISC архітектура процесорів, яку розробила компанія ARM Limited. Широко застосовується у розробці портативних пристроїв. Головною причиною цього є використання енергоощадних технологій. Саме тому ця архітектура домінує у пристроях, головною ідеєю яких є енергоощадність [10].

Сьогодні сімейство ARM займає приблизно 75 % всіх портативних 32-бітних RISC процесорів, що робить його найвикористовуванішим серед усіх 32-бітних архітектур. Процесори ARM знайшли своє застосування у багатьох пристроях (мобільні телефони і смартфони, планшети, КПК, цифрові аудіоплеєри, калькулятори, ігрові консолі тощо), комп'ютерній периферії: маршрутизатори, NAS-сервери, апаратні брендмауери [10].

MSP430 — сімейство 16-розрядних мікроконтролерів фірми «Texas Instruments». Ядро MSP430 16-бітове. Систему команд постаралися зробити максимально ортогональною з різноманітними способами адресації. Ортогональність системи команд означає, що у будь-якій команді може використовуватися будь-який спосіб адресації операнда: константа, прямий з регістра або пам'яті, непрямий і т. п.

На момент народження сімейства основний акцент був зроблений на зниження енергоспоживання. Однак з тих пір економія енергії стала ідеєю-фікс електронної техніки, і MSP430 активно тіснять на цьому п'єдесталі інші виробники зі своїми архітектурою [11].

Ключовою відмінністю і «візитною карткою» сімейства MSP430 є можливість тактіровать будь-який модуль периферії асинхронно від ядра. У переважній більшості однокристальних мікроконтролерів периферія синхронна з ядром (за винятком окремих спеціальних вузлів). Така особливість дозволяє гнучко управляти швидкістю (а значить, і споживанням) кожного модуля. Крім того, унікальним є також модуль USB інтерфейсу: він має окремий домен харчування, що спрощує розробку батарейних приладів з USB-інтерфейсом.

Розглянемо декілька найбільш використовуваних налагоджувальних платформ на базі цих мікроконтролерів:

- ArduinoNano V 3.0;
- STM32F103C8T6;
- MSP430F5529.

Результати порівняння налагоджувальних платформ наведено в табл. 1.1.

Таблиця 1.1 – Порівняння налагоджувальних платформ

Назва платформи	ArduinoNano 3	STM32F103C8T6	MSP430F5529
Мікроконтролер	ATmega328P	ARM32cortex-M3	MSP430
Робоча напруга	5В	5В	5В

Напруга живлення (рекомендована)	7-12В	2-3,6 В	1,8-3,6 В
-------------------------------------	-------	---------	-----------

Продовження таблиці 1.1

Назва платформи	ArduinoNano 3	STM32F103C8T6	MSP430F5529
Цифрові входи / виходи	14(6 - ШІМ)	15	66
Аналогові входи	8	15	-
Flash-пам'ять	32КБ	64 КБ	128 КБ
SRAM	2КБ	20 КБ(RAM)	8 КБ(RAM)
Тактова частота	16 МГц	72 МГц	25 МГц
Ціна	170 грн	276грн	500 грн

З таблиці порівняння можна зробити висновок, що ArduinoNano 3 є найдешевшим аналогом на ринку мікроконтролерів такого типу, при цьому, треба відмітити декілька особливо важливих пунктів, які стосуються конструювання системи автоматичного контролю освітлення, а саме напругу живлення(рекомендована), кількість пам'яті, тактову частоту і ціну. Параметри дорожчих аналогів кращі ніж у ArduinoNano 3, але для побудови потрібної системи такі потужності не потрібні, тому ArduinoNano 3 буде достатньо для створення проєкту.

1.3 Огляд існуючих рішень та стандартів освітлення в приміщеннях

Основними нормованими показниками освітленості робочого місця є загальний індекс кольору, коефіцієнт пульсацій освітленості. Для всіх робочих місць усередині приміщень та для робочих місць поза приміщеннями, на яких виконується конкретна робота (залізничні станції, аеропорти, кар'єри тощо), основною величиною, що нормується, є освітленість на робочому місці. Розмір нормованої освітленості залежить, передусім, від характеру виконуваної роботи. При освітленні вулиць і доріг величиною, що нормується, служить

яскравість дорожнього покриття. Вона встановлюється залежно від категорії осіб, інтенсивності руху, характеру навколишнього оточення [12].

Загальний індекс кольору – відношення відтворення кольорів предметів при освітленні їх даним джерелом світла до відтворення кольорів цих же предметів, що висвітлюються джерелом світла, прийнятим за зразок. За «стандартне» джерело було прийнято світло теплових випромінювачів, ламп розжарювання – їхній загальний індекс кольоропередачі прийнятий рівним 100. Прийнято таку систему оцінки якості кольору: $R_a = 90$ – відмінна якість; $90 > R_a > 80$ – дуже добре; $80 > R_a > 70$ – добрий; $70 > R_a > 60$ – задовільний; $60 > R_a > 40$ – прийнятне; $R_a < 40$ – низька [12].

Наприклад, в Україні норми освітлення встановлюють таке: для підприємств поліграфічної, текстильної, лакофарбової галузей промисловості, а також для хірургічних відділень лікарень загальний індекс кольору має бути не нижчим за 90. В Україні нормується також коефіцієнт пульсації освітленості.

У газорозрядних джерел світла - люмінесцентних, металогалогенні, натрієві лампи - величина світлового потоку змінюється з подвоєною частотою струму мережі. В Україні, Росії, США, країнах СНД, Європи та Азії частота змінного струму в електричних мережах дорівнює 50 Гц. Отже, світловий потік ламп змінюється («пульсує») 100 або 120 разів на секунду - всі газорозрядні лампи мерехтять з такою частотою [12]. Око цих мерехтіння не помічає, але вони сприймаються периферійним зором, організмом і на підсвідомому рівні можуть викликати неприємні явища – підвищену стомлюваність, біль голови, можливо стреси. Крім цього, при освітленні пульсуючим світлом предметів, що обертаються або вібрують, виникає так званий «стробоскопічний ефект», коли при збігу частоти обертання або вібрації з частотою пульсацій світла предмети здаються нерухомими, а при неповному збігу – що обертаються з дуже малими швидкостями. Це викликає у людей помилкові реакції і є однією із серйозних причин травматизму на виробництві [13].

Глибина пульсацій вимірюється коефіцієнтом пульсації освітленості. В українських нормах освітлення встановлено, що глибина пульсації освітленості на робочих місцях не повинна перевищувати 20%, а для деяких видів

виробництва – 15%. В Україні головним документом, що встановлює параметри освітлення, є Будівельні норми та правила БНіП 23-05-95. Крім цих норм, існують Санітарні правила та норми СанПіН 2.21/2.1.1.1278-03 та безліч галузевих норм [13].

У Європі існують загальні Європейські норми освітлення, кілька десятків спеціалізованих норм, а також багато національних норм та правил. У Європейських нормах освітленості для ряду приміщень введено ще один параметр, що нормується: для робочих місць, оснащених моніторами (тобто практично для всіх робочих місць в офісах) встановлюються вимоги до максимальної яскравості тих поверхонь світильників, які можуть відображатися в екранах [12].

Освітленість вимірюється у Лк – люксах; 1 люмен/м. кв. (люмен Lm – одиниця величини світлового потоку) – показник, якому мають відповідати офісні та інші нежитлові будинки, для величини якого існують вітчизняні та міжнародні норми. Норми освітленості в Україні представлені в таблиці 1.2.

Таблиця 1.2 – Стандарти освітлення для різних приміщень [12]

Освітлювані об'єкти	Середня горизонтальна освітленість, лк
Класні кімнати, кабінети, аудиторії загальноосвітніх шкіл, шкіл інтернатів, середньоспеціальних та професійно-технічних установ, лабораторії, навчальні кабінети фізики, хімії, біології та інші	500
Аудиторії, навчальні кабінети, лабораторії технікумів та вищих навчальних закладів	400
Кабінети інформатики та обчислювальної техніки	200
Навчальні кабінети технічного креслення та малювання	500
Лаборантські при навчальних кабінетах	400

Продовження таблиці 1.2

Лабораторії органічної та неорганічної хімії, препараторські	400
Майстерні з обробки металів та деревини	300
Освітлювані об'єкти	Середня горизонтальна освітленість, лк
Інструментальна кімната майстра інструктора	300
Кабінети обслуговуючих видів праці	400
Спортивні зали	200
Снарядні, інвентарні, господарські комори	50
Криті басейни	150
Актові зали, кіноаудиторії	200
Естради актових залів, кабінети та кімнати викладачів	300
Рекреації	150

На даний час існує багато рішень проблеми автономної роботи системи закриття вікон, у різних цінових категоріях. Для українського ринку було проаналізовано найпопулярніші магазини електроніки в яких є можливість придбати розумні електрокарнізи. Для отримання найкращих результатів було проведено аналіз товарів у різних цінових категоріях, було проаналізовано:

- функції;
- ціна;
- матеріал;
- вид;
- спосіб кріплення;
- потужність мотора;
- вага.

Для порівняння було обрано три найпопулярніші продукти у найбільших магазинах України. Результати проведеного аналізу записано в таблиці 1.3.

Таблиця 1.3 – Порівняння готових рішень схожих підсистем

Назва	Xiaomi Aqara B1 Curtain Rail	Torro	Forest BCS
Функції	Управління за допомогою спеціального пульта від цієї компанії або за допомогою мобільного пристрою	Управління за допомогою пульта	Управління за допомогою системи «Розумний будинок», за допомогою спеціального пульта
Ціна	2900-5000 грн.	6900-8900 грн.	16700-18900 грн.
Матеріал	Алюміній	Алюміній	Алюміній
Потужність мотора	50 Гц	40 Гц	60 Гц
Вага	1.3 кг	1.6 кг	3.0 кг

З таблиці порівняння можна зробити висновок, що цілком автономних популярних рішень на ринку дуже мало, а ті, що приведені у таблиці можуть працювати автономно тільки після доопрацювання.

1.4 Аналіз компонентів

Для розробки системи автоматичного контролю освітлення у приміщення необхідно обрати об'єкт. Проведений аналіз показав, що досить мало рішень для керування кутом нахилу та ступенем відкриття штор. Таким чином, розробка системи автоматичного контролю освітлення у приміщення за допомогою «розумних» штор є актуальним завданням. Дослідження складових частин показало, що для створення подібної системи потрібно обрати такі основні елементи:

- мікроконтролер;
- двигуни;
- драйвер двигунів;
- датчики освітленості;
- джерела живлення.

Проведений аналіз ринку Arduino мікроконтролерів свідчить про те що він досить великий, особливо це стосується підгрупи контролерів для розробки різноманітних систем.

Мікроконтролер виконаний у вигляді мікросхеми спеціалізований комп'ютер, що включає мікропроцесор, оперативну та постійну пам'ять для збереження виконуваного коду програм і даних, порти вводу-виводу і блоки зі спеціальними функціями [14]. Мікроконтролери досить широко використовують у багатьох система керування будь-якого процесу, особливо це стосується таких відомих систем як «Розумний будинок». Для досягнення рівня автоматичного контролю будь-якого процесу треба проаналізувати ринок мікроконтролерів та обрати найкращій варіант для обраної системи. Через те що перед обраною системою стоїть задача керування «розумними шторами», розміри даної системи повинні бути мінімальними, але добре виконувати свої функції, для цього системі потрібно зчитувати данні з шести датчиків, обраховувати та порівнювати їх з даними взятих зі стандартів, та передавати потрібні команди на два мотори, які в свою чергу будуть регулювати кут нахили та ступінь відкриття штор [14].

Тому, було прийняте рішення обрати мікроконтролер більш малих розмірів, але він повинен виконувати всі описані вище завдання та функції. Також слід розглянути можливість доопрацювання системи, та додавання додаткових функцій, наприклад:

- керування за допомогою пульта;
- керування ступенем яскравості встановлених ламп;
- підключення до інших «розумних» систем.

Тому вибір пав на Arduino Nano V3 ATmega328P-AU. У свою чергу ринок Arduino мікроконтролерів дуже великий, тому якщо побудувати систему на цій платформі, вона буде не лише дешевою та доступною, але і матиме великі можливості в доопрацюванні. Технічні характеристики обраної налагоджувальної плати:

- мікроконтролер: ATmega328P;
- тип корпусу: TQFP-32;
- робоча напруга: 5В;
- вхідна напруга (рекомендована): 7-12В;
- цифрових входів/виходів: 14 (з яких 6 можуть бути використані як ШІМ);
- аналогових входів: 8;
- сила струму на входах/виходах: 40 мА;
- сила струму для виходу 3.3В: 50 мА;
- пам'ять: 32 кБ;
- SRAM: 2 кБ;
- EEPROM: 1 кБ;
- частота: 16 МГц.

Як вже обговорювалося вище, через специфіку проекту остаточні розміри повинні бути максимально малими, тому розміри всіх обраних компонентів повинні бути мінімальними, але це не повинно відражатись на функціональних можливостях обраних компонентів. Тому було розглянуто три популярних рішення на ринку Arduino, а саме:

- кроковий двигун 28BYJ-48;
- сервопривід SG90;
- двигун з редуктором.

Таблиця порівняння цих трьох двигунів приведено в табл. 1.4.

Таблиця 1.4 – Характеристики обраних популярних двигунів

Назва	Кроковий двигун 28BYJ- 48	Сервопривід SG90	Двигун з редуктором
Напруга живлення	5 В	3,5 – 5 В	6-8 В
Кут обертання	360°	180°	360°
Розміри	39 x 29 x 28 мм	33x 30 x 13 мм	64x20x20 мм
Назва	Кроковий двигун 28BYJ- 48	Сервопривід SG90	Двигун з редуктором
Крутний момент	34,3 мН / м	196,1 мН / м	78,4 мН / м
Ціна	39 грн	51 грн	39грн

З таблиці порівняння можна зробити кінцевий вибір, а саме кроковий двигун 28BYJ-48, тому що напруга живлення підійде до раніше обраної налагоджувальної плати, кут обертання 360° вирішить багато інженерних проблем, розміри досить малі, крутний момент дозволить повертати, рухати майбутні конструкції.

Щоб керувати обертанням мотора, будь-якому контролеру необхідне спеціальний пристрій, яке називають драйвером мотора. Головна функція драйвера – змінювати полярність на навантаженні. А якщо як навантаження буде двигун постійного струму, то зміна полярності призведе до зміни напрямку його обертання [15]. Двигуни або мотори частіше всього рекомендують використовувати з конкретними драйверами, виходячи з цього було обрано для порівняння крокового двигуна ULN2003. Для обраного проєкту, в якому буде використовуватись два двигуни, було б доцільно вибрати драйвер який може одночасно працювати з двома двигунами, завдяки цьому буде досягнута економія місця та спрощена схема підключення, для цієї цілі було обрано драйвер двох двигунів на L298N, а також ще одне рішення, яке ще менше за розмірами - драйвер двигунів двоканальний на TB6612FNG [16].

Таблиця характеристик цих трьох драйверів двигунів представлено у табл. 1.5.

Таблиця 1.5 – Характеристики драйверів двигунів

Назва	Драйвер крокового двигуна ULN2003	Драйвер двигунів двоканальний на TB6612FNG	Драйвер двох двигунів на L298N
Напруга живлення двигуна	5/12 В	2.5/15 В	5/35 В
Напруга управління	5 В	5 В	5 В
Робочий струм драйвера	2 А	2 А	2 А
Розміри	35 x 33 x 10 мм	21x18x3 мм	43,5 x 43,2 x 29,4 мм
Вага	15 г	10 г	26 г
Мікросхема драйвера	ULN2003	TB6612FNG	L298N
Ціна	22 грн	67 грн	62 грн

З таблиці видно що драйвер крокового двигуна ULN2003 підходить по всім технічним параметрам для обраних двигунів, а також він має потрібні невеликі розміри та ціну.

Для того, щоб отримувати дані про освітленість в кімнаті існує декілька рішень, а саме:

- датчик освітленості на основі фоторезистора;
- датчик освітленості на основі фототранзистора.

Датчики освітленості — це датчики, які призначені для автоматичного увімкнення або вимкнення світла в залежності від освітлення навколишнього середовища. Принцип роботи датчиків освітленості, або як їх ще називають сутінкових датчиків, досить простий: такий датчик включає фототранзистор або фоторезистор, що здатні реєструвати рівень освітленості і при зміні параметрів передають сигнал на запуск заданих процесів, наприклад, включення або вимкнення світла в приміщенні [17].

Фоторезистор — фотоелектричний напівпровідниковий приймач випромінювання, принцип дії якого ґрунтується на ефекті фотопровідності — явищі зменшення опору напівпровідника у разі збудження носіїв заряду світлом. Характеризується однаковою провідністю незалежно від напрямку протікання струму [18].

Фототранзистор — біполярний транзистор, в якому інжекція нерівноважних носіїв здійснюється на основі внутрішнього фотоефекту; служить для перетворення світлових сигналів в електричні з одночасним посиленням останніх [19].

Фототранзистор являється більш точною та більш чутливою технологією, а також фототранзистор за здатністю оцінювати навколишню освітленість більш схожий на око людини [20].

Для порівняння було обрано три різних рішення, які представлені в табл. 1.6.

Таблиця 1.6 – Характеристики драйверів двигунів

Назва	Датчик освітленості GY-49 MAX44009	Датчик освітленості цифровий GY-302 BH1750FVI	Датчик освітленості ТЕМТ6000
Тип датчику	Фототранзистор	Фоторезистор	Фототранзистор
Напруга живлення	3,3 – 5 В	3 – 5 В	5 В
Інтерфейс підключення	I2C	I2C	Аналоговий
Діапазон освітленості	0,045 – 188000 Лк	0-65535 Лк	0,045 – 188000 Лк
Розміри	20 x 10 мм	14 x 18,5 мм	14 x 7 мм
Ціна	176 грн	36 грн	34 грн

З таблиці порівняння видно що датчик освітленості ТЕМТ6000 являється найкращим варіантом, бо має великий діапазон освітленості, малі розміри, просту технологію підключення, та невелику ціну.

Існує багато способів забезпечити живленням свою систему.

Li-ion — це гелеподібний або рідкий електроліт, у випадку з Li-Pol — спеціальний щільний полімер, насичений розчином літію [21]. Ємність таких акумуляторів напряду залежить від розміру, тобто чим більший акумулятор, тим довше буде літати квадрокоптер, але через великі розміри такі акумулятори досить важко встановити на раму, до того ж вага таких акумуляторів досить велика. Тому краще розглянути інші способи живлення, а саме:

- літій-іонний акумулятор;
- батарейка типу «крона».

Літій-іонний акумулятор (англ. Lithium-ion battery, скорочено Li-ion) — один з двох основних типів літієвих електричних акумуляторів з категорії вторинних електричних батарей, який відрізняється з літій-полімерним акумулятором лише типом електроліту, що використовується при їх виготовленні. Такі акумулятори менші за розмірами та мають ємність і невеликий розмір, декілька таких акумуляторів можна об'єднувати в одну велику батарею, що є досить великим плюсом [22].

Батарея «Крона» (також 1604, 6F22, 6R61 (сольова / цинкова); 1604A, 6LF22, 6LR61, MN1604, MX1604 (лужна / алкалінова); PP3, E-Block, 9V Brick Battery, АМ6, Корунд, 522, 6АМ6, CR-9V, ER9V (літієва)) - типорозмір батарейок. Назва «Крона» походить від марки випускалися в СРСР вугільно-марганцевих батарей цього типорозміру. Найголовнішим плюсом таких батарейок являється їх розмір і велика потужність [23].

Після обрання двох представників кожного типу джерел живлення, було проведено порівняння та результати було занесено у табл. 1.7.

Таблиця 1.7 – Порівняння джерел живлення

Назва	«Крона»	Samsung ICR18650-22P
Форма	Прямокутний паралелепіпед	Циліндр
Розміри	48,5 мм × 26,5 мм × 17,5 мм	65 мм × 17 мм
Маса	53 гр	40 гр
Назва	«Крона»	Samsung ICR18650-22P
Напруга	9 В	4.2 В
Електроємність	625 мА/ч	2200 мА/ч
Повторна зарядка	-	+
Цена	12 грн	70 грн

З таблиці наглядно видно що одна «крона» видає дуже велику напругу, але її електроємність не являється великою, її досить, щоб проводити тести, наприклад коли основні акумулятори заряджаються, у висновку «крону» можна використовувати у проєкті. Найкращим рішенням являється акумулятори Samsung, через велику електроємність, малу ціну, можливість повторної швидкої зарядки, та можливість об'єднання декількох акумуляторів в одну велику батарею.

1.5 Аналіз та обґрунтування вибору програмних засобів

Мова програмування (англ. Programming language) — це штучна мова, створена для передачі команд машинам, зокрема комп'ютерам. Мови програмування використовуються для створення програм, котрі контролюють поведінку машин, та запису алгоритмів.

Найпоширенішими мовами програмування у робототехніці являються C++ та Python.

C++ - компільований, статично типізований мова програмування загального призначення.

Підтримує такі парадигми програмування, як процедурне програмування, об'єктно-орієнтоване програмування, узагальнене програмування. Мова має багату стандартну бібліотеку, яка включає в себе поширені контейнери і алгоритми, введення-виведення, регулярні вирази, підтримку багатопоточності і інші можливості. С ++ поєднує властивості як високорівневих, так і низькорівневих мов. У порівнянні з його попередником - мовою С, - найбільшу увагу приділено підтримці об'єктно-орієнтованого і узагальненого програмування [24].

С ++ широко використовується для розробки програмного забезпечення, будучи одним з найпопулярніших мов програмування. Область його застосування включає створення операційних систем, різноманітних прикладних програм, драйверів пристроїв, додатків для вбудованих систем, високопродуктивних серверів, а також ігор. Існує безліч реалізацій мови С ++, як безкоштовних, так і комерційних і для різних платформ.

Python - високорівнева мова програмування загального призначення, орієнтований на підвищення продуктивності розробника і читання коду. Синтаксис ядра Python мінімалістичний. У той же час стандартна бібліотека включає великий обсяг корисних функцій [24].

Python підтримує структурний, об'єктно-орієнтоване, функціональне, імперативне і аспектно-орієнтоване програмування. Основні архітектурні риси - динамічна типізація, автоматичне керування пам'яттю, повна інтроспекція, механізм обробки виключень, підтримка багатопоточних обчислень, високорівневі структури даних. Підтримується розбиття програм на модулі, які, в свою чергу, можуть об'єднуватися в пакети [25].

Еталонної реалізацією Python є інтерпретатор CPython, що підтримує більшість активно використовуваних платформ. Він поширюється під вільною ліцензією Python Software Foundation License, що дозволяє використовувати його без обмежень в будь-яких додатках, включаючи пропрієтарні. Є реалізація інтерпретатора для JVM з можливістю компіляції, CLR, LLVM, інші незалежні реалізації. Проєкт PyPy використовує JIT-компіляцію, яка значно збільшує швидкість виконання Python-програм.

Позитивні сторони C++ у порівнянні з Python:

- продуктивність;
- ефективна робота з ресурсами.

Позитивні сторони Python у порівнянні з C++:

- швидкість розробки;
- різноманітність;
- легкість вивчення.

Для порівняння цих двох мов було обрано одну задачу: реалізувати бінарний пошук в масиві з 12 елементів. Найголовніший критерій який нас цікавить це час виконання програми, отже яка мова буде виконуватися швидше з нею і будемо працювати, для будуть приведені скріншоти з часом виконання програм.

Код для C++ у кінцевому варіанті представлений у додатку А.1.

Результати виконання програми приведені на рис. 1.1.

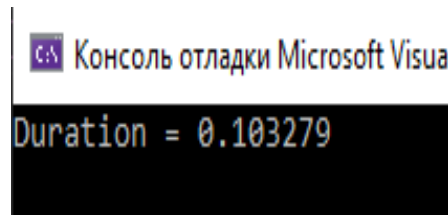


Рисунок 1.1 – Результати виконання програми для C++

Кінцевий код для виконання поставленої задачі для Python представлений у додатку А.2.

Результати виконання програми для Python приведені на рис. 1.2.

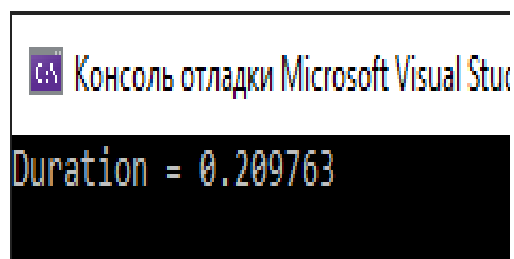


Рисунок 1.2 – Результати виконання програми для Python

Проведено декілька тестів, а саме п'ятнадцять для кожної програми, і C++ показав швидше виконання програми ніж Python, приблизно у два рази, це видно на рис. 1.1 та рис. 1.2.

Найпопулярнішим рішенням для розробки програм для мікроконтролерів Arduino являється інтегроване середовище розробки ArduinoIDE для Windows, MacOS та Linux, розроблене на C та C++, в свою чергу це рішення дуже спрощено для користування, бо було розроблено в навчальних цілях, але має всі необхідні функції для створення обраного проєкту.

Вихідний код середовища випущений під загальнодоступною ліцензією GNU версії 2. Підтримує мови Cі та C++ з використанням спеціальних правил структурування коду. Arduino IDE надає бібліотеку програмного забезпечення з проєкту Wiring, яка надає безліч загальних процедур введення та виведення. Для написаного користувачем коду потрібні лише дві базові функції для запуску ескізу та основного циклу програми, які скомпільовані та пов'язані з заглушкою програми `main()` у виконувану циклічну програму з ланцюжком інструментів GNU, також включеною до дистрибутиву IDE. Використовує програму `avrdude` для перетворення коду, що виконується, в текстовий файл у шістнадцятковому кодуванні, який завантажується в плату Arduino програмою-завантажувачем у вбудованому програмному забезпеченні плати [25].

1.6 Постановка задачі розробки та дослідження

Після проведеного аналізу предметної області, необхідних компонентів та існуючих рішень, зроблено висновок, що не існує системи розумних жалюзі які будуть контролювати освітленість у різних приміщеннях згідно стандартів освітленості в Україні. У існуючих рішень існує маса недоліків, наприклад, велика ціна, неможливість доробки різними датчиками та сенсорами, підключення до різних існуючих систем «Розумних будинків».

Проведено дослідження щодо обрання відповідних деталей підсистеми, а саме:

- мікроконтролер;
- двигуни;
- драйвер двигунів;
- датчик освітленості;
- джерело живлення.

Обрання деталей проведено після порівняння як мінімум трьох різних варіантів, результати порівняння занесено у таблиці, деталі що були обрані ідеально підходять для вирішення поставленої задачі, та для розвитку проєкту у майбутньому.

Після проведення дослідження та порівняння двох найпопулярніших мов програмування в робототехніці, результати занесено у таблицю. Мовою програмування якою уде писатися основний код став C++. Також було обрано середовище розробки, а саме – ArduinoIDE.

2 РОЗРОБКА МАКЕТУ ПІДСИСТЕМИ АВТОМАТИЧНОГО КОНТРОЛЮ ОСВІТЛЕННЯ У ПРИМІЩЕННІ

2.1 Схема електрична принципова

Основне призначення принципів схем - відображення взаємного зв'язку окремих приладів, засобів автоматизації (СА) та допоміжної апаратури, що входять до складу функціональних вузлів систем автоматизації, а також послідовності їх роботи та принципу дії. Ці схеми відображають дію системи автоматизації, вони необхідні при виробництві налагоджувальних робіт та експлуатації [23]. Ці схеми відбивають принцип дії систем управління, сигналізації, вимірювання та взаємодії між окремими елементами, а також спосіб електроживлення приладів та СА. На важливих електричних схемах в умовному вигляді зображують прилади, апарати, лінії зв'язку між окремими елементами, блоками та модулями цих пристроїв. У випадку принципові електричні схеми повинні містити:

- умовні позначення принципу дії тієї чи іншої
- функціонального вузла системи автоматизації;
- пояснювальні написи;
- частини окремих елементів (приладів, СА, електричних апаратів), що використовуються в інших схемах;
- діаграми перемикачів багатопозиційних контактів
- пристроїв;
- перелік використовуваних у цій схемі приладів, СА, апаратури;
- перелік креслень, що належать до даної схеми, загальні пояснення та примітки.

Принципові електричні схеми систем контролю та управління за призначенням можуть поділятися на схеми управління, технологічного контролю та сигналізації, автоматичного регулювання та живлення.

На рис. 2.1 представлено принципову схему підсистеми автоматичного контролю освітлення у приміщенні.

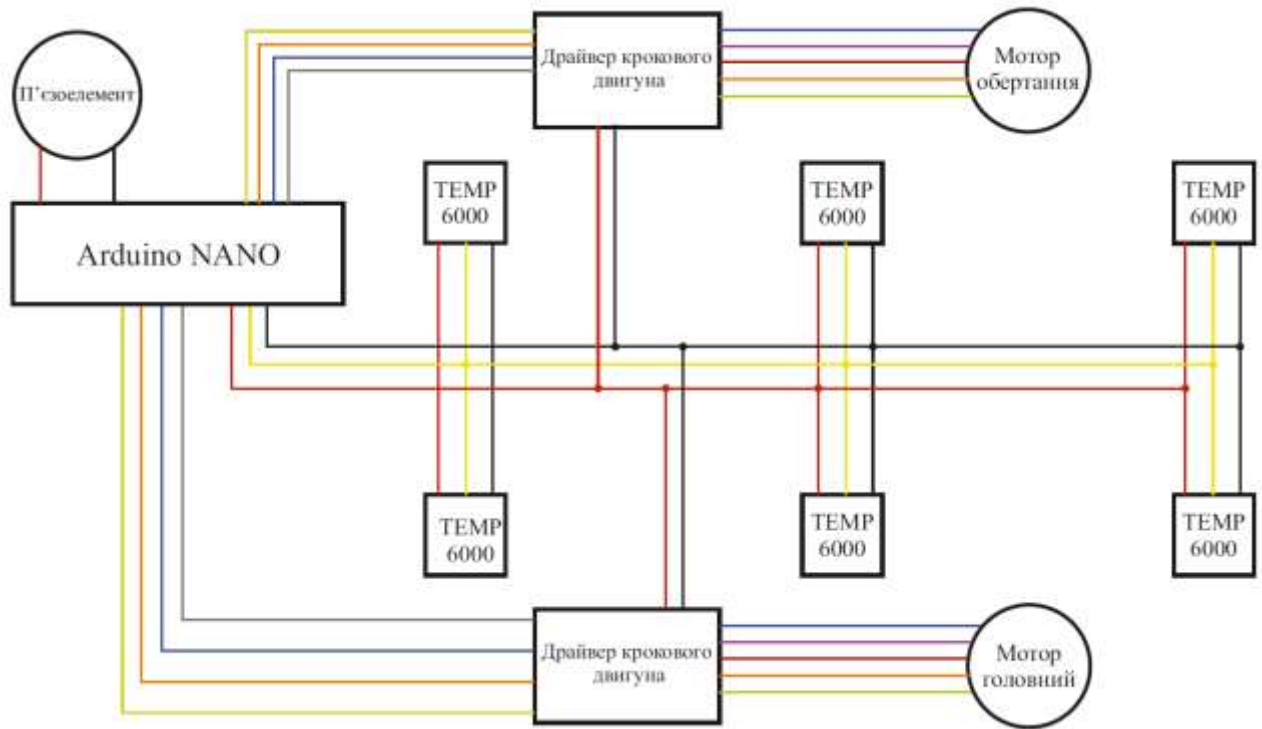


Рисунок 2.1 – Схема електрична принципова підсистеми автоматичного контролю освітлення у приміщенні

На схемі зображені основні елементи підсистеми, а саме:

- Arduino NANO V3.0 – налагоджувальна плата;
- п'єзоелемент – пристрій за допомогою якого будуть створюватись звукові сповіщення;
- TEMP 6000 – датчик освітленості;
- драйвер крокового двигуна;
- мотор обертання жалюзі;
- мотор головний –пристрій за допомогою якого будуть відчинятись або зачинятись жалюзі.

2.2 Прототипування на платформі Tinkercad

Tinkercad – це онлайн-сервіс, який зараз належить мастодонту світу CAD-систем – компанії Autodesk. Tinkercad вже давно відомий багатьом як

просте та безкоштовне середовище для навчання 3D-моделювання. З її допомогою можна досить легко створювати свої моделі та відправляти їх на 3D-друк. Єдиним обмеженням було відсутня місцева локалізація інтерфейсу, зараз ця ситуація виправляється [24].

Нещодавно Tinkercad отримав можливість створення електронних схем і підключення їх до симулятора віртуальної плати Arduino. Ці вкрай важливі і потужні інструменти здатні істотно полегшити розробникам Arduino початківцям процеси навчання, проектування та програмування нових схем.

Список основного функціоналу Tinkercad Circuits:

- онлайн платформа, для роботи не потрібно нічого, крім браузера та стійкого інтернету;
- зручний графічний редактор для візуальної побудови електронних схем;
- набір моделей більшості популярних електронних компонентів, відсортований за типами компонентів;
- симулятор електронних схем, за допомогою якого можна підключити створений віртуальний пристрій до віртуального джерела живлення та простежити, як він працюватиме;
- симулятори датчиків та інструментів зовнішнього впливу. Ви можете змінювати показання датчиків, спостерігаючи, як на них реагує система;
- вбудований редактор Arduino з монітором порту та можливістю покрокової налагодження;
- готові для розгортання проекти Arduino зі схемами та кодом;
- візуальний редактор коду Arduino;
- вбудовані підручники та величезна спільнота з колекцією готових проектів [25].

Великим недоліком платформи є невелика кількість можливих пристроїв, і саме тому на платформі не існують:

- датчик освітленості TEMP6000;
- кроковий двигун 28BYJ-48;

- драйвер крокового двигуна ULN2003.

Але існують їх аналоги, принцип роботи яких співпадають з принципом роботи пристроїв обраних для побудови проєкту, тому було обрано наступні пристрої:

- фоторезистор;
- кроковий двигун;
- Arduino UNO.

Налагоджувальна платформа Arduino UNO є аналогом Arduino NANO, але з покращеними характеристиками, більшими розмірами та з більшою кількістю пінів всіх видів.

На рис. 2.2 представлено остаточну схему макету підсистеми автоматичного контролю освітленості у приміщенні.

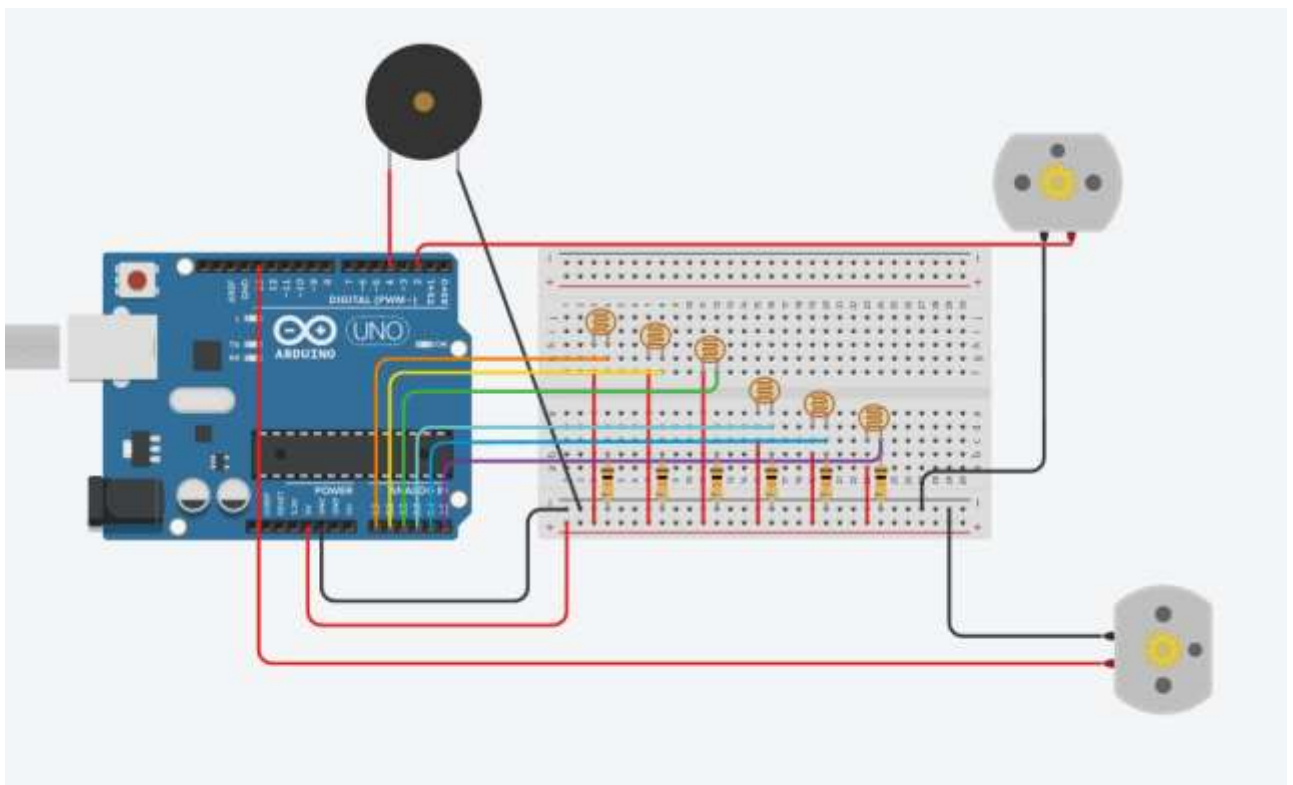


Рисунок 2.2 – Схема підсистеми, створена на платформі Tinkercad

На схемі розміщено такі елементи:

- arduino UNO;
- п'єзоелемент;

- макетну плату;
- шість фоторезисторів;
- шість резисторів;
- два крокових двигуна.

Arduino UNO використовується як головний контролер, який приймає та передає дані та команди до усіх елементів [26]. П'єзоелемент використовується для подання звукових команд про увімкнення підсистеми та сигналу, що свідчить про те що потрібно увімкнути світло у приміщенні. Приєднується до цифрового піна, та до піна заземлення. Макетна плата використовується для кращого використання можливостей налагоджувальної платформи, а саме для розмноження пінів живлення та пінів заземлення [27]. Чотири фоторезистора, які використовуються як аналог датчика освітленості TEMP6000, підключаються до аналогових пінів. Являються системою датчиків які зчитують інформацію про освітленість у кімнаті, та передають цю інформацію на головний контролер. Два фоторезистори являються системою контролю освітленості на вулиці, та передають зчитану інформацію на головний контролер. Два крокові двигуни, один з яких виконує функцію обертання жалюзі, а інший відкриття та зачинення.

2.3 Інструкція створення макету на платформі Tinkercad

Для створення макету на платформі Tinkercad потрібно мати комп'ютер та вихід до мережі.

Першим кроком буде перехід до меню створення схем, що називаються Circuits, далі потрібно натиснути на кнопку Створити схему, відчиниться поле створення схем, що представлено на рис. 2.3.

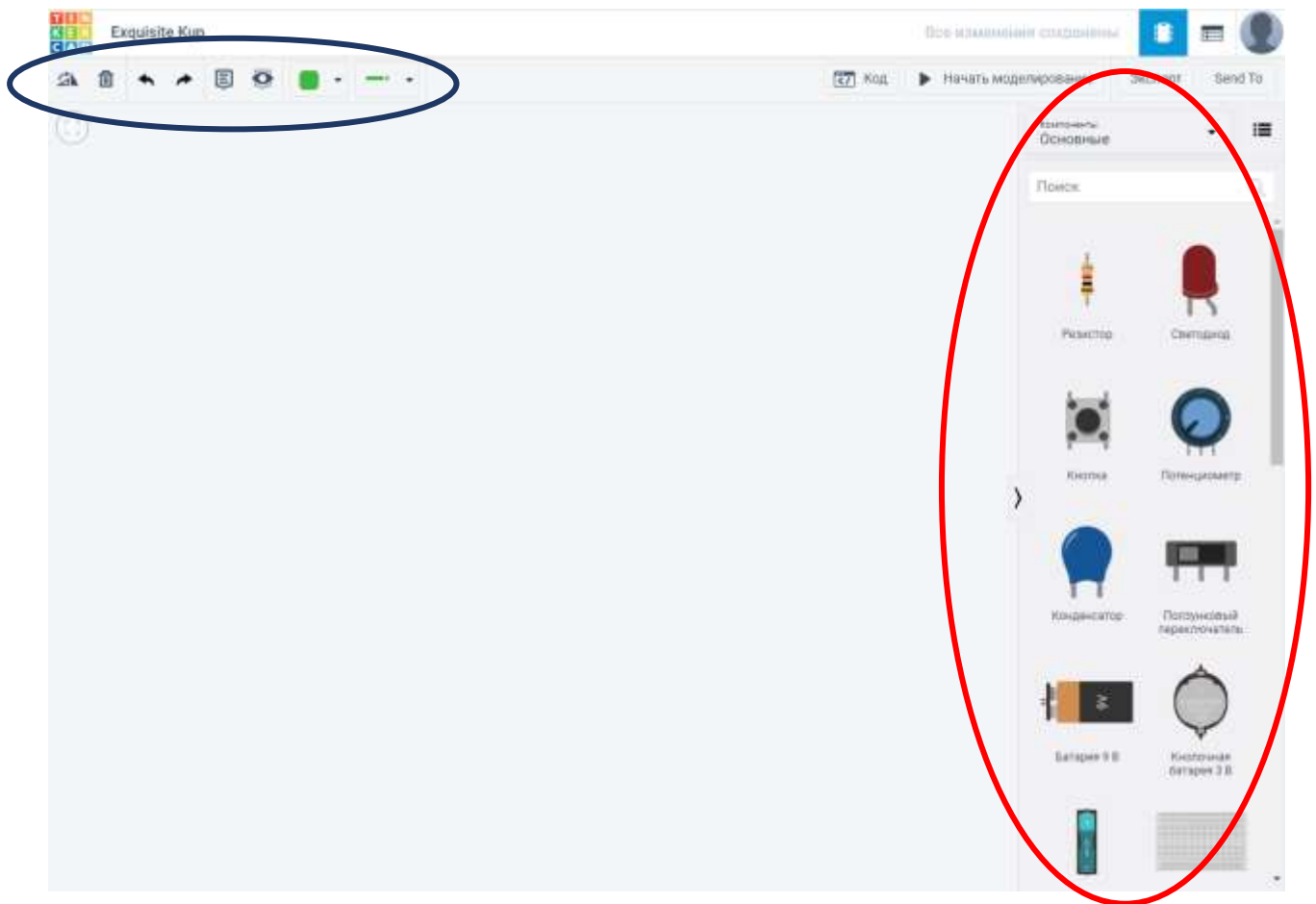


Рисунок 2.3 – Середовище створення схем на платформі Tinkercad

Синім кольором на рис. 2.3 представлено меню управління проектом та компонентами де знаходяться наступні кнопки:

- перевернути компонент;
- видалити компонент;
- відмінити дію;
- повторити дію;
- інструмент примітки;
- перемикання видимості примітки;
- вибір кольору дроту;
- вибір типу дроту.

Червоним кольором на рис. 2.3 представлено меню компонентів проекту. Для того щоб створити компонент треба обрати потрібний мишкою та за допомогою перетягування розмістити компонент на пустому полі зліва від

меню компонентів проекту. Таким чином треба розмістити раніше вказані пристрої, як показано на рис. 2.4.

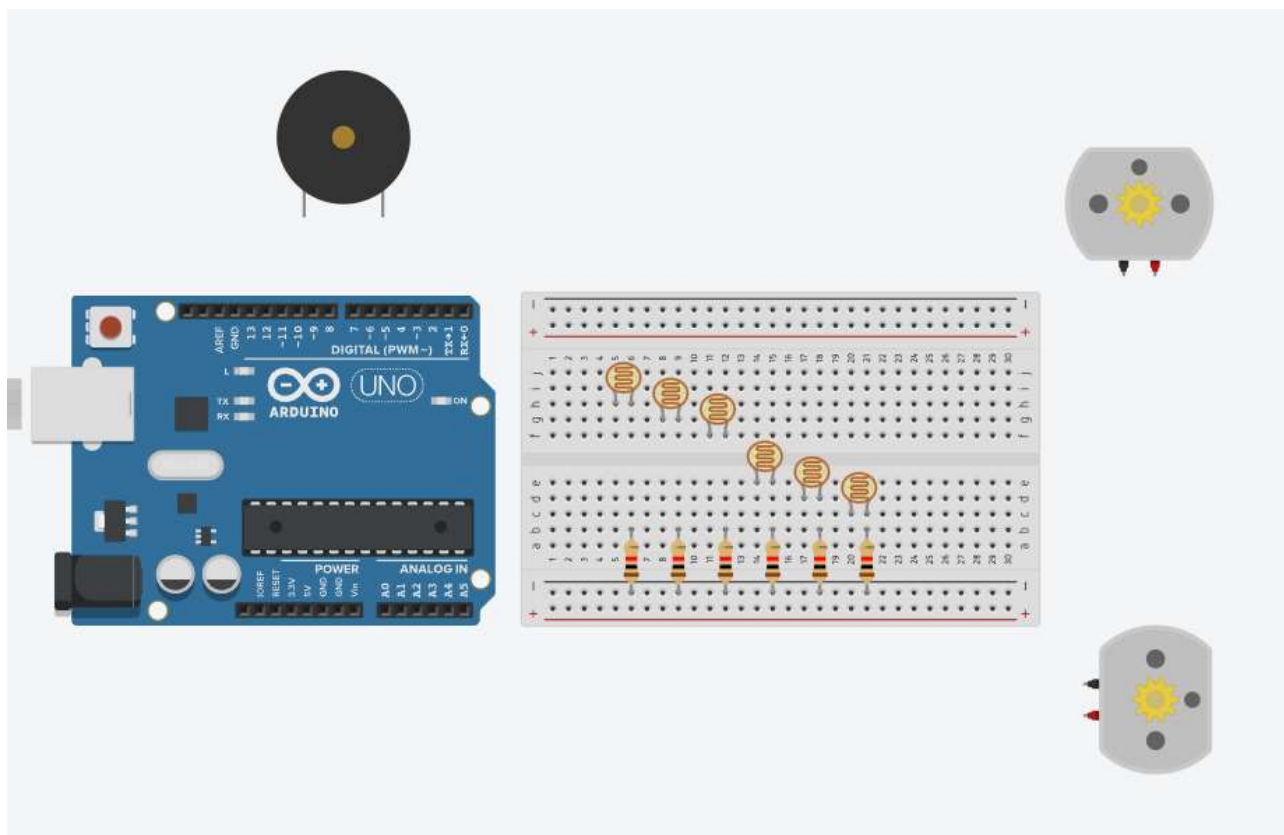


Рисунок 2.4 – Розміщення обраних пристроїв

Після розміщення основних компонентів, потрібно обрати потрібне опір резисторів, що з'єднані з фоторезисторами, для цього необхідно вибрати резистори та у вікні що з'явилося у пункті опір встановити цифру 10, як показано на рис. 2.5.

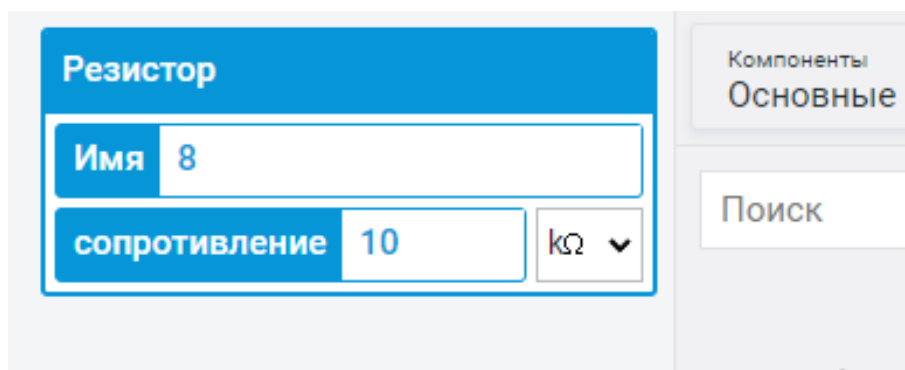


Рисунок 2.5 – Вікно для зміни опору резистора

Наступним кроком буде з'єднання дротами пристроїв з головним контролером. Фоторезистори треба під'єднати до аналогових виходів, це ті виходи біля яких написано літеру А, наприклад, А0. Для підключення п'єзoeлементу необхідно ліву ніжку під'єднати до будь-якого цифрового виходу, а праву ніжку до заземлення. Щоб підключити двигуни необхідно червону ніжку підключити до живлення, що становить 5V, а чорну ніжку до заземлення. Через те, що на обраній налагоджувальній платі вихід, що має потужність 5V лише один, а виходів заземлення лише три, потрібно використовувати макетну плату. Вихід потужністю 5V треба за допомогою дроту під'єднати до строки макетної плати біля якої стоїть знак +, а вихід заземлення під'єднати до строки біля якої стоїть знак -. Щоб створити дріт необхідно обрати необхідний вихід лівою кнопкою миші та приєднати до кінцевого виходу, існує можливість гнути дроти під необхідні кути, для цього потрібно в обраній точці натиснути лівою кнопкою миші та продовжити створювати схему. Також для того, щоб усі дроти не були одного кольору існує можливість змінювати їх кольори, що дуже важливо під час створення макетів, адже колір дроту визначає його функцію, наприклад дроти живлення та заземлення частіше всього червоного та чорного кольору відповідно. Потрібно отримати макет, що вказано на рис. 2.6.

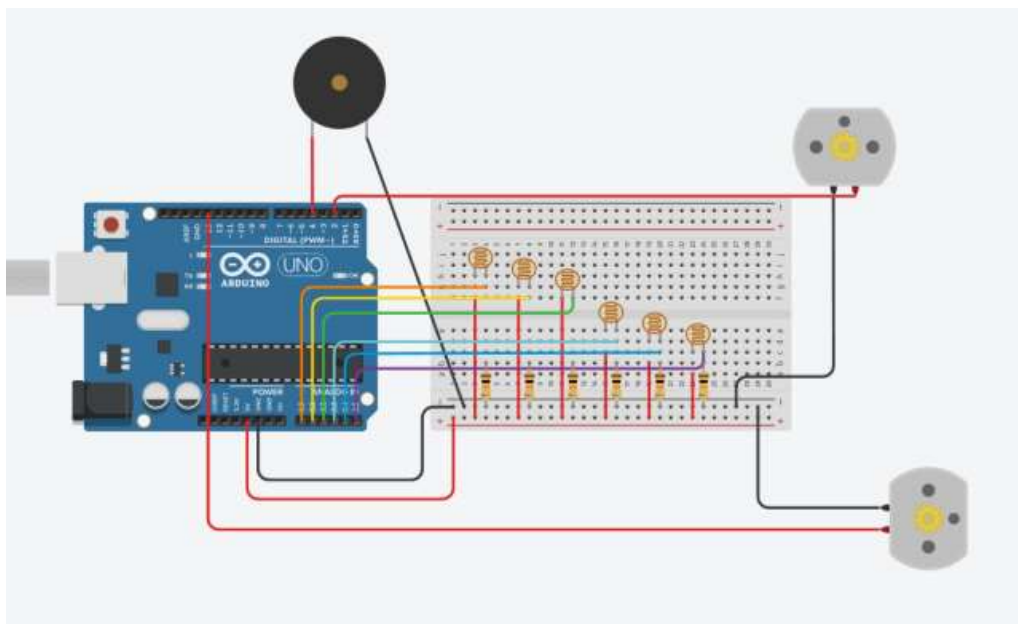


Рисунок 2.6 – Створений макет на платформі Tinkercad

2.4 Інструкція створення робочого макету підсистеми

Для створення робочого макету підсистеми використовуються усі обрані раніше компоненти, а також як і у випадку створення макету на платформі Tinkercad, використовується макетна плата для розширення максимальної кількості пінів. У якості каркасу, що буде являтихь макетом кімнати, використовується коробка з щотного картону габаритні розміри якої складають 400x350x285 мм.

Для початку необхідно зробити квадратний отвір в одній із стінок коробки, що буде виступати в якості вікна.

Наступним кроком буде розміщення датчиків освітленості. Чотири датчика розміщено на внутрішніх стінках коробки, ближче до верхніх кутів, що будуть направлені до центру кімнати, як показано на рис. 2.7.



Рисунок 2.7 – Розміщення датчиків освітленості всередині макету

Далі потрібно розмістити макетну плату, закріпивши її на внутрішній стінці макету, як показано на рис. 2.8.

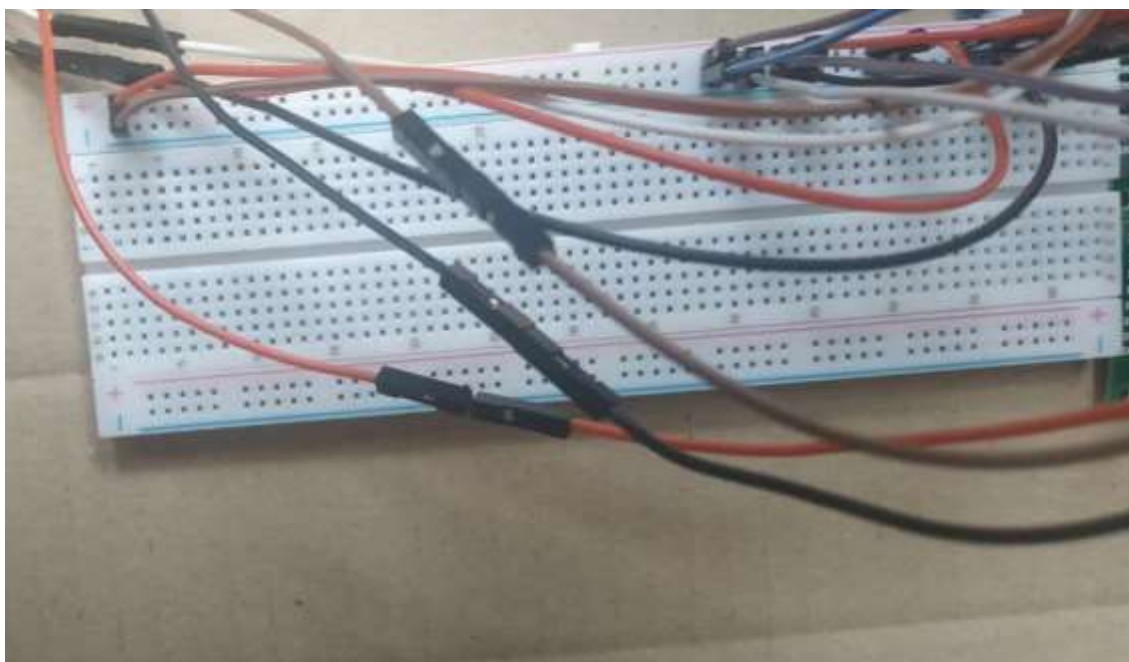


Рисунок 2.8 – Розміщення макетної плати

Наступним кроком буде розміщення головного контролера, тобто Arduino NANO, що треба розмістити на внутрішній стінці макету, над макетною платою, як показано на рис. 2.9.

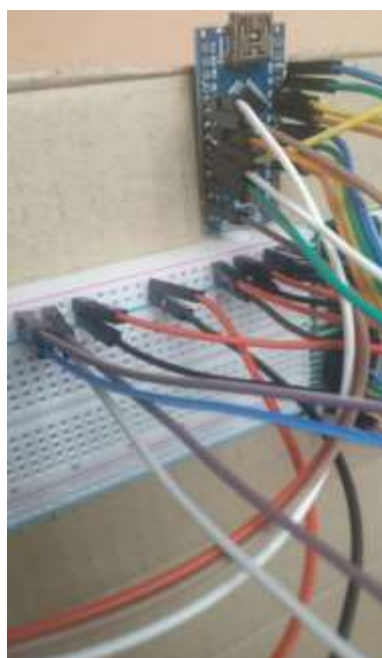


Рисунок 2.9 – Розміщення Arduino NANO

Далі потрібно розмістити драйвери двигунів, як показано на рис. 2.10, необхідно розмістити їх біля макетної плати.

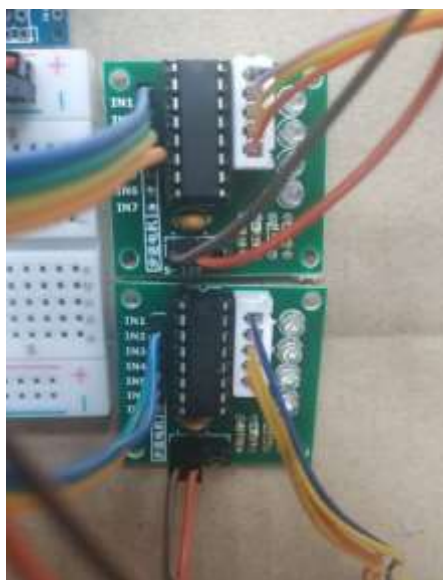


Рисунок 2.10 – Установка драйверів двигунів

Після розміщення драйверів потрібно розмістити самі двигуни, один з яких потрібно розмістити на внутрішній стінці де зроблено імітацію вікна, а інший на сусідній стінці. Двигун, що розміщено біля вікна називається головним двигуном, тому що буде відповідати за відкриття та закриття вікна. Інший двигун буде називатись поворотним, тому що буде відповідати за кут повороту жалюзі. Розміщення двигунів представлено на рис. 2.11.



Рисунок 2.11 – Розміщення двигунів

Після розміщення усіх компонентів підсистеми, треба з'єднати датчики з головним контролером та макетною платою, двигуни з драйверами, а драйвера з макетною платою та головним контролером. З'єднання відбувається за допомогою дротів зі спеціальними клемами, а приєднання дротів до головного контролера виконана за допомогою паяння контактів.

Кінцевий варіант розробленого макету підсистеми автоматичного контролю освітлення у приміщенні представлено на рис. 2.12.

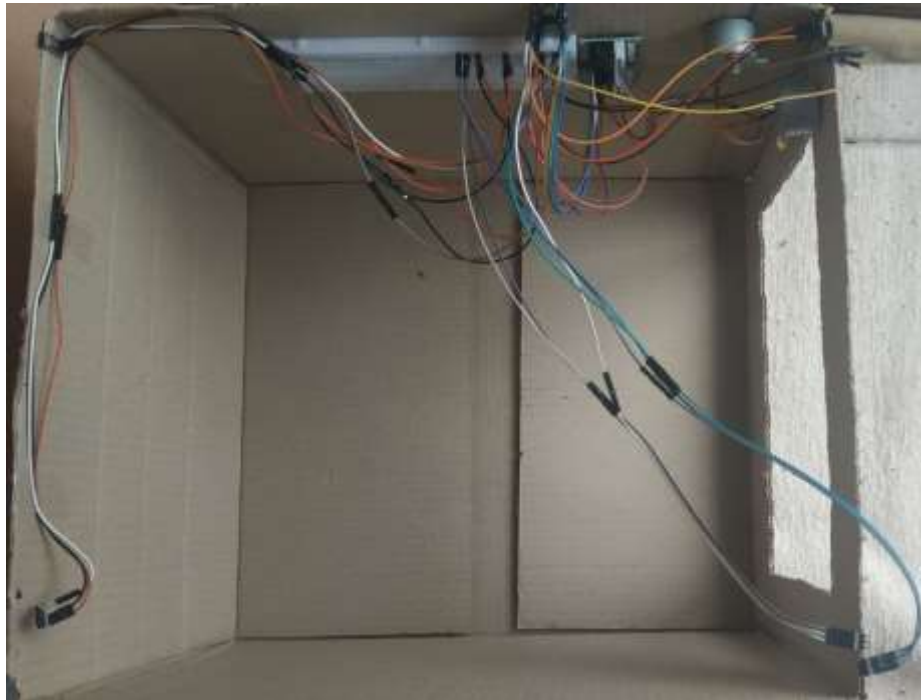


Рисунок 2.12 – Розроблений макет

2.5 Висновок до розділу

У розділі проведено роботу по створенню макетів підсистеми автоматичного контролю освітлення у приміщенні. Розроблено принципову схему підсистеми автоматичного контролю освітлення у приміщенні, за допомогою якої було розроблено віртуальний макет на платформі Tinkercad. Після створення макету за допомогою САПР, розроблено покрокову інструкцію зі створення робочого макету підсистеми.

3 РОЗРОБКА ПРОГРАМНОЇ РЕАЛІЗАЦІЇ ПІДСИСТЕМИ

3.1 Структура програми

На рис. 3.1 представлена структура програмного керування підсистеми автоматичного контролю освітлення у приміщенні. На схемі виділено основні функції які виконуються під час роботи програми.

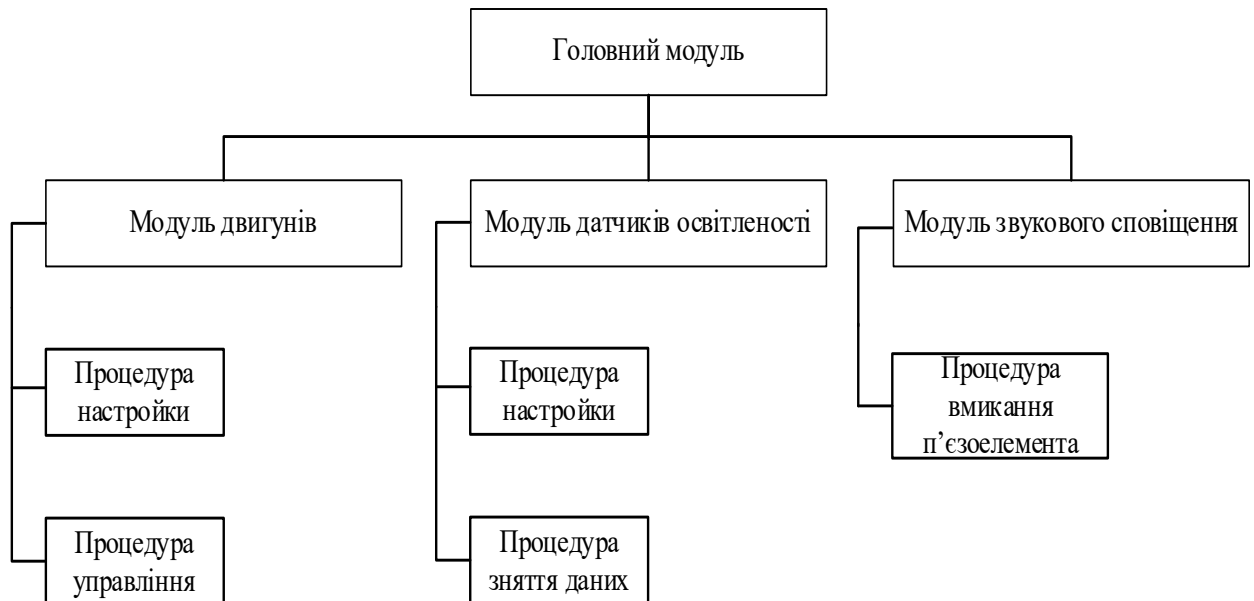


Рисунок 3.1 – Структура програми керування підсистеми

Модуль двигунів – включає в себе процедуру налагоджування під час якої ініціалізуються потрібні виходи. Та процедуру управління під час якої здійснюється зміна сторони обертання роторів моторів.

Модуль датчиків освітленості – включає в себе процедуру налагоджування під час якої ініціалізуються шість датчиків освітленості. Та процедуру зняття даних під час якої знімаються необхідні дані та записуються у пам'ять для подальшої передачі.

Модуль звукового оповіщення – включає в себе процедуру вмикання п'єзоелемента яка сигналізує про увімкнення, або помилку, або про надання додаткової інформації для людини.

3.2 Алгоритм роботи програми

На рис. 3.2 представлено блок-схему алгоритму роботи основної програми.

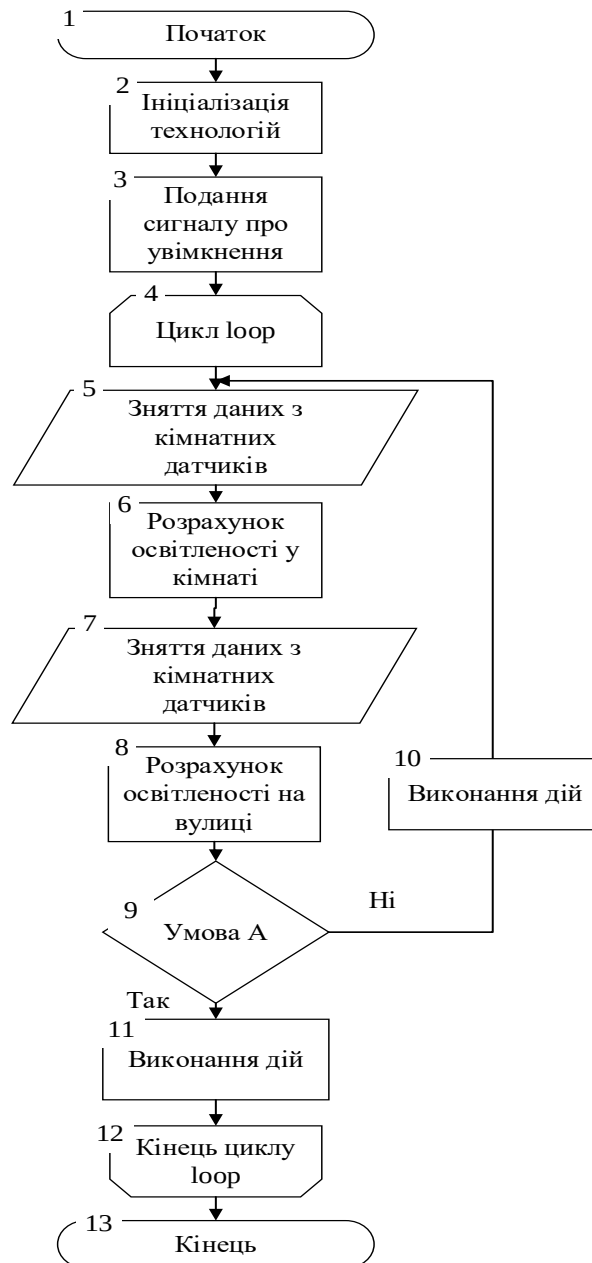


Рисунок 3.2 – Блок-схема алгоритму роботи основної програми

Блок 1. Початок алгоритму.

Блок 2. Ініціалізацій датчиків, двигунів, п'єзоелемента.

Блок 3. Подання сигналу про увімкнення. П'єзoeлемент видасть три короткі сигнали, що будуть сигналізувати про увімкнення пристрою.

Блок 4. Вхід до нескінченного циклу loop, в якому будуть виконуватись основні дії.

Блок 5. Зняття даних з датчиків, які знаходяться у кімнаті.

Блок 6. Розрахунок освітленості у кімнаті виходячи з даних отриманих з датчиків.

Блок 7. Зняття даних з датчиків, які знаходяться на вулиці.

Блок 8. Розрахунок освітленості на вулиці виходячи з даних отриманих з датчиків.

Блок 9. Умова А. Якщо на вулиці освітленість більша ніж ідеальне значення.

Блок 10. Виконання дій. Якщо менша, тоді закриваються вікна та подається сигнал п'єзoeлементом, що свідчить про те, що потрібно ввімкнути світло у кімнаті.

Блок 11. Виконання дій. Якщо більша, тоді вмикаються мотор обертання жалюзі та обертається поки у кімнаті не буде досягнутий рівень освітленості близький до ідеального.

Блок 12. Кінець циклу.

Блок 13. Вихід з програми.

3.3 Розробка коду на платформі Tinkercad

Як вже було вказано раніше, на платформі Tinkercad окрім можливості побудувати схему пристрою, також існує можливість написання коду для побудованої схеми. Код написаний на платформі Tinkercad сумісний з Arduino IDE та використовую в якості мови програмування – мову C++. Також існує можливість написання коду на більш простій блочній мові програмування.

Для того, щоб почати писати код програми на платформі Tinkercad треба відкрити потрібне поле, яке вказано на рис. 3.3.

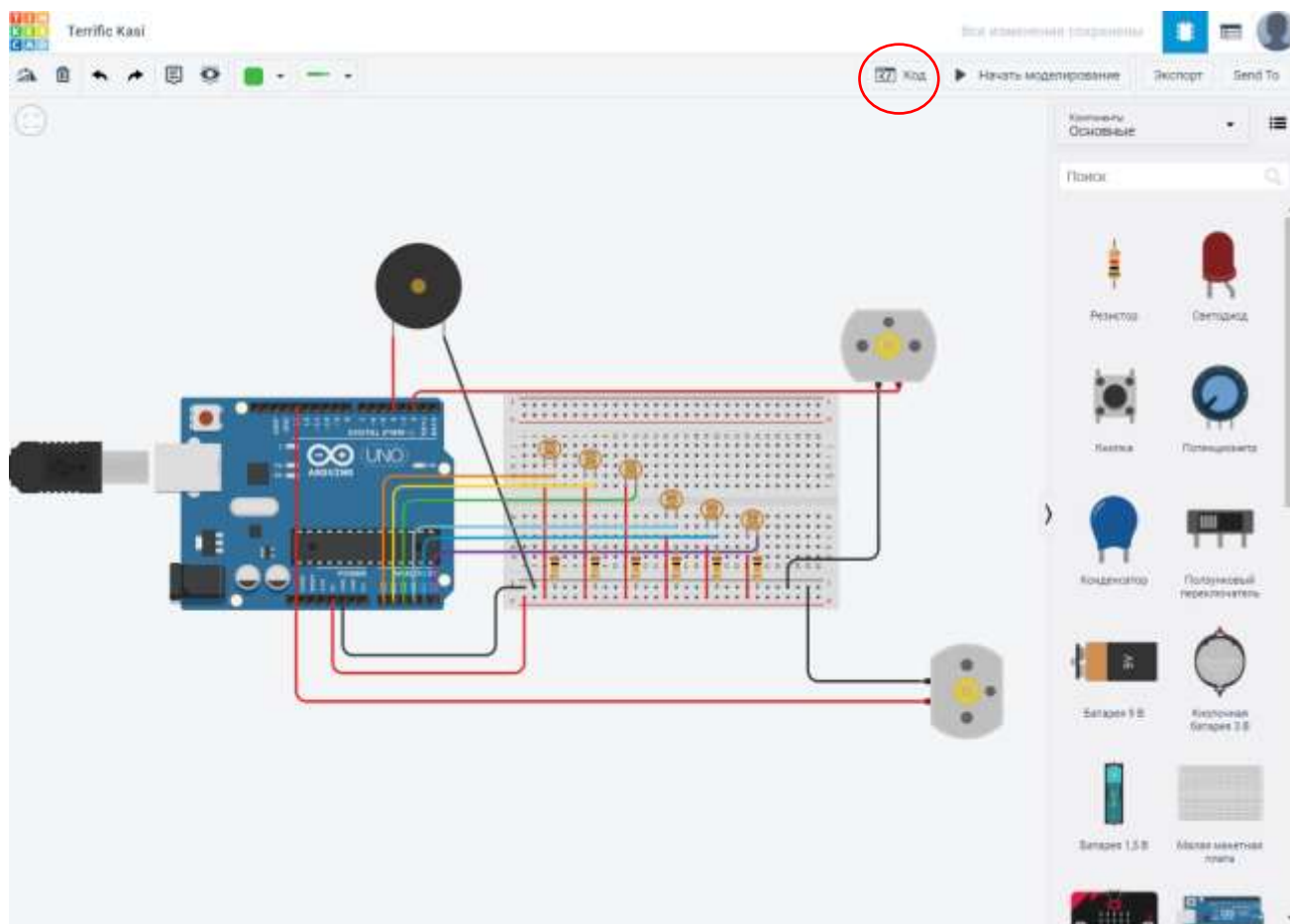


Рисунок 3.3 –Приклад розробки проєктів на платформі Tinkercad

Відчиниться спеціальне поле, яке вказано на рис. 3.4, в якому вже будуть створені дві основні функції, які використовують для програмування мікроконтролерів Arduino. Функція `setup` виконується один раз на початку виконання програми, і частіше всього використовується для одиничного налагоджування будь-яких технологій, наприклад для ініціалізації монітору послідовного інтерфейсу, в обраному проєкті в функції `setup` буде також подача звукового сигналу про увімкнення пристрою. Функція `loop` являється основною функцією для написання коду, тому що являє собою нескінченний цикл під час виконання якого будуть виконуватись запрограмовані команди.

```

1  #include <Servo.h>
2
3  #define ROOM_SENSOR_1      A0
4  #define ROOM_SENSOR_2      A1
5  #define ROOM_SENSOR_3      A2
6  #define ROOM_SENSOR_4      A3
7  #define STREET_SENSOR_1     A4
8  #define STREET_SENSOR_2     A5
9  #define SOUND_PIN          4
10 #define MOTOR_OPENING       13
11 #define MOTOR_ROTATING      2
12
13 enum Command
14 {
15     CMD_NONE = 0,
16     CMD_OPEN = 1,
17     CMD_CLOSE = 2
18 };
19
20 int cmd = CMD_NONE;
21 int GOST_value = 800;
22 float street_light = 0;
23 float room_light = 0;
24
25 Servo srv;
26
27 void setup()
28 {
29     Serial.begin(9600);
30     tone(SOUND_PIN, 500, 100);
31     delay(200);
32     tone(SOUND_PIN, 500, 100);
33     delay(200);
34     tone(SOUND_PIN, 500, 100);
35     delay(200);
36 }
37
38 void loop()
39 {
40     int value_street_sensor_1 = analogRead(STREET_SENSOR_1);
41     int value_street_sensor_2 = analogRead(STREET_SENSOR_2);
42     street_light = (value_street_sensor_1 + value_street_sensor_2);
43
44     int value_room_sensor_1 = analogRead(ROOM_SENSOR_1);
45     int value_room_sensor_2 = analogRead(ROOM_SENSOR_2);
46     int value_room_sensor_3 = analogRead(ROOM_SENSOR_3);
47     int value_room_sensor_4 = analogRead(ROOM_SENSOR_4);
48     room_light = (value_room_sensor_1 + value_room_sensor_2 + value
49
50     //если ГОСТ>на улице тогда передать cmd_CLOSE и подать звуковой с
51     if (GOST_value > street_light && (cmd == CMD_OPEN)){
52
53

```

Монитор последовательного интерфейса

Рисунок 3.4 – Фрагмент розробленого програмного забезпечення на платформі Tinkercad

Функція loop - це те місце, куди поміщені команди, які будуть виконуватися весь час, поки включена плата Arduino. Почавши виконання з першої команди, мікроконтролер дійде до кінця і відразу перестрибне на

початок, щоб повторити ту ж послідовність. І так нескінченна кількість разів (до тих пір, поки на плату буде подано електрику). Найбільш доречний переклад англійського слова loop у разі – це цикл (петля). По своїй суті, void loop – это главная функция, точка входа в нашу программу.

Наступним кроком буде обрання режиму редагування за допомогою кнопки, що вказано на рис. 3.5, потрібно обрати Текст.

Для того, щоб перевірити правильність написання програми використовується кнопка розпочати моделювання, що вказана на рис. 3.5.

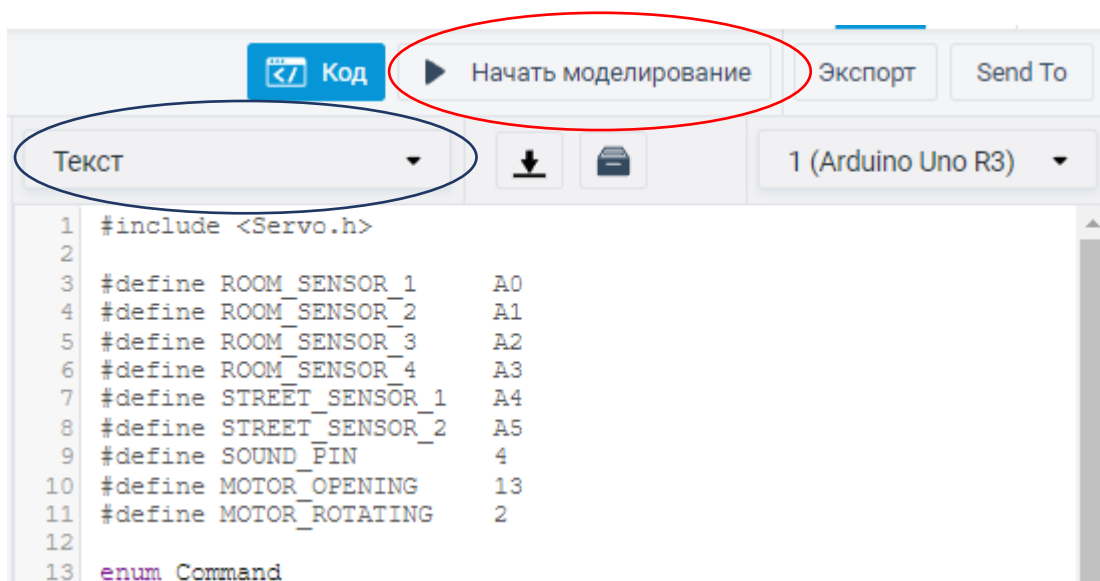


Рисунок 3.5 – Обрання режиму редагування та кнопка перевірки програми

Якщо програма написана без помилок, які не зміг знайти компілятор, тоді кнопка Розпочати моделювання загориться зеленим кольором, як вказано на рис. 3.6, а в середовищі розробки схем будуть відображатись дії, які будуть виконуватись під час виконання програми.

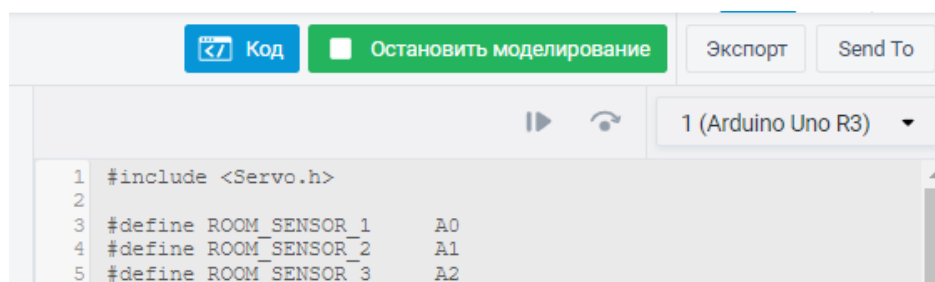


Рисунок 3.6 – Відображення запущеної програми

Якщо компілятор зміг знайти помилку у написаному коді, тоді внизу поля з'явиться вікно з кодом помилки та інформацією що допоможе вирішити цю помилку, приклад вказано на рис. 3.7.

```

1  #include <Servo.h>
2
3  #define ROOM_SENSOR_1      A0
4  #define ROOM_SENSOR_2      A1
5  #define ROOM_SENSOR_3      A2
6  #define ROOM_SENSOR_4      A3
7  #define STREET_SENSOR_1    A4
8  #define STREET_SENSOR_2    A5
9  #define SOUND_PIN          4
10 #define MOTOR_OPENING      13
11 #define MOTOR_ROTATING      2
12
13 enum Command
14 {
15     CMD_NONE = 0,
16     CMD_OPEN = 1,
17     CMD_CLOSE = 2
18 };
19
20 int cmd = CMD_NONE;
21 int GOST_value = 800;
22 float street_light = 0;
23 float room_light = 0;
24
25 Servo srv;ж
26
27 void setup()
28 {
29     Serial.begin(9600);
30     tone(SOUND_PIN, 500, 100);
31     delay(200);
32     tone(SOUND_PIN, 500, 100);
33     delay(200);
34     tone(SOUND_PIN, 500, 100);
35     delay(200);
36 }
37
38 void loop()
39 {
40     int value_street_sensor_1 = analogRead(STREET_SENSOR_1);
41     int value_street_sensor_2 = analogRead(STREET_SENSOR_2);
42     street_light = (value_street_sensor_1 + value_street_sensor_2);
43
44     int value_room_sensor_1 = analogRead(ROOM_SENSOR_1);
45

```

Sorry, it seems like your code has some errors. Close

```

25:11: error: stray '\\328' in program
25:12: error: stray '\\266' in program
exit status 1

```

Рисунок 3.7 – Вікно з кодом помилки що допущена в написанні коду

Під час виконання програми неможливо вносити зміну до коду програми, щоб відредагувати код моделювання потрібно зупинити.

Під час моделювання існує можливість змінювати значення на потрібних датчиках або сенсорах, як вказано на рис. 3.8, також зміни будуть відбуватись самостійно, якщо це передбачено програмою, в обраному проєкті такими змінами являється звук п'єзoeлементу.

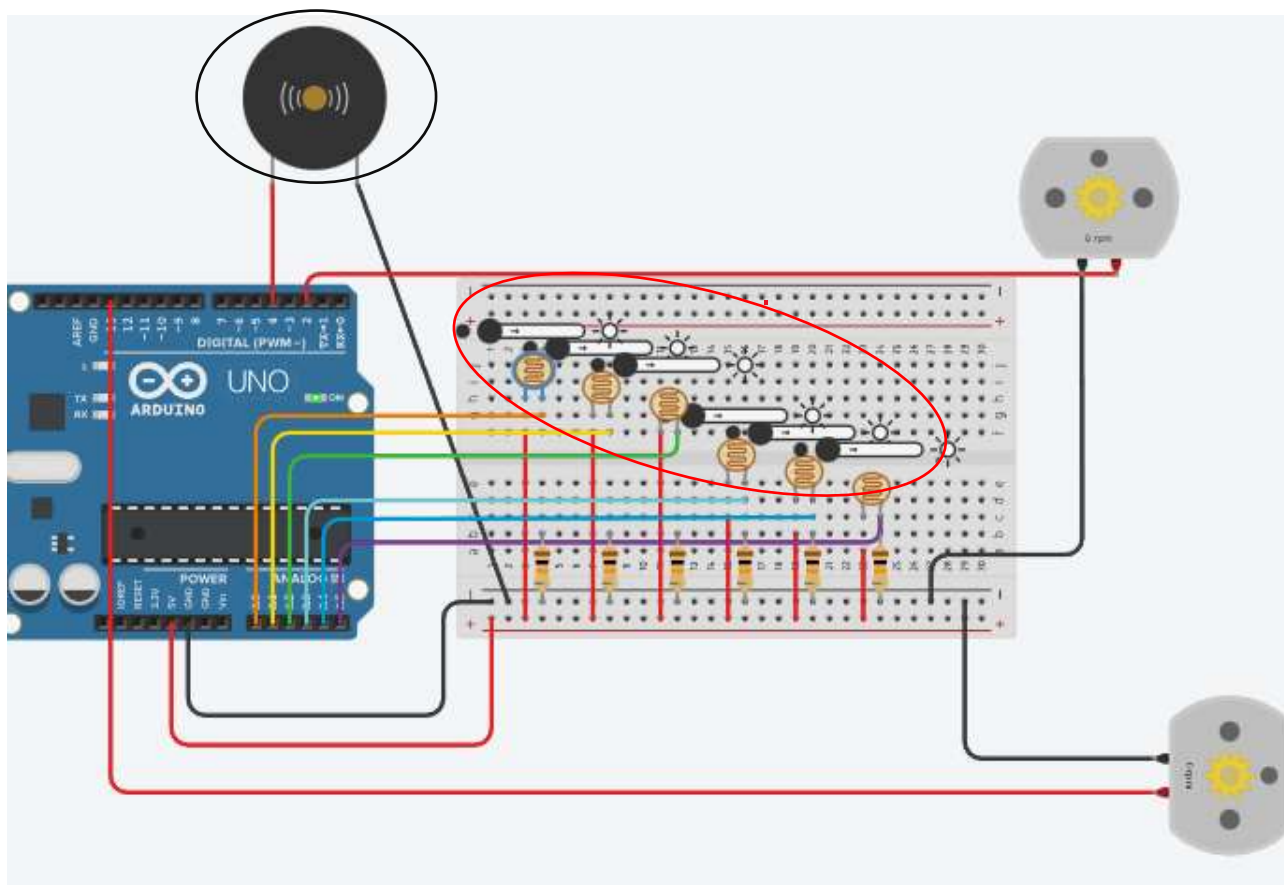


Рисунок 3.8 – Відображення змін на датчиках

Також існує можливість відкрити монітор послідовного інтерфейсу, який представлено на рис. 3.9, за допомогою якого можливо виводу різну інформацію під час виконання програми або передавати команди або інформацію контролеру. Монітор послідовного інтерфейсу являється головним інструментом під час відладки будь-якої програми, бо дає можливість вивести будь-яку інформацію в будь-якому місці виконання програми. Щоб відчинити монітор послідовного інтерфейсу треба натиснути на кнопку, що вказана на рис. 3.9.

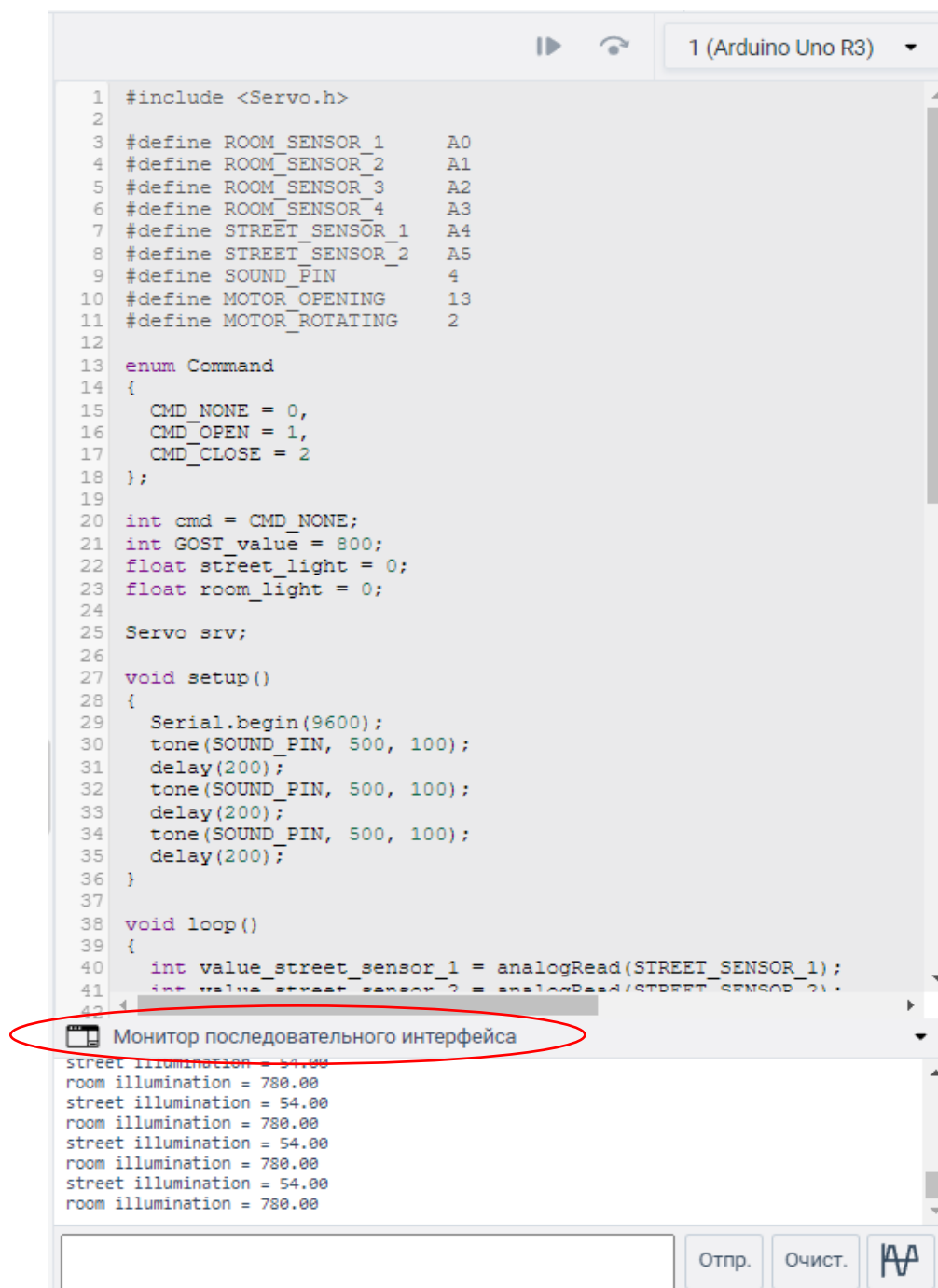


Рисунок 3.9 – Поле відчищеного монітора послідовного інтерфейсу

Для того, щоб стара інформація не заважала відображенню нових повідомлень, існує можливість очистити монітор послідовного інтерфейсу за допомогою кнопки Очистити.

3.4 Розробка програми за допомогою Arduino IDE

Для того, щоб розпочати розробку, необхідно створити новий скетч. Для цього треба обрати у верхньому меню пункт Файл та у випадаючому меню обрати пункт Новий, як вказано на рис. 3.10.

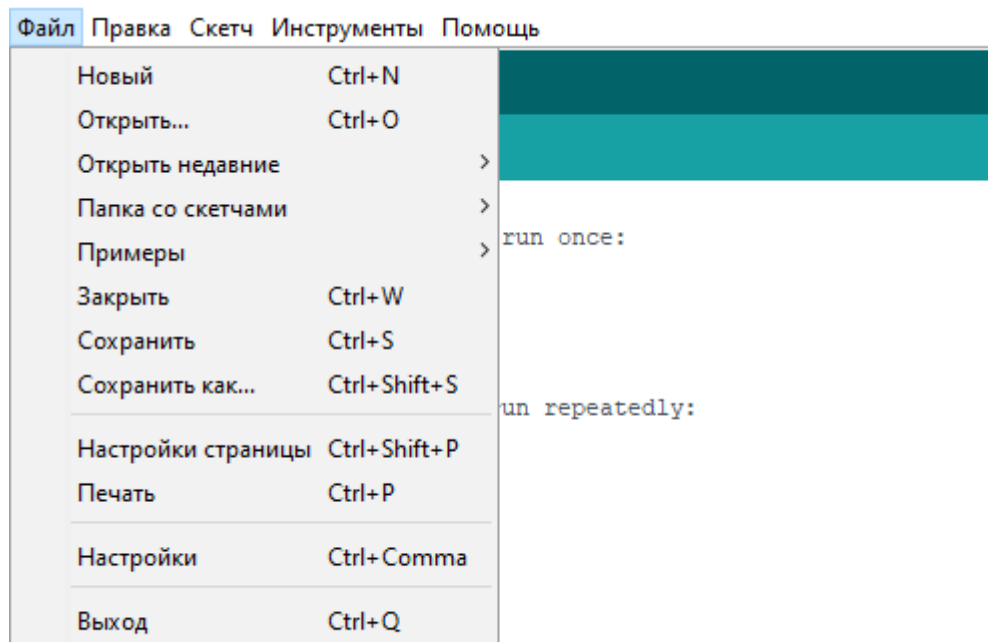


Рисунок 3.10 – Створення нового скетчу в ArduinoIDE

Після чого створиться стандартна програма, яка буде складатись з двох головних функцій, як і на платформі Tinkercad.

ArduinoIDE використовує такі ж функції як і Tinkercad, але володіє і власними, наприклад перевірка правильності написання тексту програми за допомогою кнопки Перевірити, що вказано червоним кольором на рис. 3.11, а також можливість завантаження програми до налагоджувальної платформи за допомогою кнопки завантажити, що вказано жовтим кольором на рис. 3.11.



Рисунок 3.11 – Кнопки перевірки та завантаження програми

Як і на платформі Tinkercad, ArduinoIDE володіє монітором послідовного інтерфейсу, який в ArduinoIDE називається монітором порта, та який вказано на рис. 3.12

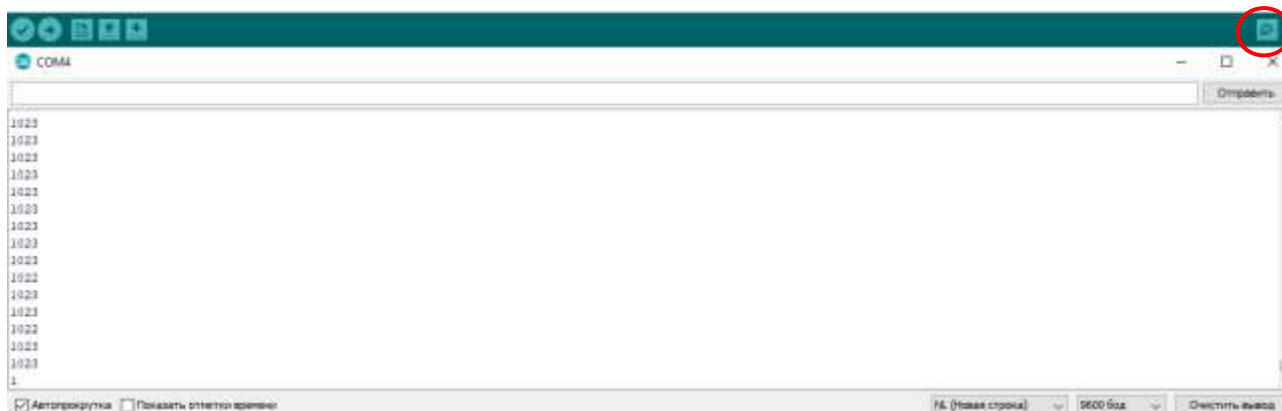


Рисунок 3.12 – Вікно відображення монітора порта

Для того, щоб була можливість завантажити написану програму, потрібно обрати правильну налагоджувальну платформу та послідовний порт до якого підключено платформу, як показано на рис. 3.13, для цього треба у верхньому меню обрати пункт Інструменти та у випадаючому меню обрати пункт плата і обрати ArduinoNano, а в пункт порт обрати COM-порт до якого підключено налагоджувальну платформу.

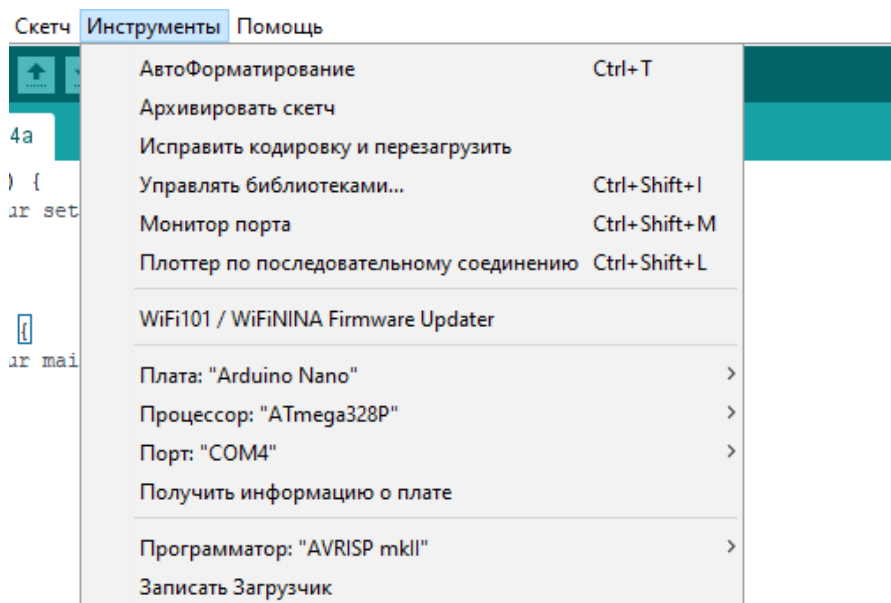


Рисунок 3.13 – Меню Інструменти

Для того, щоб підключити необхідні бібліотеки використовується директива `#include`, що у свою чергу пропонує компілятору увімкнути інший вихідний файл, ім'я якого вказується після директиви. Ім'я файлу заключають в подвійні лапки або `<>`. Наприклад, такі дві директиви вказують компілятору на необхідність підключити заголовний файл стандартної бібліотеки вводу/виводу:

- `#include "stdio.h";`
- `#include <stdio.h>;`

Файли, що підключаються, також можуть мати директиви `#include`. Якщо це має місце, то говорять про вкладені підключення.

Якщо файл, що підключається, вказаний у `<>`, то пошук буде відбуватися в стандартних каталогах, призначених для зберігання заголовних файлів. У випадку, якщо файл, що підключається, укладено в подвійні лапки, пошук буде відбуватися в поточному робочому каталозі. Якщо файл не знайдено, пошук продовжується у стандартних каталогах.

Для того, щоб визначити необхідні константи, наприклад, назви необхідних значень, назви виходів, сенсорів та датчиків використовується директива `#define`, що у свою чергу визначає ідентифікатор і послідовність символів, якою буде заміщатися ідентифікатор при його виявленні в тексті програми. Ідентифікатор також називається іменем макросу, а процес заміщення називається підстановкою макросу. Стандартний вид директиви наступний:

```
#define ім'я_макросу послідовність_символів
```

У цьому операторі відсутня точка з комою. Між ідентифікатором та послідовністю символів може бути будь-яка кількість прогалін. Макрос завершується лише переходом на новий рядок.

Дуже часто макроси використовують для визначення магічних чисел, що використовуються в програмі. Наприклад, програма може визначати масив та мати кілька процедур для роботи з ним. Замість того, щоб жорстко кодувати розмір масиву, краще визначити макрос, що відповідає розміру масиву, і використовувати його в тих місцях, де потрібне використання розміру. Таким

чином, якщо необхідно змінити розмір масиву, єдине, що потрібно зробити, це змінити оператор `#define` і перекомпілювати програму. Скрізь, де використовувався цей макрос, відбудуться автоматичні зміни [29].

Для того, щоб визначити команди підсистеми, що будуть передаватись до двигунів та п'єзоелемента використовується такий інструмент як перерахування який у свою чергу є набором іменованих цілих констант, що визначає всі допустимі значення, які може приймати змінна. Перерахування визначаються за допомогою ключового слова `enum`, яке вказує на початок типу, що перераховується. Стандартний вид перерахувань наступний:

`enum` ярлик { список перерахувань } список змінних;

Як ім'я перерахування – ярлик, і список змінних необов'язкові, але з них має бути присутнім. Список перерахувань - це розділений комами список ідентифікаторів. Як і структурах, ярлик використовується для оголошення змінних цього типу. Важливо зрозуміти, що у перерахуваннях кожному символу ставиться у відповідність ціле чисельне значення і тому перерахування можуть використовуватись у будь-яких цілочисельних виразах.

Можна визначити значення одного або кількох символів, використовуючи ініціалізатор. Це робиться шляхом приміщення за символом знака рівності та цілісного значення. При використанні ініціалізатора символи, що йдуть за ініціалізаційним значенням, отримують значення більше, ніж зазначене перед цим.

Головним інструментом для написання коду є функції `setup` та `loop`, але також існує можливість створювати нові функції, що взагалі є базовими блоками, у яких виконуються всі операції.

Специфікатор типу визначає тип значення, що повертається функцією за допомогою оператора `return`. Це може бути будь-який допустимий тип. Якщо тип не вказано, передбачається, що функція повертає цілі. Список параметрів - це розділений комами список змінних, що отримує значення аргументів під час виклику функції. Функція може бути без налаштувань і в такому випадку список параметрів містить ключове слово `void`.

Такожодним з головних інструментом являється конструкція `if-else-if`, що використовується для порівняння визначених змінних та констант та обрання подальших дій в залежності від результату порівняння. Умови обчислюються згори донизу. Коли виявляється справжнє умова, то виконується оператор, що з цією умовою, а решта конструкції ігнорується. Якщо не знайдено жодної справжньої умови, виконується оператор, який відповідає останньому `else`. Останній оператор `else` часто відіграє роль оператора, що виконується за умовчанням, тобто якщо всі умови помилкові, то виконується оператор, відповідний останньому `else`. Якщо останній оператор `else` відсутній, то не виконується жодних дій у разі неправдивості всіх умов.

Оператор `default` виконується, якщо не знайдено відповідності, `default` необов'язковий і, якщо його немає, то у разі відсутності збігів нічого не відбувається. Коли виявляється збіг, оператори, асоційовані з відповідним `case`, виконуються доти, доки не зустрінеться оператор `break`. У випадку `default` (або останнього `case`, якщо немає `default`), оператор `switch` закінчує роботу при виявленні кінця.

Слід знати про три важливі моменти оператора `switch`:

- `switch` відрізняється від `if` тим, що він може виконувати тільки операції перевірки суворої рівності, у той час як `if` може обчислювати логічні вирази та стосунки;
- не може бути двох констант в одному операторі `switch`, що мають однакові значення. Звичайно, оператор `switch`, що включає інший оператор `switch`, може містити аналогічні константи;
- якщо в операторі `switch` використовуються символьні константи, вони автоматично перетворюються на цілі.

З технічного погляду оператори `break` є необов'язковими в операторі `switch`. Вони використовуються для закінчення роботи послідовності операторів, асоційованих із цією константою. Якщо оператор `break` відсутній, продовжують виконувати оператори наступного розділу, доки не буде досягнуто оператора `break` або кінця оператора `switch`. Про константи вибору можна думати як про мітки. Виконання починається з мітки, що відповідає

шуканому значенню, і продовжується, доки не буде досягнуто break або кінець оператора switch [24].

Switch відрізняється від if тим, що він може виконувати тільки операції перевірки суворої рівності, у той час як if може обчислювати логічні вирази та стосунки/

Не може бути двох констант в одному операторі switch, що мають однакові значення. Звичайно, оператор switch, що включає інший оператор switch, може містити аналогічні константи [25].

Якщо в операторі switch використовуються символічні константи, вони автоматично перетворюються на цілі.

3.5 Висновки до розділу

У розділі представлено структуру програми, в якій показано головні модулі та процедури. Розроблено алгоритм роботи програми та представлено у вигляді блок-схеми. Описано процес створення програм на платформі Tinkercad та за допомогою ArduinoIDE. Описано головні принципи створення програм, та використані інструменти.

4 ТЕСТУВАННЯ СИСТЕМИ ТА КЕРІВНИЦТВО КОРИСТУВАЧА

4.1 Тестування системи

Тестування програми відбувалось після реалізації віртуального макету підсистеми на платформі Tinkercad. Перші версії програми були написані та протестовані на цій платформі, за допомогою монітора послідовного порта та зібраного макету, як показано на рис. 4.1.

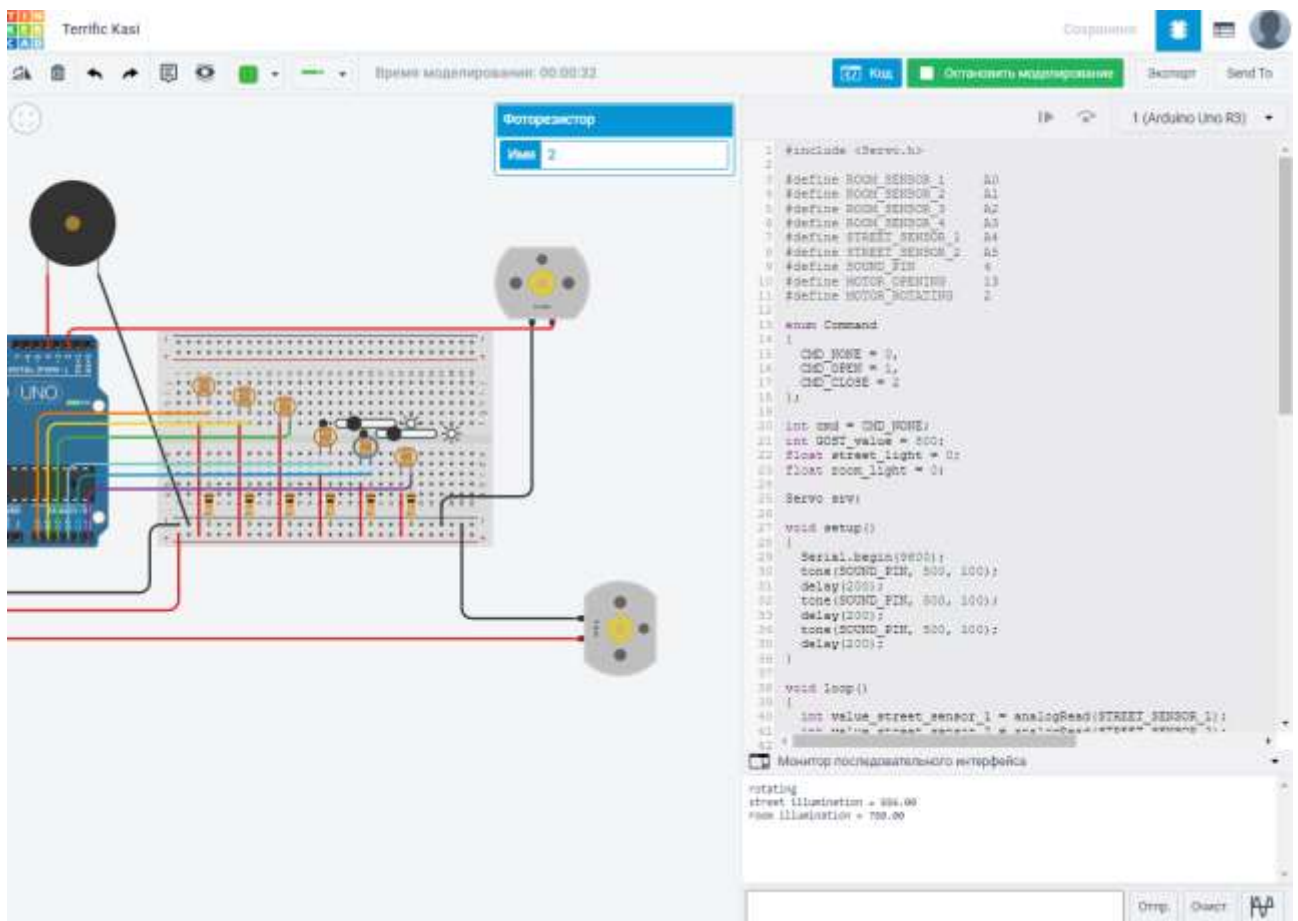


Рисунок 4.1 – Тестування програми на платформі Tinkercad

Також на платформі є можливість виводи графіки залежності будь-якої величини від часу. Тому доцільно виводи залежність яскравості освітлення кімнати від часу, але через те що на платформі вручну міняються значення на фоторезисторах графіки не можуть бути ідеальними, а можуть бути тільки наближені до реальних умов. Приклад графіку залежності яскравості вулиці від часу приведено на рис. 4.2.



Рисунок 4.2 – Залежність освітленості вулиці від часу на платформі Tinkercad

Після вдалих тестів до удосконалення програми на платформі Tinkercad, наступним кроком буде тестування реального макету програма до якого розроблена за допомогою ArduinoIDE. В ArduinoIDE також існує можливість виводи графіки залежності, але завдяки робочого макету графіку будуть приближені до ідеальних, бо макет тестується у реальних умовах експлуатації, графіки залежності робочого макету представлені на рис. 4.3.

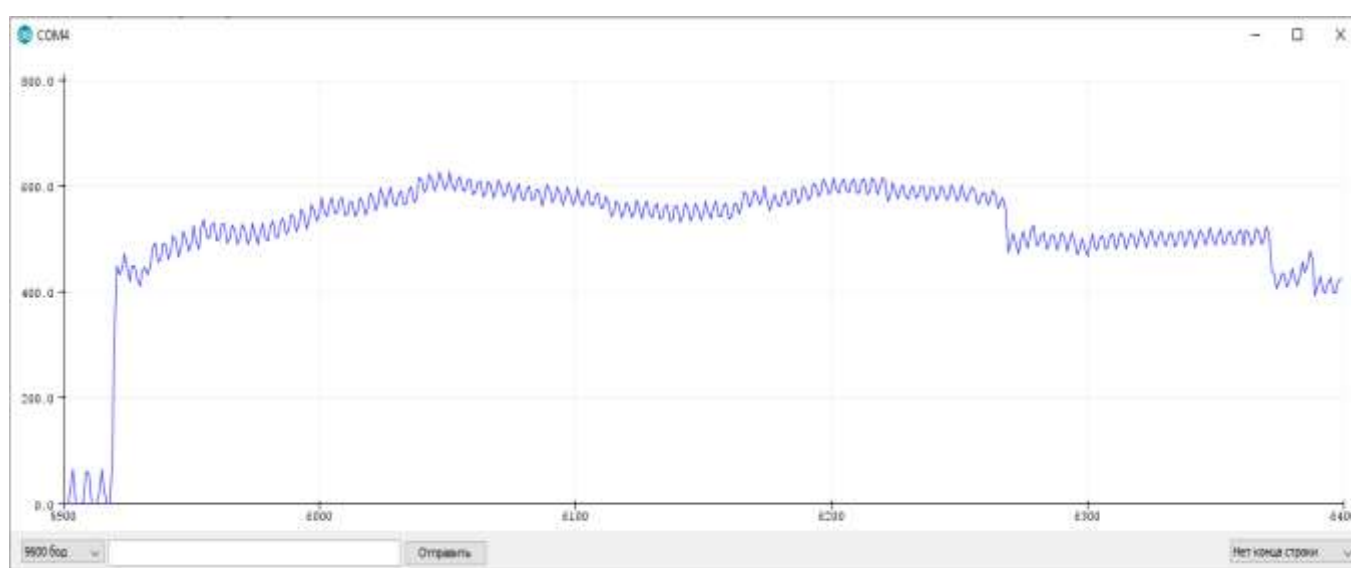


Рисунок 4.3 – Графік залежності робочого макету

Тестування останньої версії програми також показало недоліки системи, перш за все це неможливість користуванням пультом дистанційного керування, відсутність третього двигуна ускладнює систему закриття та відкриття вікон, якщо система буде використовуватись у великих приміщеннях кількість датчиків освітлення треба збільшувати.

4.2 Керівництво користувача

Для наглядного відображення роботи системи розроблено UMLдіаграму, що представлено на рис. 4.4.

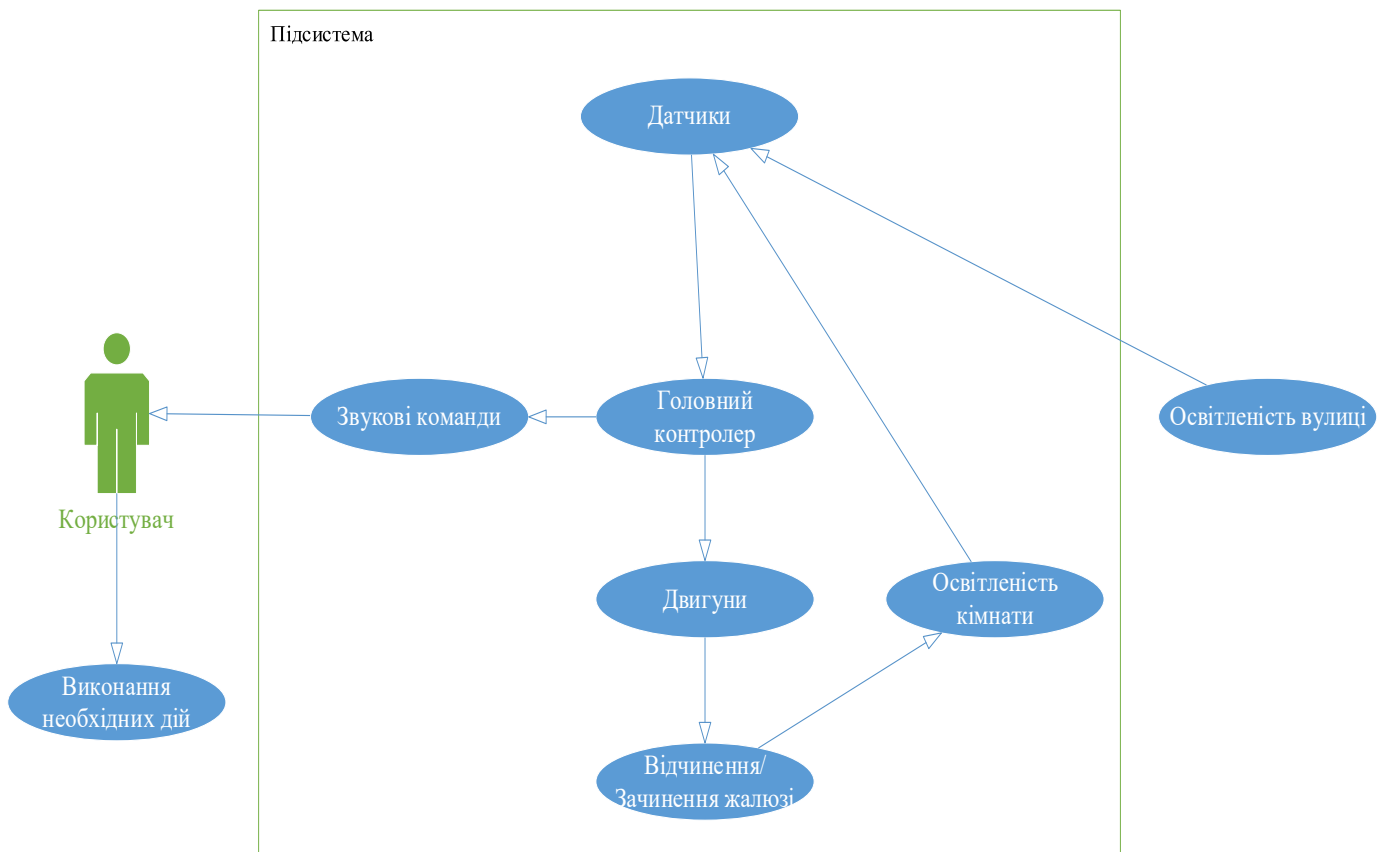


Рисунок 4.4 – UMLдіаграма роботи системи

Для роботи необхідно забезпечити підсистему живленням, для цього, якщо є можливість підключитись до системи живлення будівлі, в іншому

випадку забезпечити живлення підсистеми можливо через акумулятори необхідної потужності.

Робота підсистеми починається з звукової команди, що свідчить про увімкнення та успішне налаштування усіх модулів.

Через датчики освітленості, що знаходяться на вулиці біля вікна робочого приміщення, головний контролер підсистеми отримує інформацію про освітленість вулиці і в залежності від цих показників відбуваються необхідні дії. Отримані дані порівнюються з ідеальним значенням, що було отримано після аналізу державних стандартів України. Це значення може змінюватись в залежності від типу приміщення де працює підсистема.

Якщо освітленість вулиці менша ніж ідеальне значення тоді відбувається зачинення вікна, передачею відповідних команд до двигунів головним контролером, та роздається звукова команда, що свідчить про те, що користувачу потрібно увімкнути світло.

Якщо освітленість вулиці більша ніж ідеальне значення тоді головний контролер передає команди до двигунів, які в свою чергу корегують кут нахилу жалюзі для того, щоб освітленість в кімнаті стала наближена до ідеального.

Через датчики що розміщені у робочому приміщенні, бажано щоб ці датчики знаходились в кутах кімнати та були направлені до центру кімнати, головний контролер отримує дані про освітленість кімнати та порівнює їх з ідеальним значенням.

Задачі користувача полягають у тому, щоб слухати звукові команди підсистеми. Після подання команди про увімкнення світла у користувача є час, що залежить від типу робочого приміщення, для увімкнення світла. Також існує команда, що свідчить про помилку в роботі системи, після подання такої команди потрібно, за можливістю відімкнути систему від живлення, та викликати інженера підсистеми, що встановить причину помилки та розробить план дій для вирішення встановлених проблем.

ВИСНОВКИ

Проведено дослідження щодо обрання популярного мікроконтролера для побудови обраної підсистеми, та обрання налагоджувальної плати малих розмірів.

Також приведено список потрібних деталей та порівняння декількох представників кожного пункту, на основі цих досліджень було створено таблиці порівняння та обрано найкращі рішення.

Обрано мову програмування шляхом порівняння за допомогою написання простих програм, на основі обраної мови програмування було відібрано середовище розробки коду.

Розроблено принципову схему підсистеми за якою було створено спочатку макет за допомогою САПР, а після створено робочий макет. Після чого було розроблено структуру програми та приведено блок-схему алгоритму роботи, описано як створювати програми на платформі Tinkercad та у середовищі розробки ArduinoIDE, описано основні інструменти для написання робочої версії програми.

За результатами роботи було розроблено детальне керівництво користувача, створено віртуальний та робочий макет підсистеми, та протестовано написане програмне забезпечення.

Усі поставлені задачі виконані, мета роботи досягнена.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Why Are There Still So Many Jobs? The History and Future of Workplace Automation [Electronic resource]. – Access mode : <https://www.aeaweb.org/articles?id=10.1257/jep.29.3.3>
2. Robot Tactile Sensing [Electronic resource]. – Access mode : <https://www.frontiersin.org/articles/10.3389/fnbot.2019.00008/full>
3. Капустин, Н. М. Автоматизация производственных процессов в машиностроении [Текст] / Н.М. Капустин // Варфел. – 2004. – С. 325-330.
4. Юревич, Е.И. Основы робототехники [Текст] / Е.И. Юревич, В.Ф. Миткевич // БХВ-Петербург. – 2005. – № 2. – С. 287-295.
5. РОБОТОТЕХНИКА [Электрон. ресурс]. – Режим доступа : <https://exponenta.ru/robotics>
6. Порівняння можливостей CAD [Електрон. ресурс]. – Режим доступу : <https://web.archive.org/web/20110712223138/http://www.gstarcad.ru/comparison#GstarCAD>
7. Макаров, И.М. Робототехника: История и перспективы [Текст] / И.М. Макаров, Ю. И. Топчеев // МАИ. – 2003. – С. 134-140 с.
8. Брага, Н.А. Создание роботов в домашних условиях [Текст] / Н.А. Брага // Пресс. – 2007. – С. 59-68.
9. Зенкевич, С.Л. Основы управления манипуляционными роботами [Текст] / С.Л. Зенкевич, А.С Ющенко // МГТ. – 2004. – № 2. – 45-48 с.
10. Глушков, В.А. Енциклопедія кібернетики [Текст] / В. А. Глушков // Либідь. – 2007. – 467-469 с.
11. AVR [Электрон. ресурс]. – Режим доступа : <https://www.microchip.com/>
12. ARM [Электрон. ресурс]. – Режим доступа : <https://www.arm.com/>
13. Офіційна сторінка MSP430 на сайті Texas Instruments [Електрон. ресурс]. – Режим доступу : <http://www.ti.com/microcontrollers/msp430-ultra-low-power-mcus/overview.html>

14. Нормы освещенности в Украине [Электрон. ресурс]. – Режим доступа : <https://sunpower.ua/a132993-normy-osveschennosti-ukraine.html>
15. Lighting of indoor workplaces [Electronic resource]. – Access mode : <https://www.fagerhult.com/knowledge-hub/EN-12464-1/>
16. Офіційна сторінка Arduino [Електрон. ресурс]. – Режим доступу : <https://www.arduino.cc/education>
17. Электроприводы [Электрон. ресурс]. – Режим доступа : <http://wiki.amperka.ru/projects:irjalousie>
18. Шаговые двигатели [Электрон. ресурс]. – Режим доступа : <https://arduinomaster.ru/motor-dvigatel-privod/shagovye-dvigateli-i-motory-arduino/>
19. Датчики освещенности [Электрон. ресурс]. – Режим доступа : <https://arduinomaster.ru/datchiki-arduino/photorezistor-arduino-datchik-sveta/>
20. Офіційна сторінка ST [Електрон. ресурс]. – Режим доступу : https://www.st.com/content/st_com/en/support/learning/stm32-education.html
21. Офіційна сторінка Texas Instruments [Електрон. ресурс]. – Режим доступу : https://studylib.ru/doc/180038/special._nye-seminary-texas-instruments-dlya-universitetov-po
22. Літій-іонний акумулятор [Електрон. ресурс]. – Режим доступу : https://uk.wikipedia.org/wiki/Літій-іонний_акумулятор
23. Батарейка «Крона» [Електрон. ресурс]. – Режим доступу : https://uk.wikipedia.org/wiki/Батарейка_«Крона»
24. Программирование на C [Электрон. ресурс]. – Режим доступа : <http://www.c-cpp.ru/>
25. Приемы программирования МК [Электрон. ресурс]. – Режим доступа : <http://myowndevise.ru/index.php/theory/item/37-priemy-programmirovaniya-mk>
26. Разработка принципиальных схем [Электрон. ресурс]. – Режим доступа : <https://books.ifmo.ru/file/pdf/1051.pdf>
27. Про роботов и робототехнику [Электрон. ресурс]. – Режим доступа : <https://robotics.ua/shows>
28. TinkercadArduino – онлайн симулятор ардуино [Электрон. ресурс]. –

Режим доступа : https://arduinomaster.ru/program/simulyator-arduino-tinkercad-circuits/#_Tinkercad__Arduino

29. Домашние роботы [Электрон. ресурс]. – Режим доступа : <http://robotrends.ru/robopedia/domashnie-roboty>

30. Макетная плата для монтажа [Электрон. ресурс]. – Режим доступа : <https://arduinomaster.ru/platy-arduino/maketnaya-plata-arduino>

31. Отображение данных Arduino в виде графика [Электрон. ресурс]. – Режим доступа : <https://voltiq.ru/data-visualization-with-sfmonitor/>

32. Типы памяти микроконтроллеров [Электрон. ресурс]. – Режим доступа : <http://csaa.ru/tipy-pamjati-mikrokontrollerov-pamjat-programm/>

ДОДАТОК А
Текст програми

A.1 Код C++ для порівняння

```

#include<iostream>
#include <chrono>
#include <vector>
using namespace std;

auto start = chrono::high_resolution_clock::now();

vector<int>::const_iterator binarySearch(const vector<int>& container, int
element)
{
    const auto endIt = end(container);
    auto left = begin(container);
    auto right = endIt;

    if (container.size() == 0
        || container.front() > element
        || container.back() < element) {
        return endIt;
    }

    while (distance(left, right) > 0) {
        const auto mid = left + distance(left, right)/2;
        if (element <= *mid)
            right = mid;
        else
            left = mid + 1;
    }

    if (*right == element)

```

```
        return right;

    return endIt;
}

int main()
{
    const vector<int>vec = { 1,2,3,4,5,6,7,8,9,10 };
    const int element = 12;
    auto foundIt = binarySearch(vec, element);
    if (foundIt != vec.end())
    cout<< *foundIt<<endl;

    auto end = chrono::high_resolution_clock::now();
    chrono::duration<float> duration = end - start;
    cout<< "Duration = " <<duration.count() <<endl;

    return 0;

}
```

A.2 Код Python для порівняння

```
import time
start_time = time.time()
def binarySearch(lst, x):
    l = 0
    r = len(lst)
    while r - l > 1:
        m = (l + r) // 2
        if x < lst[m]:
            r = m
        else:
            l = m
    return l if lst[l] == x else None

def main():
    lst = [0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
    x = 8
    print(binarySearch(lst, x))

if __name__ == "__main__":
    main()
    print ("Duration = " (time.time() - start_time))
```

A.3 Код програми

```
#include <Servo.h>

#define ROOM_SENSOR_1  A0
#define ROOM_SENSOR_2  A1
#define ROOM_SENSOR_3  A2
#define ROOM_SENSOR_4  A3
#define STREET_SENSOR_1 A4
#define STREET_SENSOR_2 A5
#define SOUND_PIN      4
#define MOTOR_OPENING  13
#define MOTOR_ROTATING 2

enum Command
{
    CMD_NONE = 0,
    CMD_OPEN = 1,
    CMD_CLOSE = 2
};

int cmd = CMD_NONE;
int GOST_value = 800;
float street_light = 0;
float room_light = 0;

Servo srv;

void setup()
{
    Serial.begin(9600);
```

```

tone(SOUND_PIN, 500, 100);
delay(200);
tone(SOUND_PIN, 500, 100);
delay(200);
tone(SOUND_PIN, 500, 100);
delay(200);
}

```

```

void loop()
{
    int value_street_sensor_1 = analogRead(STREET_SENSOR_1);
    int value_street_sensor_2 = analogRead(STREET_SENSOR_2);
    street_light = (value_street_sensor_1 + value_street_sensor_2)/2;
    Serial.println(street_light);

```

```

    int value_room_sensor_1 = analogRead(ROOM_SENSOR_1);
    int value_room_sensor_2 = analogRead(ROOM_SENSOR_2);
    int value_room_sensor_3 = analogRead(ROOM_SENSOR_3);
    int value_room_sensor_4 = analogRead(ROOM_SENSOR_4);
    room_light = (value_room_sensor_1 + value_room_sensor_2 +
value_room_sensor_3 + value_room_sensor_4)/4;
    room_light = map(room_light, 0, 1023, 0, 500);
    Serial.println(room_light);
    delay(100);

```

//если ГОСТ>на улице тогда передать cmd_CLOSE и подать звуковой сигнал о включение света

```

if (GOST_value > street_light && (cmd == CMD_OPEN)){
    srv.attach(MOTOR_OPENING);
    srv.write(1000);

```

```

    srv.detach();
    tone(SOUND_PIN, 500, 2000);
    cmd = CMD_CLOSE;

    //иначе на улице>в комнате тогда вращать пока в комнате не будет +-100
= ГОСТelse
    } else if(street_light > room_light){
        do{
            srv.attach(MOTOR_ROTATING);
            srv.write(30);
            srv.detach();
            Serial.println("rotating");
            Serial.print("street illumination = ");
                Serial.println(street_light);
                Serial.print("room illumination = ");
                Serial.println(room_light);
            value_street_sensor_1 = analogRead(STREET_SENSOR_1);
            value_street_sensor_2 = analogRead(STREET_SENSOR_2);
            street_light = (value_street_sensor_1 + value_street_sensor_2)/2;

            value_room_sensor_1 = analogRead(ROOM_SENSOR_1);
            value_room_sensor_2 = analogRead(ROOM_SENSOR_2);
            value_room_sensor_3 = analogRead(ROOM_SENSOR_3);
            value_room_sensor_4 = analogRead(ROOM_SENSOR_4);
            room_light = (value_room_sensor_1 + value_room_sensor_2 +
value_room_sensor_3 + value_room_sensor_4)/4;
            delay(10000);
        } while ((room_light >= GOST_value-100) || (room_light <=
GOST_value+100));
    }

    if (street_light < room_light && (cmd == CMD_OPEN || cmd ==
CMD_NONE)) {

```

```

    srv.attach(MOTOR_ROTATING);
    srv.write(120);
    // ждём указанное время открытия/закрытие жалюзей
    delay(2000);

    srv.detach();
    cmd = CMD_CLOSE;
    } else if (street_light > room_light && (cmd == CMD_CLOSE || cmd ==
CMD_NONE)) {
        srv.attach(MOTOR_OPENING);
        srv.write(30);
        delay(2000);

        srv.detach();
        cmd = CMD_OPEN;
        if (irrecv.decode(&results)) {

            irrecv.resume();

            codeV = results.value;
            codeL = results.bits;

            if (codeL >= 16) {
                if (codeV == codeOpen && cmd != CMD_CLOSE) {
                    cmd = CMD_OPEN;
                    cmdStartT = millis();

                    start();
                } else if (codeV == codeClose && cmd != CMD_OPEN) {

```

```

        cmd = CMD_CLOSE;
        cmdStartT = millis();

        start();
    }
} else if (codeL == 0 && (cmd == CMD_OPEN || cmd == CMD_CLOSE))
{
    cmdStartT = millis();
}

delay(100);
}

if (cmd == CMD_OPEN || cmd == CMD_CLOSE) {
    if (millis() - cmdStartT > 500) {
        cmd = CMD_NONE;
        stop();
        delay(1000);
    }
}
}
}
}
}

```

ДОДАТОК Б
Слайди презентації

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «ЗАПОРІЗЬКА ПОЛІТЕХНІКА»

Дипломна кваліфікаційна робота
на тему:

ДОСЛІДЖЕННЯ ТА ПРОГРАМНА РЕАЛІЗАЦІЯ ПІДСИСТЕМИ КОМПЛЕКСУ
АВТОМАТИЧНОГО КОНТРОЛЮ ОСВІТЛЕННЯ У ПРИМІЩЕННІ
RESEARCH AND SOFTWARE IMPLEMENTATION OF THE AUTOMATIC CONTROL
SUBSYSTEM FOR ROOM LIGHTING CONTROL

Виконав:

ст. гр. КНТ - 116

Д.І. Зіненко

Керівник:

к.т.н., доцент

І.Я. Зеленьова

1

Рисунок Б.1 – Слайд 1

Різновид прототипування у робототехниці



2

Рисунок Б.2 – Слайд 2

Таблиця порівняння налагоджувальних плат

Назва платформи	Arduino Nano 3	STM32F103C8T6	MSP430F529
Мікроконтролер	ATmega328P	ARM32 cortex M3	MSP430
Робоча напруга	5В	5В	5В
Напруга живлення (рекомендована)	7-12В	2-3,6 В	1,8-3,6 В
Цифрові входи / виходи	14(6 - ШІМ)	15	66
Аналогові входи	8	15	-
Flash-пам'ять	32 КБ	64 КБ	128 КБ
SRAM	2 КБ	20 КБ (RAM)	8 КБ(RAM)
Тактова частота	16 МГц	72 МГц	25 МГц
Ціна	170 грн	276 грн	500 грн

3

Рисунок Б.3 – Слайд 3

Стандарти освітлення в Україні

Освітлювані об'єкти	Середня горизонтальна освітленість, лк
Класні кімнати, кабінети, аудиторії загальноосвітніх шкіл, шкіл інтернатів, середньоспеціальних та професійнотехнічних установ, лабораторії, навчальні кабінети фізики, хімії, біології та інші	500
Аудиторії, навчальні кабінети, лабораторії технікумів та вищих навчальних закладів	400
Кабінети інформатики та обчислювальної техніки	200
Навчальні кабінети технічного креслення та малювання	500
Лаборантські при навчальних кабінетах	400
Лабораторії органічної та неорганічної хімії, препараторські	400
Майстерні з обробки металів та деревини	300
Інструментальна кімната майстра інструктора	300
Кабінети обслуговуючих видів праці	400
Спортивні зали	200
Снарядні, інвентарні, господарські комори	50
Криті басейни	150
Актові зали, кіноаудиторії	200
Естради актових залів, кабінети та кімнати викладачів	300
Рекреації	150

4

Рисунок Б.4 – Слайд 4

Мета та Завдання роботи

Мета роботи – реалізувати проєкт підсистеми комплексу автоматичного контролю освітлення у приміщенні, розробити програмні, апаратні рішення та інструкції з використання технічних компонентів.

Задачі:

- аналіз мікроконтролерів ;
- аналіз ринку деталей;
- аналіз готових рішень ;
- аналіз мови програмування;
- аналіз середовища розробки;
- розробка готових рішень ;
- розробка інструкцій .

5

Рисунок Б.5 – Слайд 5

Аналіз ринку деталей				Назва	Драйвер крокового двигуна ULN2003	Драйвер двигунів двоканальний на TB6612FNG	Драйвер двох двигунів на L298N
Назва	Кроковий двигун 28BY-48	Сервопривід SG90	Двигун з редуктором	Напруга живлення двигуна	5/12 В	2.5/15 В	5/35 В
Напруга живлення	5 В	3,5 – 5 В	6-8 В	Напруга управління	5 В	5 В	5 В
Кут обертання	360°	180°	360°	Робочий струм драйвера	2 А	2 А	2 А
Розміри	39 x 29 x 28 мм	33 x 30 x 13 мм	64x20x20 мм	Розміри	35 x 33 x 10 мм	21 x 18 x 3 мм	43,5 x 43,2 x 29,4 мм
Крутний момент	34,3 мН / м	196,1 мН / м	78,4 мН / м	Вага	15 г	10 г	26 г
Ціна	39 грн	51 грн	39 грн	Мікросхема драйвера	ULN2003	TB6612FNG	L298N
				Ціна	22 грн	67 грн	62 грн

Назва	Датчик освітленості GY-49 MAX44009	Датчик освітленості цифровий GY-302 BH1750FVI	Датчик освітленості TEMT6000	Назва	«Крона»	Samsung ICR18650-22P
Тип датчику	Фототранзистор	Фоторезистор	Фототранзистор	Розміри	48,5 мм × 26,5 мм × 17,5 мм	65 мм × 17 мм
Напруга живлення	3,3 – 5 В	3 – 5 В	5 В	Маса	53 гр	40 гр
Інтерфейс підключення	I2C	I2C	Аналоговий	Напруга	9 В	4.2 В
Діапазон освітленості	0,045 – 188000 Лк	0-65535 Лк	0,045 – 188000 Лк	Електроємність	625 мА/ч	2200 мА/ч
Розміри	20 x 10 мм	14 x 18,5 мм	14 x 7 мм	Повторна зарядка	-	+
Ціна	176 грн	36 грн	34 грн	Ціна	12 грн	70 грн

Рисунок Б.6 – Слайд 6

Обрання мови програмування та середовища розробки

C++



Рисунок 2.1 – Результати виконання програми для C++

PYTHON



Рисунок 2.2 – Результати виконання програми для Python

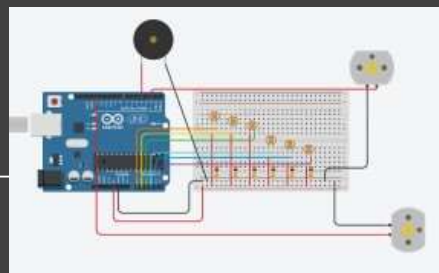
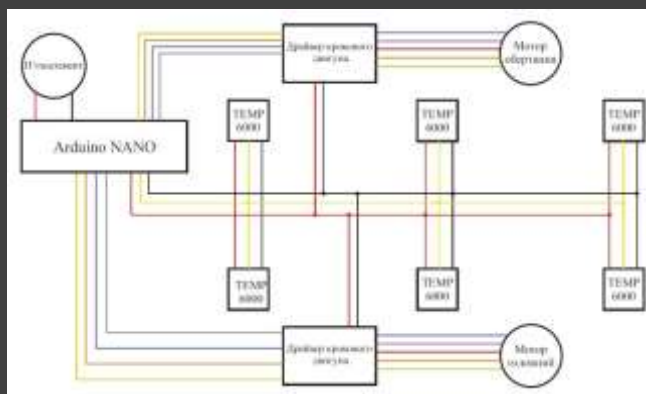
- НАЙПОПУЛЯРНИШИМ РІШЕННЯМ ДЛЯ РОЗРОБКИ ПРОГРАМ ДЛЯ МІКРОКОНТРОЛЕРІВ ARDUINO ЯВЛЯЄТЬСЯ ІНТЕГРОВАНЕ СЕРЕДОВИЩЕ РОЗРОБКИ ARDUINO IDE

- ПІДТРИМУЄ МОВИ С ТА C++ З ВИКОРИСТАННЯМ СПЕЦІАЛЬНИХ ПРАВИЛ ФОРМУВАННЯ КОДУ.

7

Рисунок Б.7 – Слайд 7

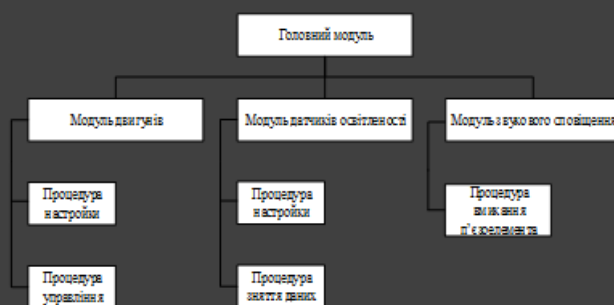
Розробка макетів



8

Рисунок Б.8 – Слайд 8

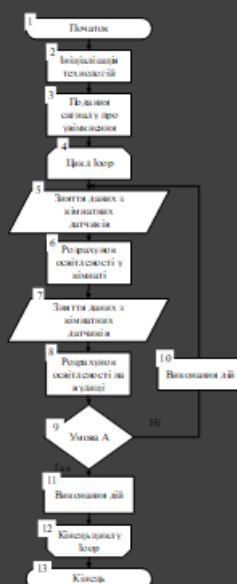
Структура програми



9

Рисунок Б.9 – Слайд 9

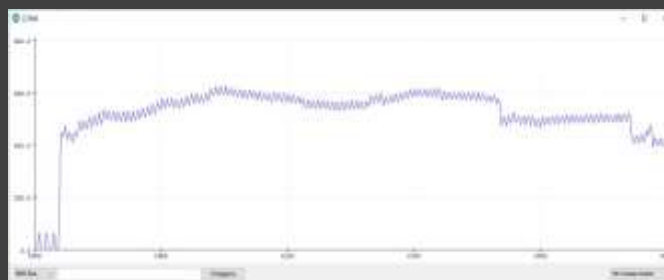
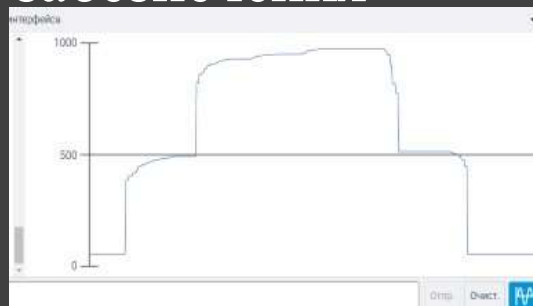
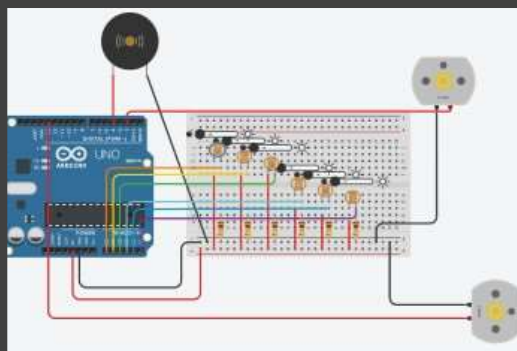
Блок-схема алгоритма роботи програми



10

Рисунок Б.10 – Слайд 10

Тестування програмного забезпечення



11

Рисунок Б.11 – Слайд 11

Висновки

Проведено дослідження щодо обрання популярного мікроконтролера для побудови обраної підсистеми, та обрання налагоджувальної плати малих розмірів.

Також приведено список потрібних деталей та порівняння декількох представників кожного пункту, на основі цих досліджень було створено таблиці порівняння та обрано найкращі рішення.

Обрано мову програмування шляхом порівняння за допомогою написання простих програм, на основі обраної мови програмування було відібрано середовище розробки коду.

Розроблено принципову схему підсистеми за якою було створено спочатку макет за допомогою САПР, а після створено робочий макет. Після чого було розроблено структуру програми та приведено блок-схему алгоритму роботи, описано як створювати програми на платформі Tinkercad та у середовищі розробки Arduino IDE, описано основні інструменти для написання робочої версії програми.

За результатами роботи було розроблено детальне керівництво користувача, створено робочий макет підсистеми, та протестовано написане програмне забезпечення.

Усі поставлені задачі виконані, мета роботи досягнена.

12

Рисунок Б.12 – Слайд 12