МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ПРИВАТНЕ АКЦІОНЕРНЕ ТОВАРИСТВО «ПРИВАТНИЙ ВИЩИЙ НАВЧАЛЬНИЙ ЗАКЛАД «ЗАПОРІЗЬКИЙ ІНСТИТУТ ЕКОНОМІКИ ТА ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ»

Кафедра інформаційних технологій

ДО ЗАХИСТУ ДОПУЩЕНА

Зав.кафедрою _____

д.е.н., доц. Левицький С.І.

КВАЛІФІКАЦІЙНА БАКАЛАВРСЬКА РОБОТА

РОЗРОБКА МІКРОПРОЦЕСОРНОЇ СИСТЕМИ РЕГУЛЮВАННЯ ДОРОЖНЬОГО РУХУ НА ПЕРЕХРЕСТІ

Виконав
ст. гр. КІ-119 _____ К.О. Попова

Керівник
професор _____ С.О. Сабанов

Запоріжжя

2023

З А В Д А Н Н Я

НА КВАЛІФІКАЦІЙНУ БАКАЛАВРСЬКУ РОБОТУ

студенту гр. КІ-119,

спеціальності 123 - «Комп'ютерна інженерія»

Поповій Катерині Олександрівні

1.Тема: «Розробка мікропроцесорної системи регулювання дорожнього руху на перехресті»

затверджена наказом № 02-02 від 11 січня 2023 р.

2. Термін здачі студентом закінченої роботи: 20 червня 2023 р.

3. Перелік питань, що підлягають розробці:

1. Провести огляд предметної області, ознайомитися з літературою та інтернет-джерелами, що присвячені тематиці роботи.
2. Провести огляд та аналіз популярних аналогів, зробити висновки про вимоги до проекту.
3. Провести вибір та огляд апаратної складової проекту.
4. Провести вибір та огляд програмної складової проекту.
5. Здійснити проектування розробляємої системи.
6. Розробити та реалізувати алгоритм функціонування системи
7. Здійснити програмування мікроконтролера платформи.
8. Здійснити зборку та налагодження розробленої системи.
9. Оформити звіт за результатами роботи

4. Календарний графік підготовки бакалаврської дипломної роботи

№ етапу	Зміст	Терміни виконання	Готовність по графіку %, підпис керівника	Підпис керівника про повну готовність етапу, дата
1	Збір практичного матеріалу за темою дипломної роботи	16.01.23-11.02.23		
2	І атестація І розділ бакалаврської дипломної роботи	27.03.23-31.03.23		
3	ІІ атестація ІІ розділ бакалаврської дипломної роботи	24.04.23-28.04.23		
4	ІІІ атестація ІІІ розділ бакалаврської дипломної роботи, висновки та рекомендації, додатки, реферат	22.05.23-26.05.23		
5	Перевірка дипломної роботи на оригінальність	15.05.23-12.06.23		
6	Доопрацювання бакалаврської дипломної роботи, підготовка презентації, отримання відгуку керівника і рецензії	29.05.23-12.06.23		
7	Попередній захист бакалаврської дипломної роботи	12.06.23-18.06.23		
8	Подача бакалаврської дипломної роботи на кафедру	за 3 дні до захисту		
9	Захист бакалаврської дипломної роботи	19.06.23-24.06.23		

Дата видачі завдання: 15 жовтня 2022 р.

Керівник бакалаврської роботи _____ С.О. Сабанов

Завдання отримав до виконання _____ К.О. Попова

РЕФЕРАТ

Бакалаврська дипломна робота містить: 51 сторінка, 8 рисунків, 2 таблиці, 15 першоджерел та 3 додатки.

Об’єктом розробки є інтелектуальні системи управління дорожнім рухом (ІСУДР).

Мета роботи: створення ефективної діючої моделі однієї з найважливіших підсистем ІСУДР – автономного світлофора з програмним управлінням, що регулює рух на перехресті.

У роботі детально розглядаються всі етапи розробки проекту, починаючи від планування та аналізу вимог до збірки апаратної частини пристрою, її програмування, тестування та налагодження.

В роботі використовується широко розповсюджена бюджетна мікроконтролерна платформа Arduino Nano, що повністю забезпечує необхідний функціонал пристрою.

Окрема увага приділена реалізації різних режимів роботи світлофора, зокрема звичайного режиму та режиму економії електроенергії. Розглянуті різні програмні методи управління платою контролера.

Результати, отримані в даній роботі, можуть бути корисним для практичного використання під час створення реальних пристроїв та для подальшого вдосконалення алгоритмів роботи ІСУДР в цілому.

ARDUINO IDE, ARDUINO NANO, ATMEGA328,
ІСУДР, МІКРОКОНТРОЛЕР, СИСТЕМИ КЕРУВАННЯ ДОРОЖНІМ РУХОМ, СКЕТЧ

ЗМІСТ

ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ

Слово / словосполучення	Скорочення
A	
Агрегатна система засобів управління дорожнім рухом	АСЗУДР
E	
European On-Board Diagnostics	EOBD
G	
General Packet Radio Service	GPRS
Global Positioning System	GPS
I	
In-Circuit Serial Programming	ICSP
Інтегрирована середа розробки (англ. Integrated Development Environment)	IDE
Intelligent Transport System	ITS
L	
Рідкокристалічний дисплей; англ. Liquid Crystal Display	LCD
T	
Thin-Film Transistor	TFT
U	
Universal Serial Bus	USB
В	
Вольт	В
Ватт	Вт
І	
Інтелектуальні системи управління дорожнім рухом	ІСУДР
К	

Кілобайт	кБ
Кілограм	кг
М	
Міліампер	мА
Мегагерц	МГц
Мікроконтролер	МК
П	
Правила дорожнього руху	ПДР
Т	
Транспортні засоби	ТЗ
Ш	
Швидкість, кілометри за годину	км/год
Широтно-імпульсна модуляція	ШІМ

ВСТУП

Інтелектуальні системи управління дорожнім рухом (ІСУДР) є важливою частиною сучасних міських інфраструктур та можуть допомогти покращити безпеку й комфорт життя громадян, скоротити тимчасові затримки на дорогах та зменшити кількість аварій. Вони використовуються для підвищення ефективності керування дорожнім рухом на основі збору, обробки та аналізу даних, отриманих за допомогою різних датчиків, камер, радарів та інших пристроїв, встановлених на дорогах та транспортних засобах.

Системи управління дорожнім рухом дозволяють автоматично контролювати та регулювати транспортний потік, керувати світлофорами, контролювати швидкість, відстань та поведінку водіїв, оптимізувати маршрути руху, забезпечувати безпеку пішоходів та учасників дорожнього руху загалом.

Історія ІСУДД розпочалася у 1960-х роках з розробки перших систем управління світлофорами на основі програмного управління. У 1970-х роках почали з'являтися складніші системи управління дорожнім рухом, які використовують датчики та комп'ютерну обробку даних.

У 1980-х роках було створено перші системи управління швидкістю та маршрутами, а також системи контролю перетинів. У 1990-х роках почали з'являтися системи автоматичного паркування та системи керування дорожніми знаками.

З розвитком технологій, таких як бездротовий зв'язок, GPS та комп'ютерний зір, ІСУДД стали більш точними, надійними та ефективними. Сьогодні ІСУД широко використовується в усьому світі, щоб покращити управління дорожнім рухом та підвищити безпеку на дорогах.

До складу сучасних систем зазвичай включають блоки керування світлофорами, модулі динамічного керування швидкістю, підсистеми

керування дорожніми знаками, контролю перетинів та автоматичного паркування.

У сучасному світі автоматизація та роботизація процесів стають все більш актуальними завданнями. Однією з найбільш перспективних галузей є автоматизація дорожнього руху, що дозволяє знизити кількість аварій та збільшити безпеку на дорогах.

Кваліфікаційна дипломна робота має на меті розробити ефективну діючу модель однієї з найважливіших підсистем ІСУДР – світлофор, що регулює рух на перехресті для забезпечення безпеки та підвищення якості регулювання транспортного руху.

У роботі детально розглянуті всі етапи розробки проекту з використанням мікроконтролерної платформи Arduino Nano, починаючи від планування і аналізу вимог до програмування та збірки апаратної частини пристрою. Робота також включає опис різних етапів розробки програмного забезпечення, включаючи кодування, тестування та налагодження.

Окрема увага приділена реалізації різних режимів роботи світлофора, зокрема звичайного режиму та режиму економії електроенергії. Розглянуті різні методи управління платою контролера та їх вплив на продуктивність і функціональність пристрою.

РОЗДІЛ 1
СИСТЕМИ КЕРУВАННЯ ДОРОЖНІМ РУХОМ

1.1. Система інтелектуального керування дорожнім рухом

Затори – одна з основних проблем міста, з якою можна намагатись боротись, зокрема, розширюючи дороги та будуючи нові розв'язки. Проте такий підхід застарів, та змінює обличчя міста в гірший бік. Більш прогресивний спосіб – впровадження розумної системи керування дорожнім рухом.

Система інтелектуального управління дорожнім рухом є складною багаторівневою системою.

Перший рівень представлений виконавчими пристроями та датчиками транспортного потоку: світлофорами, камерами відеоспостереження з аналітичними можливостями.

Другий рівень представлений дорожнім контролером: контролер дозволяє керувати світлофорами та інформаційними табло, збирати інформацію про інтенсивність руху та затори на перехрестях, аналізувати та обробляти отриману інформацію, а також швидко орієнтуватися в заторах на основі отриманих даних перехресть, спілкуватися з центральним диспетчерським центром і додавати до архіву свої дії для отримання додаткової інформації.

Третій рівень призначений для передачі даних від дорожнього диспетчера в диспетчерський центр і, при необхідності, отримання команд для управління обладнанням першого рівня.

Четвертий рівень представлений центральними серверними вузлами, центральними диспетчерськими, робочими місцями, які надають інформацію іншим системам і дозволяють керувати окремими пристроями і системою в цілому.

Впровадження системи управління дорожнім рухом на маршруті громадського транспорту дозволить вирішити такі завдання:

- поліпшення інфраструктури управління дорожнім рухом;
- скорочення шкідливих викидів від автомобілів, що стоять у пробках;
- скорочення часу в дорозі та загальне скорочення споживання палива;
- управління пріоритетом руху громадського транспорту за маршрутом перед рештою учасників дорожнього руху;
- максимальне сприяння дотриманню розкладу руху громадського транспорту;
- попередження виникнення заторів та зменшення часу ліквідації заторів на складних та насичених перехрестях;
- надання інформації міським службам про стан завантаженості магістралей, надання можливостей оперативного втручання у дорожню ситуацію для вирішення екстрених ситуацій.

Інтелектуальна система керування дорожнім рухом автоматично керує світлофорами на перехресті, щоб транспортні засоби мали пріоритет при проїзді через перехрестя. [1]

1.2. Вимоги до управління дорожнім рухом

У сучасному світі кількість автомобілів зростає щодня. Якість життя людини все сильніше залежить від ступеня комфорту, якості та швидкості пересування. Забезпечення якісного процесу життєдіяльності міст залежить від ступеня мобільності людини. Активний розвиток автомобільного транспорту спричиняє і низку негативних факторів. Велика кількість автотранспортних засобів на магістралях веде до утворення заторів, зниження швидкості перевезень, подорожчання та зменшення продуктивності процесу перевезень.

Основні характерні риси магістралей це:

- інтенсивний транспортний потік;
- відсутність найближчого зустрічного транспорту;
- багатосмуговість;
- нерівномірність розподілу транспортного навантаження за часом.

Аналіз особливостей руху на магістралях показує необхідність створення і застосування спеціальних автоматизованих систем управління рухом, що враховують характеристики транспортного процесу, що постійно змінюються, і пропонують оптимальне рішення при кожному конкретному випадку. Завдяки даним системам очевидним є збільшення ефективності роботи автотранспорту, постійний моніторинг характеристик транспортного потоку призведе до оптимізації всього процесу руху.

Оптимізація контролю за транспортним потоком на всіх рівнях підвищить продуктивність транспорту, збільшить мобільність населення, зменшить час на перевезення та їх вартість. Підвищення безпеки руху, зниження кількості дорожньо-транспортних пригод та порушень правил руху.

Проблеми, пов’язані з погіршенням функціонування вулично-дорожньої мережі, вплинули на функціонування всього транспортного комплексу міста. Затримки руху під час руху, виникнення заторів, що характеризується збільшенням часу в дорозі, погіршенням транспортного обслуговування, що призводить до збільшення забруднення міського середовища, а також підвищенням рівня шуму, збільшенням кількості ДТП із зазначенням міських вулиць та дороги існують відмінності в можливостях сучасних рівнів автомобілізації.

Найбільші труднощі в раціональній організації дорожнього руху виникають на перехрестях вулиць, оскільки вони є «вузькими місцями» вулично-дорожньої мережі з точки зору ефективного та безпечного забезпечення транспортного та пішохідних потоків.

Для управління рухом на перехрестях найчастіше використовується світлофорне регулювання, яке може підвищити безпеку руху, зменшити

затримки для учасників дорожнього руху, зменшити енергоспоживання та негативний вплив на навколишнє середовище, що, в свою чергу, значно впливає на якість міського життя.

1.3. Структура автоматизованої системи управління дорожнім рухом

Одним із основних напрямків розвитку техніки управління дорожнім рухом є створення агрегатної системи засобів управління дорожнім рухом (АСЗУДР).

Рисунок 1.1 – Перехрестя зі світлофорами

Створення АСЗУДР дозволяє знизити питомі витрати на проектування систем, скоротити терміни їх впровадження, зменшити затримки транспортних засобів за рахунок скорочення часу розвитку систем відповідно до зміни характеристик транспортних потоків.

У додатках до АСЗУДР розглядаються структура та методики визначення доцільності впровадження, обстеження об'єкта управління, розміщення периферійного обладнання, типові вимоги до суміжних частин проекту, інструкції з підготовки до впровадження системи та ін. [2]

1.4. Огляд систем управління дорожнім рухом

1.4.1. SEA TCS (TrafficControlSystem)

Компанія SEA пропонує сучасну систему контролю дорожнього руху власного виробництва – SEA TCS (TrafficControlSystem) – яка дозволяє вирішувати наступні завдання:

- зменшити затори на дорогах і ймовірність утворення заторів;
- запобігання надмірного зносу дорожнього покриття;
- забезпечення зв'язку зі світлофорними об'єктами та оперативний контроль їх функціонування;
- узгодити схеми світлофорів для зменшення інтенсивності руху;
- зменшити рівень шуму та концентрацію вихлопних газів у місцях скупчення транспортних засобів;
- значно підвищити рівень безпеки дорожнього руху та ефективність організації дорожнього руху.

Функціонування системи забезпечується високотехнологічним світлофорним обладнанням та спеціалізованим програмним забезпеченням АСКДР розробки компанії СЕА.

Основні функції:

- Локальне та узгоджене управління рухом об'єктів вулично-дорожньої мережі.
- Відправляє один світлофорний об'єкт і його групи.
- Постійний моніторинг та діагностика стану периферійних пристроїв (РТК дорожніх контролерів, світлофорів та ін.).
- Відображення стану сигналу світлофора та циклічних діаграм у реальному часі.

- Динамічний режим «Зелена хвиля» – одночасно контролює освітлення світлофорів, інформаційних панелей, електронних дорожніх знаків і пішохідних переходів.

Персонал диспетчерського центру має доступ до всіх функцій системи автоматичного керування рухом SEA TCS через веб-інтерфейс диспетчера. [3]

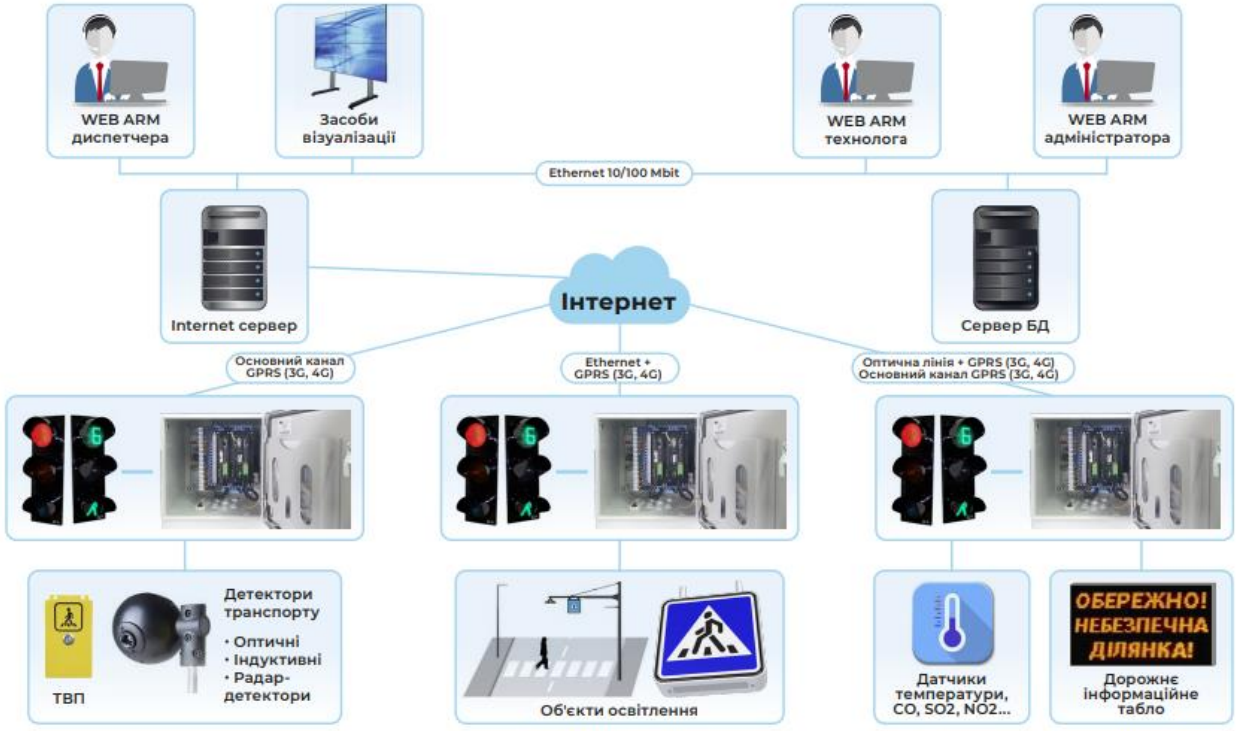


Рисунок 1.2 – Компоненти TrafficControlSystem

І хоча на трасах поки що продовжується так звана «аварійна війна», можливо, у майбутньому завдяки впровадженню подібних систем вона нарешті закінчиться. Тоді безпечне, швидке, комфортне та економічне перевезення пасажирів і вантажів стане звичайним явищем.

1.4.2. Дорожній менеджер

Система «Дорожній менеджер» призначена для моніторингу, моделювання, прогнозування міських транспортних потоків та інтелектуального керування дорожнім рухом у реальному режимі часу. [2]

Дорожній менеджер включає такі основні підсистеми:

- Лінійка автономних апаратно-програмних модулів для відеомоніторингу транспортних потоків із функціями підрахунку, класифікації та ідентифікації автотранспорту на вулицях міста, включаючи номери транспортних засобів (МС). Забезпечує формування додаткових джерел інформації про транспортні потоки в точках, необхідні побудови повної картини руху ТЗ.
- Цифрова платформа, що забезпечує збирання та попередню обробку інформації від різних джерел (модулі відеомоніторингу, оглядові камери, дані GPS-навігації пасажирського транспорту та ін.), прогнозування розвитку пробок та візуалізацію результатів.
- Підсистема інтелектуального керування дорожнім рухом з можливістю її постійного навчання та розвитку.
- Сервіс для населення, що рекомендує найшвидші та безпечніші маршрути.

Кожна з підсистем є автономною і може поставлятись окремо за бажанням замовника. [4]

1.4.3. Росток-ЕЛЕКОМ

Підприємство «РОСТОК-ЕЛЕКОМ» – виробник і системний інтегратор програмно-апаратних рішень в області управління дорожнім рухом на принципах ITS (Intelligent Transport System).

Росток-ЕЛЕКОМ має науково-технічний потенціал та виробничі потужності для вирішення наступних завдань:

- Аналіз дорожньої обстановки, проектування систем управління дорожнім рухом.
- Виробництво і постачання повного спектру апаратно-програмних засобів керування дорожнім рухом.
- Компанія має двадцятирічний досвід створення автоматизованих систем керування дорожнім рухом в містах України (Київ, Житомир, Одеса, Дніпропетровськ, Чернівці, Миколаїв, Кременчук) і за кордоном (Молдова, Кишинів, 2001 г.; Латвія, Рига, 2002 г.; Грузія, Тбілісі, 2012 р.) та елементів до них: центри керування, контролери, світлофори, комплекти зв'язку з різними способами передачі даних, спеціалізоване програмне забезпечення. Наше устаткування працює більш ніж в 60 містах України. [4]

Росток-ЕЛЕКОМ випускає різноманітні модулі для створення систем управління дорожнім рухом деякі з них представлено далі.

Комплект відео детекторів транспортних засобів RE2023 призначений для виявлення транспортних засобів на контрольованих ділянках дорожньої мережі та визначення характеристик їх руху.

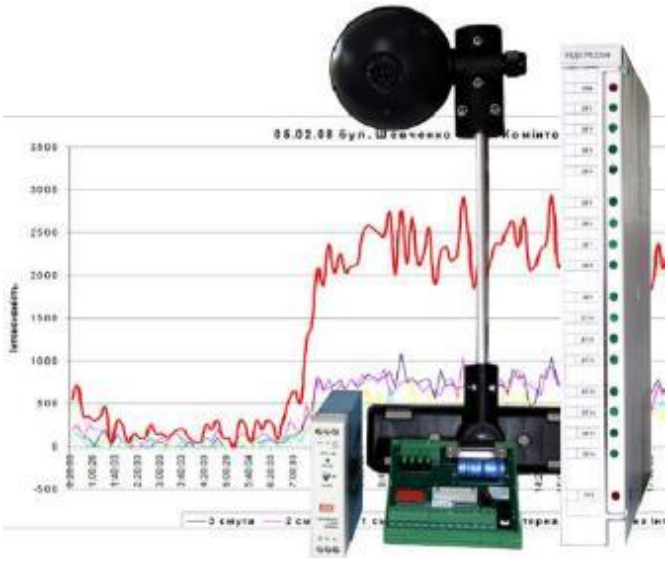


Рисунок 1.3 – Комплект РЕ2023

Детектори (датчики) транспортних засобів – це технічні пристрої, призначені для виявлення транспортних засобів, реєстрації кількості транспортних засобів, що проїжджають через транспортні розв'язки, визначення параметрів транспортного потоку тощо.

Таблиця 1.1 – Характеристики відеодетекторів PE2023

Основні технічні характеристики	
Похибка виявлення транспортних засобів, не більше, %	4
Розрахунок інтенсивності транспортного потоку, фізичних одиниць на годину	от 0 до 1500
Розрахунок швидкості руху транспортних засобів з похибкою не більше 4 %, км/год	от 3 до 120
Максимальна кількість детекторів транспорту, не менше	4
Максимальна кількість полос руху, які контролюються, не менше	16
Потужність, що споживається виробом від однофазної мережі перемінного току, не більше, Вт	18
Маса складових частин ВДТ, не більше, кг	5

Виявлення інтенсивності руху за допомогою детекторів транспортних засобів дозволяє визначати швидкість, типі кількість транспортних засобів та адаптивно керувати світлофорами.

Детектори транспортних засобів класифікуються за призначенням, принципом роботи чутливого елемента та спеціальними характеристиками (параметрами, що вимірюються).

1.4.4. TransportTelematics

Сучасне визначення телематики тісно пов'язані з практикою відстеження автомобілів та інших активів, коли знання їх місцезнаходження як реального часу має вирішальне значення для безперебійної роботи бізнесу. Такі технології, як GPS-стеження, є прикладом телематики, яка використовується у сучасному керуванні автопарком.

Для передачі, прийому та зберігання даних телеметрії на автомобілі, що входять до складу автопарку, встановлюються пристрої, що входять до складу телематичної системи керування автопарком. SIM-картка встановлює зв'язок із бортовою системою діагностики або портом CAN-BUS автомобіля. Інформація передається через бездротову мережу через вбудований модем.

Потім ця інформація надсилається на централізований сервер через службу пакетного радіозв'язку загального призначення (GPRS), мобільні дані 4G (скоро буде 5G) та стільникові мережі або супутникове з'єднання. Пристрій також збирає різні дані про транспортні засоби на додаток до GPS-відстеження автопарку. Однак інформація надходить на сервер, який потім відображає її на захищеному веб-сайті, який можна переглядати за допомогою мобільних пристроїв, таких як планшети та смартфони. Передача інформації між постачальником телематичних послуг і автомобілем управляє компанія, що спеціалізується на телекомунікаціях.

Пристрій здатний записувати різні дані, у тому числі розташування автомобіля, його швидкість, будь-яке різке прискорення або гальмування, кількість часу, яке він провів на холостому ходу, витрата палива та оцінку загального стану двигуна. Інформація після оцінки та обробки може надати вам глибше розуміння всього вашого автопарку.

Взагалі, існує шість різних категорій телематичних систем. По-перше, це системи «чорна скринька», що є стандартом де-факто для проектів UBI у Великобританії. Технологія «чорної скриньки» збирає та генерує потік даних від підключених транспортних засобів. Чорні ящики або інші стаціонарні електричні пристрої, безпечно поміщені в транспортний засіб, записують та

надсилають дані про поїздку та будь-які нещасні випадки, які могли статися. Цей метод часто використовується в місцях, де часто викрадають автомобілі, оскільки це перевірений та надійний метод, який не можна легко обійти і який призводить до швидкого повернення вкрадених автомобілів. Однак ці аксесуари для автозапчастин мають бути професійно встановлені у транспортних засобах, що збільшує вартість будівництва.

Телематичні системи Obd II – це одна з ранніх розробок в області автомобільної телематики, яка з 1996 року є обов'язковою для всіх автомобілів, що продаються в США. Аналогічні стандарти можна знайти в Європі, де вони називають європейською бортовою діагностикою (EOBD). Будучи давнім і добре відомим обхідним рішенням, пристрої OBD, що повністю підключаються, завоювали довіру і надійність на ринку. У більшості випадків інформація про водіїв миттєво передається мобільними мережами. Доступний варіант, який можна використовувати у поєднанні з можливістю підключення смартфона для підвищення залучення пасажирів та водіїв.

Ще один різновид – 12-вольтові телематичні системи, що підключаються, для самостійної установки. Джерелом живлення цих інструментів є розетка 12 В у автомобілі.

Також можна відмітити систему управління міською мобільністю MyCity: швидка, гнучка технологія для розумніших, екологічніших міст. Програмне забезпечення для управління міською мобільністю SWARCO MyCity для малих, середніх і великих міст. Його було розроблено для вирішення проблем, пов’язаних із двома ключовими проблемами, з якими стикаються міста: швидкі зміни типів міської мобільності та IT-ландшафту, необхідного для її підтримки; швидка урбанізація та її вплив на міське середовище.

Рисунок 1.4 – Інтерфейс SWARCO MyCity

Система управління дорожнім рухом MyCity (TMS) знімає навантаження, пов’язану з керуванням транспортними потоками в малих і великих містах. Його зручний та інтуїтивно зрозумілий інтерфейс дозволяє операторам швидше та легше завчасно керувати трафіком на міських дорогах. Це допомагає забезпечити більш ефективні транспортні потоки, таким чином знижуючи ризик заторів, зменшуючи забруднення повітря, скорочуючи час у дорозі та спонукаючи більше людей обирати альтернативні форми мобільності. [5]

1.4.5. Traficon

Traficon – це мобільна система моніторингу дорожнього руху для виявлення порушень швидкості, збору статистики та можливого переслідування порушників ПДР.

Система є портативною та складається з двох камер GoPro, підключеного комп’ютера з GPS-приймачем та спеціально розробленого програмного забезпечення. Вся система монтується та готується менш ніж за десять хвилин і не має постійно встановлених пристроїв чи обладнання.

Рисунок 1.5 – Інтерфейс програми Traficon

Наразі Traficon призначений для використання на дорогах із кількома напрямками та вимірює швидкість лише транспортних засобів, які проїжджають у тому ж напрямку.

Система використовує позиціонування GPS і бібліотеку комп’ютерного зору з відкритим кодом, а також унікальний алгоритм для аналізу відеопотоку та зображення.

Traficon складається з трьох основних програмних компонентів:

- Лінія провулку. Система стеження. Цей модуль підраховує лінії смуг і визначає довжину та відстань між ними.
- Транспортний засіб. Система ідентифікації. Цей модуль ідентифікує транспортні засоби та номерні знаки та виявляє порушення швидкості на окремих транспортних засобах шляхом порівняння пройденої відстані з часом. Інформація про транспортні засоби та власників

автоматично отримується з Національного реєстру транспортних засобів під час процесу відстеження.

- Послідовність. Модуль сканування. Ця процедура автоматично виявляє порушення швидкості та постійно зберігає інформацію в базі даних. Потім ви зможете обробити дані пізніше. Програмне забезпечення може обробляти чотири одночасних вимірювання. Нарешті, вибраний вихід може бути зашифрований і переданий на сервер для обробки в додатковому програмному забезпеченні.

Програмне забезпечення було розроблено для використання в основному в проекті економічних досліджень у співпраці між університетом і приватним бізнесом.

Traficon – це некомерційний проект, який має на меті сприяти дослідженню альтернативних соціальних доходів і можливого розвитку соціального забезпечення. [6]

1.5. Висновки за першим розділом

Напружений трафік сучасних транспортних магістралей потребує впровадження систем управління дорожнім рухом на маршруті, що дозволяє вирішити такі проблеми, як виникнення заторів, порушення розкладу руху громадського транспорту, скорочення шкідливих викидів від автомобілів, що стоять у пробках, скорочення часу в дорозі, загальне скорочення споживання палива, тощо.

Наразі широке поширення набувають інтелектуальні системи керування дорожнім рухом (агрегатні системи засобів управління дорожнім рухом – АСЗУДР). Розробка та впровадження таких систем дозволяє знизити питомі витрати на проектування та скоротити терміни їх впровадження за рахунок скорочення часу розвитку систем відповідно до зміни характеристик транспортних потоків. В якості прикладу АСЗУДР можна навести системи

контролю дорожнього руху SEA TCS, «Дорожній менеджер», Росток-ЕЛЕКОМ, TransportTelematics, SWARCO MyCity Traficon та інші.

РОЗДІЛ 2
ПРОГРАМНО-АПАРАТНІ ЗАСОБИ КОМПЛЕКСУ

2.1. Загальні відомості про платформу Arduino

Arduino – це електронна платформа з відкритим вихідним кодом, яка дозволяє взаємодіяти з навколишнім світом. Фактично це назва комплексу апаратно-програмних засобів для створення простих електронних систем автоматики та робототехніки. Вона може служити для розробки автономних інтерактивних пристроїв та працює під керуванням ПЗ, встановленого на з'єднаному з ним комп'ютерів. Система має повністю відкриту архітектуру та орієнтована на непрофесійних користувачів.

Сімейство Arduino – кілька моделей так званих налагоджувальних плат. Налагоджувальна плата являє собою друковану плату, на якій стоїть мікроконтролер. У молодших платах Arduino використовуються мікроконтролери AVR (UNO, Nano, Mega, Leonardo), у сучасних моделях стоять потужніші ARM Cortex для більш серйозних проектів. Зокрема, можна назвати такі сімейства:

- Arduino Uno;
- Arduino Leonardo;
- Arduino Nano;
- Arduino Mini;
- Arduino Micro;
- Arduino Mega;
- Arduino Due. [11]

Плати мають у своєму складі все необхідне для комфортної роботи, але їхньої функціональності часто буває недостатньо. Щоб зробити проект більш інтерактивним, можна використовувати різні модулі та плати розширень,

сумісні з платформою Arduino. Сюди входять датчики (температури, освітлення, вологи, газу/диму, атмосферного тиску), пристрої введення (клавіатури, джойстики, сенсорні панелі) та виведення (сегментні індикатори, LCD/TFT дисплеї, світлодіодні матриці).

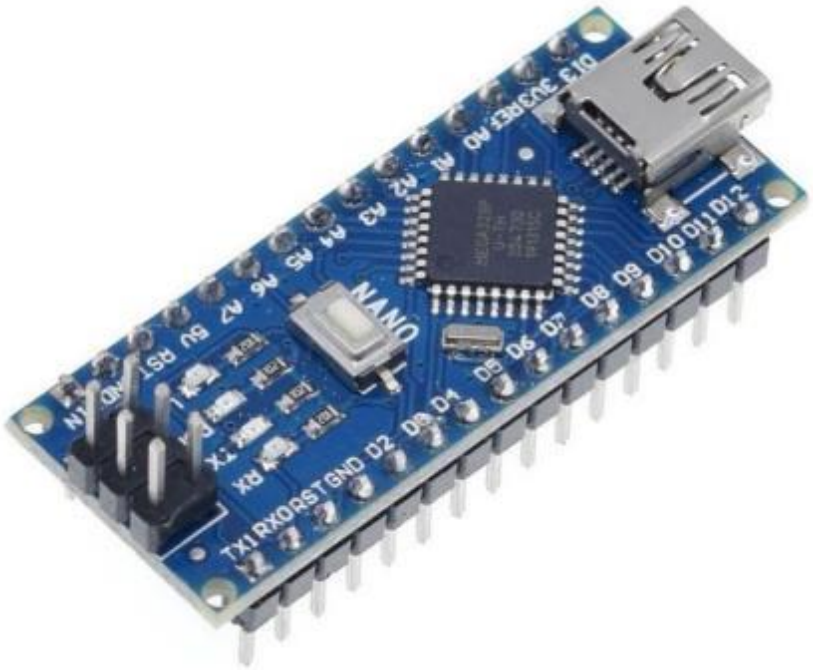


Рисунок 2.1 – Вигляд плати Arduino Nano

На програмному рівні платформа Arduino підтримується безкоштовним середовищем розробки Arduino IDE. Мікроконтролери треба програмувати мовою C++, з деякими відмінностями та полегшеннями, створеними для швидкої адаптації початківців. Компіляцію програмного коду та прошивку мікроконтролера середовище розробки бере на себе. [7]

2.2. Особливості реалізації Arduino Nano

2.2.1. Базовий мікроконтролер

Плата Arduino Nano поставляється з мікроконтролером ATmega328P, який має 32 кілобайти флеш-пам'яті, 2 кілобайти оперативної пам'яті та 14 цифрових входів/виходів. Це мікроконтролер виробництва компанії Atmel

(зараз Microchip), який використовується у багатьох проектах для керування різними електронними пристроями.

Мікроконтролер ATmega328P – це 8-бітний мікропроцесор, який заснований на архітектурі RISC (покращеною AVR). Дані мікроконтролери є одними із найпопулярніших серій лінійки AVR. Сімейство мікроконтролерів AVR, куди входить atmega328p, сьогодні широко застосовується при конструюванні електроніки різного рівня.

В залежності від корпусування, Atmega328p має 28 або 32 контакти.

Технічні характеристики ATmega328:

- Архітектура: AVR RISC.
- Робоча напруга: 1.8V - 5.5V.
- Частота роботи: Залежно від конфігурації, зазвичай 16 МГц.
- Кількість вхідно-вихідних (I/O) пінів: 23.
- Кількість аналогових вхідних пінів: 6.
- Кількість PWM (ШІМ) каналів: 6.
- Кількість таймерів: 3 (16-бітний і 8-бітні).
- Кількість USART (UART) інтерфейсів: 1.
- Кількість SPI інтерфейсів: 1.
- Кількість I2C (TWI) інтерфейсів: 1.
- Кількість вбудованих аналогово-цифрових перетворювачів (ADC): 1 – 10-бітний ADC з 6 вхідними каналами.
- Кількість вбудованих компараторів: 1.
- Кількість вбудованих програмованих таймерів/лічильників (PWM): 3.
- Кількість вбудованих програмованих таймерів/лічильників (звичайних): 3.
- Кількість флеш-пам'яті: 32 кБ (з яких 0.5 кБ використовується для завантажувача).
- Кількість оперативної пам'яті (RAM): 2 кБ.

- Кількість EEPROM: 1 кБ.
- Корпусування: 28-виводовий корпус (DIP-28), 32-виводовий корпус (TQFP-32), 32-виводовий корпус (QFN-32), 32-виводовий корпус (MLF-32).

ATmega328 є мікроконтролером, який використовується в багатьох електронних пристроях та проектах різного рівня складності, зокрема у мікроконтролерних пристроях, сенсорах, роботах, інтерактивних іграх тощо. Для розробки програмного забезпечення для ATmega328 використовуються різноманітні платформи та інструменти.

2.2.2. Технічні характеристики платформи

Arduino Nano – це налагоджувальна плата невеликого розміру (рисунок 2.2), яка входить до трійки лідерів за популярністю серед радіоаматорів-програмістів. Незважаючи на свій скромний розмір, вона практично нічим не поступається найбільш поширеній Arduino Uno за функціоналом і може використовуватися в проектах, де габарити відіграють істотну роль.

Ранні версії Arduino Nano базувалися на основі мікроконтролера ATmega168. Починаючи з версії 3.0, у них встановлені більш просунуті ATmega328, зі збільшеним обсягом FLASH та EEPROM-пам'яті, а також з більшою тактовою частотою.

Для спілкування із зовнішнім світом у цій платі передбачені штирьові колодки. Це зручно для макетування, але за бажанням їх можна не встановлювати. У такому разі дроти до потрібних виводів припаюються безпосередньо. Також штирьові колодки потрібні при використанні в проекті спеціалізованих плат розширення (шилдів), яких для даної модифікації Arduino придумано безліч.

Якщо уважно розглянути плату Arduino Nano, то на ній можна помітити не тільки мікроконтролер ATmega328, а й низку додаткових компонентів, які забезпечують життєдіяльність цього апаратного комплексу в цілому.

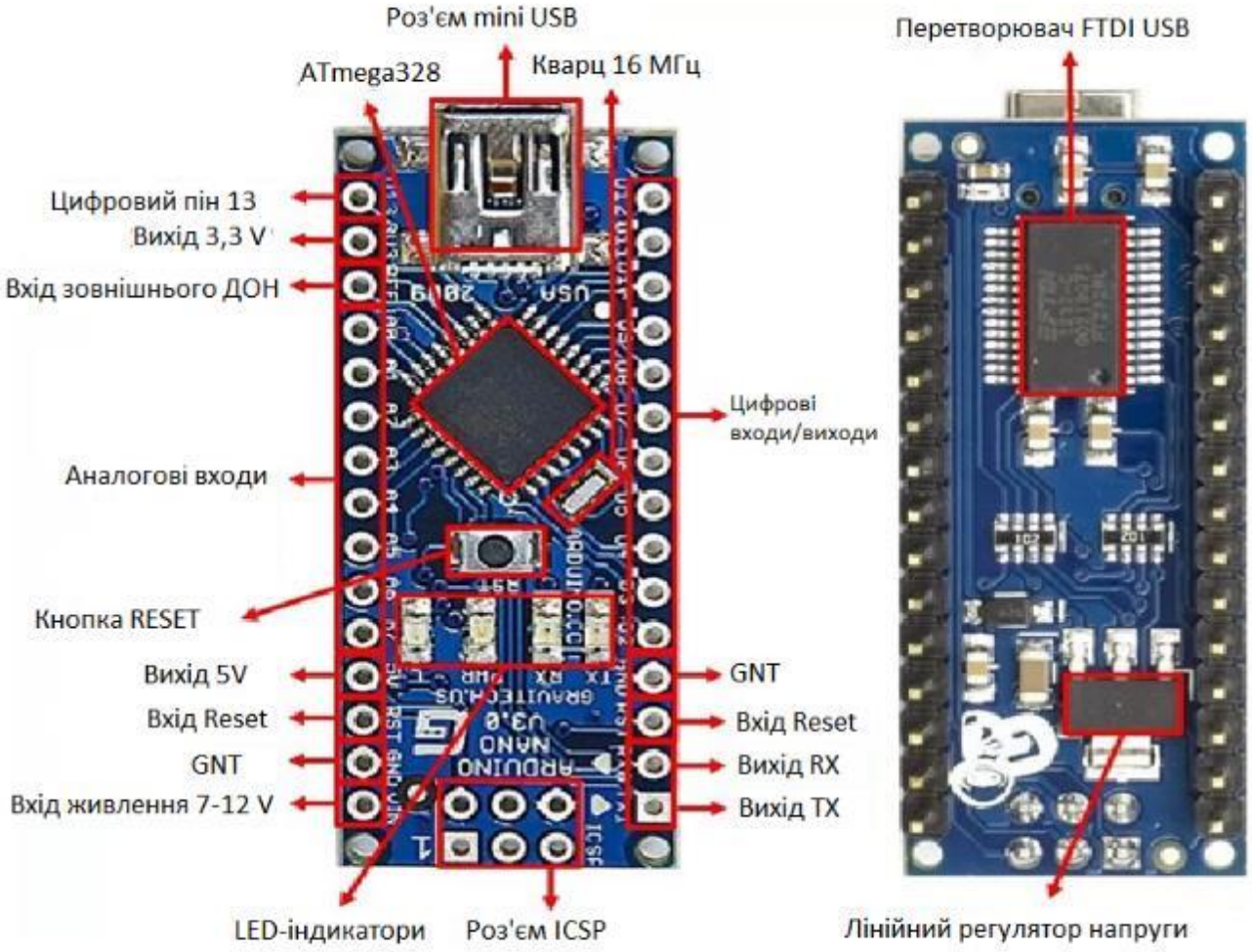


Рисунок 2.2 – Склад плати Arduino Nano

Щоб заощадити місце, розробники розташували радіоелементи з обох боків плати. З лицьового боку нанесена вся інформативна шовкографія, встановлений мікроконтролер ATmega328, кварцовий резонатор, роз'єм mini-USB, кнопка скидання та чотири індикаторні світлодіоди (TX, RX, PWR і L). Перші два світлодіоди загоряються при обміні даними плати Arduino Nano з іншими пристроями через серійний послідовний порт. [8]

Таблиця 2.1 – Характеристики платформи

Мікроконтролер	ATmega328
Робоча напруга	5 В
Вхідна напруга (рекомендована)	7-12 В
Вхідна напруга (гранична)	6-20 В
Цифрові входи/виходи	14 (6 з яких можуть використовуватися як виходи ШІМ)
Аналогові входи	6
Постійний струм через вхід/вихід	40 мА
Постійний струм для виведення 3.3	50 мА
Флеш пам`ять	32 Кб
ОЗУ	2 Кб
EEPROM	1 Кб
Тактова частота	16 МГц

Мікроконтролер загального призначення ATmega328p, побудований за RISC-архітектурою – це основа Arduino. Код програми зберігається в його внутрішній пам'яті, і він же виконує всі інструкції, що містяться в його енергонезалежній пам’яті, підвищуючи і знижуючи напругу і зчитуючи зовнішні входи. Коли говориться про підключення чогось до Arduino, насправді говориться про підключення чогось до мікроконтролера ATmega328p – середовище Arduino та друкована плата просто забезпечують зручну упаковку, яка робить підключення до ATMega328 трохи інтуїтивнішим. [10]

2.2.3. Ресурси платформи, що використані в проекті

В проекті задіяна лише незначна частка ресурсів платформи. Виходячи з того, що модель повинна забезпечити керування перехрестям, випромінювачі, що розташовані на протилежних сторонах світлофора повинні працювати синхронно. Тобто, для обслуговування пари однакового кольору можна обмежитися однією лінією. Загалом система потребує шість цифрових ліній для управління трафіком в двох напрямках. На платі для цього було обрано виводи D2-D7, сконфігурованих на виведення. Враховуючи, що максимальний струм, що може бути спожито від цифрового виходу Arduino, становить 40 мА, в моделі було підібрано пари послідовно ввімкнених світлодіодів, струм споживання яких було обмежено до 30 мА за допомогою резисторів.

Для підключення перемикача, що дозволяє встановлювати режим «блимаючий жовтий», біла використана ще одна лінія (D8). Пін сконфігуровано на введення.

Затримки формуються вбудованими таймерами мікроконтролера.

2.3. Середовище розробки Arduino IDE

Середовище розробки Arduino складається з вбудованого текстового редактора програмного коду, області повідомлень, вікна виведення тексту (консолі), панелі інструментів з кнопками команд, що часто використовуються, і кількох меню. Для завантаження програм та зв'язку середовище розробки підключається до апаратної частини Arduino.

Інтерфейс середовища розробки Ардуїно містить такі основні елементи: текстовий редактор для написання коду, область для виведення повідомлень,

текстова консоль, панель інструментів із традиційними кнопками та головне меню. Даний софт дозволяє комп'ютеру взаємодіяти з Ардуїно як передачі даних, так прошивки коду в контролер.

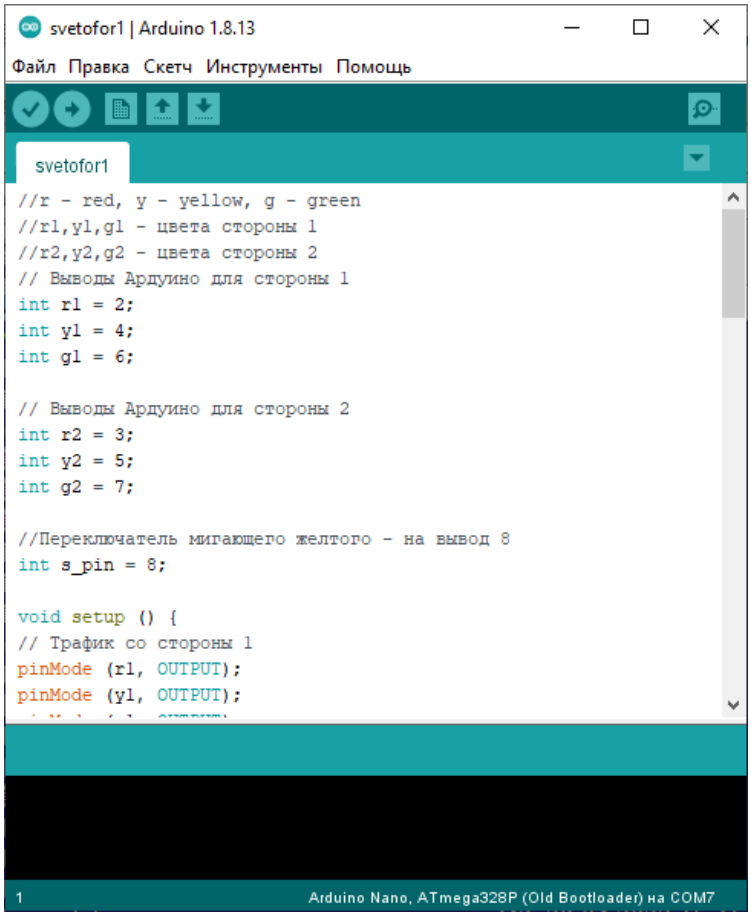


Рисунок 2.2 – Інтерфейс Arduino

Програма, написана в Arduino IDE, називається скетч. Скетч пишеться в текстовому редакторі, який має інструменти вирізки/вставки, пошуку/заміни тексту. Під час збереження та експорту проекту в області повідомлень з'являються пояснення, також можуть відображатися помилки. Вікно виведення тексту (консоль) показує повідомлення Arduino, які містять повні звіти про помилки та іншу інформацію. Кнопки панелі інструментів дозволяють перевірити та записати програму, створити, відкрити та зберегти скетч, відкрити моніторинг послідовної шини. [12]

2.4. Реалізація багатозадачності в Arduino

2.4.1. Багатозадачність з yield()

Функції yield() дозволяє виконувати свій код усередині затримок delay(). Ця функція дозволяє швидко реалізувати "паралельне" виконання двох завдань: одного по затримці, а другого – постійно. Ось приклад, у якому блимає світлодіод та використовується кнопка:

```
void setup () {
  pinMode ( 13, OUTPUT ) ;
}
void loop () {
  digitalWrite ( 13, 1 ) ;
  delay ( 1000 ) ;
  digitalWrite ( 13, 0 ) ;
  delay ( 1000 ) ;
}
void yield () {
  // а тут можна опитувати кнопку
  // і не пропустити натискання з-за delay
}
```

Не менш життєвим буде приклад зі сценарієм руху крокового двигуна або плавного руху сервоприводу, які потребують частого виклику "функцій руху".

Розглянемо абстрактний приклад руху мотора по кількох заданих точках, функція обертання мотора повинна викликатися якнайчастіше (так зроблено майже у всіх бібліотеках для крокових моторів):

```
void setup () {}
void loop () {
  // Задати цільовий кут №1
  delay ( 1000 ) ;
  // Задати цільовий кут №2
  delay ( 120 ) ;
  // Задати цільовий кут №3
  delay ( 2000 ) ;
  // Задати цільовий кут №4
  delay ( 250 ) ;
  // Задати цільовий кут №5
  delay ( 600 ) ;
}
void yield () {
  // обертати двигун
}
```

Таким чином можна швидко розписали траєкторію руху для крокового мотора за часом, не використовуючи якісь таймери і бібліотеки.

2.4.2. Багатозадачність з millis()

Більшість прикладів до різних модулів/датчиків використовують затримку delay (), як "гальмування" програми, наприклад для виведення даних з датчика в послідовний порт.

За допомогою функцій часу millis() абоmicros() можна організувати програмний таймер, яким і виконувати необхідні дії. Схема така:

- Заводимо змінну для таймера типу unsigned long (uint32_t) – саме цей тип повертає millis ().
- Шукаємо різницю між поточним часом роботи програми та змінною таймера.
- Якщо різниця більша за необхідний період – виконуємо потрібний код і скидаємо таймер.

Всі розглянуті нижче конструкції можуть також працювати з micros() для створення мікросекундних таймерів.

Реалізація класичного "таймера на millis()" виглядає так:

```
#define MY_PERIOD 500 // період мс
uint32_t tmr1;          // Змінна таймера
void setup () {}
void loop () {
  if ( millis ( ) - tmr1 > = MY_PERIOD ) { // шукаємо
                                              різницю
  tmr1 = millis () ;                        // скидання
                                               таймера

    // Виконати дію
  }
}
```

Дана конструкція виходить з періоду, якщо в коді є затримки та інші блокуючі ділянки, під час виконання яких millis() встигає збільшитися на час, більший за період таймера. Це може бути критично, наприклад, для

підрахунку часу та інших схожих ситуацій, коли період спрацьовування таймера не повинен зміщуватися.

У той же час, якщо заблокувати виконання коду на час, більший за один період – алгоритм просто скоригує цю різницю, тому що ми скидаємо його актуальним значення millis (). [13]

2.4.3. Багатозадачність із перериваннями таймера (для AVR)

Для особливо критичних до часу завдань можна використовувати виконання переривання таймера. Це можуть бути такі завдання:

- динамічна індикація;
- генерація певного сигналу/протоколу зв'язку;
- програмний ШІМ;
- "тактування" крокових моторів;
- будь-який інший приклад виконання через зазначений час або просто періодичне виконання за суворим періодом.

Для налаштування переривань за таймерами 1 і 2 є бібліотеки TimerOne та TimerTwo. Зроблено бібліотеку, GyverTimers, в якій є таймер 0 (для програмування без використання Arduino.h), а також усі таймери на Arduino MEGA (6 штук). Ознайомитися з документацією та прикладами можна на сторінці бібліотеки.

Розглянемо простий приклад, в якому "паралельно" Blink будуть відправлятися дані в порт. Приклад відірваний від реальності, так робити не можна, але він важливий для розуміння самої суті: код у перериванні виконається у будь-якому випадку, йому байдужі затримки та нескінченні цикли в основному коді.

```
#include "GyverTimers.h"
void setup () {
  Serial. begin ( 9600 ) ;
```

```
  // Встановлюємо період таймера 333000 мкс -> 0.333 с
      (3 рази на секунду)
  Timer2. setPeriod ( 300000 ) ;
  Timer2. enableISR () ;   // запускаємо переривання на
                                  каналі A таймера 2
  pinMode ( 13, OUTPUT ) ;  // блиматимемо
}
void loop () {
  // "Млинець"
  digitalWrite ( 13, 1 ) ;
  delay ( 1000 ) ;
  digitalWrite ( 13, 0 ) ;
  delay ( 1000 ) ;
}
// Переривання A таймера 2
ISR ( TIMER2_A ) {
  Serial. println ( "isr!" ) ;
}
```

Перериваня по таймеру є дуже потужним інструментом, але таймерів у МК не так багато і потрібно використовувати їх лише у разі реальної необхідності. 99% завдань можна вирішити без переривань таймера, написавши оптимальний основний цикл та грамотно застосовуючи millis (). [14]

2.5. Створення об'єктів і класів в Arduino

Клас є одним з найбільших і важливих інструментів мови C++, саме він робить мову об'єктно-орієнтованою і дуже потужною. Саме завдяки доданню класів у мову Сі і з'явилася мова C++, її навіть називають "Сі з класами". Деякі вбудовані інструменти Arduino також є об'єктами, наприклад Serial.

Клас – це просто незалежна підпрограма, в якій є свій набір змінних та функцій. В основній програмі можна створити екземпляр класу (він називається об'єкт) і користуватися тими інструментами, які є в класі. Використання класів дозволяє:

- Розділити складну програму на окремі незалежні частини.
- Створювати зручні бібліотеки.

– Використовувати свої “напрацювання” в іншому проекті, не переписуючи один і той самий код.
– Полегшити і спростити програму, якщо в ній використовуються повторювані конструкції та алгоритми.

Розглянемо приклад із бібліотеки Servo, взявши з неї приклад Knob:

```
#include <Servo.h> // підключаємо заголовний файл
                      бібліотеки, Servo.h
Servo myservo;  // створюємо ОБ'ЄКТ myservo КЛАСУ Servo
int potpin = 0;
int val;
void setup () {
  myservo.attach ( 9 ) ;  // застосовуємо МЕТОД attach
                             до ОБ'ЄКТУ myservo
}
void loop () {
  val = analogRead ( potpin ) ;
  val = map ( val, 0, 1023, 0, 180 ) ;
  myservo.write ( val ) ;        // застосовуємо МЕТОД
                                    write до ОБ'ЄКТУ
                                    myservo
  delay ( 15 ) ;
}
```

Клас є самодостатньою одиницею програми – він містить змінні та функції, які з ними взаємодіють. Всі можливості класу та взаємодія з його даними доступні у його функціях і не потребують додаткового коду.

Класи дуже схожі на структури, як за оголошенням, так і за використанням, але клас є набагато потужнішою одиницею мови завдяки механізмам успадкування.

Клас оголошується за допомогою ключового слова class і містить у собі члени класу – змінні та функції:

```
class Ім'я_класу {
  член1;
  член2;
};
```

Важлива відмінність від структури: вміст класу ділиться на області: громадські та приватні. Вони визначаються за допомогою ключових слів public і private, область діє до початку наступної області або до фігурної дужки класу, що закриває:

- Public – члени класу в цій галузі доступні для взаємодії з основної програми (скетчу), в якій буде створено об'єкт. Наприклад write() у серво.
- Private – члени класу в цій галузі доступні тільки всередині класу, тобто з програми до них не можна звернутися.

```
class Ім'я_класу {
  public :
  // Список членів, доступних у програмі
  private :
  // Список членів для використання всередині класу
} ;
```

Можна побачити в “публічному доступі” всі методи, якими можна скористатися під час роботи з Servo. Це дуже зручно, тому що не потрібно шукати документацію – написано все в описі класу. Методи оголошуються так само, як прості функції.

Приватні члени в класі серво – змінні, з їх назв можна зрозуміти, що вони зберігають. Доступу до цих змінних "зі скетчу" немає, звертатися до цих змінних можуть лише методи класу.

Може використовуватися для ініціалізації змінних, передачі налаштувань тощо.

Створення об'єкту відбувається так само, як і структури – на ім'я класу [15]:

```
назва_класу ім'я_об'єкта;       // Створити об'єкт
назва_класу ім'я_об'єкта [ 10 ] ;   // Створити масив
                                       об'єктів
```

Звернення до членів класу здійснюється так само, як у структурах: через оператор точка.:

```
ім'я_об'єкта.метод () ;          // виклик методу
ім'я_об'єкта [номер] .метод () ;  // Виклик методу для
                                     об'єкта з масиву
                                     об'єктів.
```

2.6. Висновки за другим розділом

Платформа Arduino Nano побудована на мікроконтролері ATmega328. Вона має 14 цифрових вхід/виходів (6 з яких можуть використовуватися як

виходи ШІМ), 6 аналогових входів, кварцовий генератор 16 МГц, USB-роз'єм, силовий роз'єм, роз'єм ICSP і кнопку перезавантаження. Такого апаратного забезпечення цілком достатньо для реалізації задач моделювання окремого блоку інтелектуальної системи управління дорожнім рухом, де з усього арсеналу платформи буде задіяно лише сім цифрових ліній, що будуть сконфігуровані, згідно потребам.

На етапі моделювання в якості індикації будуть використовуватися пари світлодіодів, що споживатимуть не більше 30 міліампер. Таким струмом контролер може керувати без застосування додаткових схемотехнічних рішень.

Програмування системи відбуватиметься в середовищі розробки Arduino IDE. Реалізація багатозадачності може бути виконана трьома способами:

1) за допомогою функції yield ();
2) за допомогою функцій часу millis () або micros ();
3) перериваннями таймера.

В проекті задіяна незначна частка ресурсів Arduino Nano: виводи D2-D7, що їх сконфігуровано на виведення та пін D8, що його сконфігуровано на введення.

РОЗДІЛ 3.
РЕАЛІЗАЦІЯ СИСТЕМИ РЕГУЛЮВАННЯ ДОРОЖНІМ РУХОМ

3.1. Призначення та функціональні особливості комплексу

Проект реалізує повнофункціональний модуль управління світлофором, що може бути використаний для керування транспортними потоками на перехресті. Модуль не містить силової частини, тобто в проекті використано цифрові лінії контролера Arduino, що дозволяють отримувати струм до 40 мА при напрузі 5 вольт. Цього цілком досить для живлення пар світлодіодів, які моделюють випромінювачі світлофора, що обслуговують однакові напрямки руху. Для реалізації взаємодії з реальним випромінювачем, побудованим на лампах розжарювання чи на потужних світлодіодних матрицях, схему потрібно доповнювати блоками реле або тиристорів для комутації токів живлення випромінювачів. Така модифікація не є складною і може бути виконана у вигляді окремого модуля, що буде доповнювати поточну схему без значної її зміни.

Керування виконується у двох перпендикулярних напрямках. Реалізовано два режима роботи: звичайний та економічний (блимаючий жовтий). Перемикання режимів в виконано перемикачем.

Модель дозволяє регулювати перехрестя, незалежно від кількості випромінюючих вузлів: їх може бути як один, що саме реалізовано в макеті, а може бути й чотири, як зазвичай комплектується перехрестя. Випромінювачі при цьому просто потрібно групувати паралельно підключеними парами.

Всі з’єднання з платою контролера роз’ємні, що дозволяє у разі потреби швидко замінити плану без необхідності використання додаткового інструмента.

3.2. Принципова електрична схема

Через модульність технології Ардуіно скласти даний проект не є важкою задачею. Проект був реалізований у середовищі SPlan 7.0. Сама схема під'єднання показана у додатку А.

SPlan7.0 є однією з найбільш зручних і простих призначених для креслення радіоелектронних та електричних схем.

SPlan дуже зручний у використанні. Додані компоненти просто «перетягуються» з панелі, що знаходиться зліва, праворуч від якої знаходиться панель інструментів для малювання ліній і різних геометричних форм, додавання написів, вставки растрових зображень і т.д. Нумерація компонентам може надаватися як автоматично, так і вручну. Режим можна вибрати правою кнопкою миші через властивості компонента (Properties), у тому ж таки вікні редагуються додаткові атрибути, у тому числі текстове поле (в ньому можна написати, наприклад, номінал даної деталі і т.д.).

Основа принципової електричної схеми проекту – мікроконтролерна платформа Arduino Nano. Фактично з додаткових ресурсів, що їх надає платформа, задіяно лише USB-інтерфейс. Він дозволяє програмувати мікроконтролер, підключаючи його до комп'ютера. Лінії D2-D7, D13 – це безпосередньо виводи ATmega328. Робота випромінювачів моделюється шістьма парами світлодіодів HL1-HL12. Кожна пара регулює трафік одного напрямку, дві пари обслуговують перпендикулярні напрямки:

- HL1, HL2 – червоне світло сторони 1;
- HL3, HL4 – червоне світло сторони 2;
- HL5, HL6 – жовте світло сторони 1;
- HL7, HL8 – жовте світло сторони 2;
- HL9, HL10 – зелене світло сторони 1;
- HL11, HL12 – зелене світло сторони 2.

Світлодіоди вибрані з лінійки CZ-5014, розміром 5 мм.

Резистори R1-R6 обмежують струм через світлодіоди приблизно до 30 мА (рисунок 3.1), що забезпечується підбором номіналу резисторів величиною 33 Ом. Потужність, що розсіюється на резисторі становить 30 міліват, таким чином можна вибрати резистори, що є в продажу – розраховані на мінімальну потужність 0.125 Вт.

Перемикач S1, що забезпечує перехід в економічний режим, підключено за стандартною схемою через підтягуюяий резистор R7.

Рисунок 3.1. – Результати заміру струму через світлодіоди

В проекті використано джерело живлення напругою 5 В. При цьому струм, що теоретично може споживати пристрій не перевищує 0.25 А (50 мА – Arduino, 180 мА – всі 12 світлодіодів).

3.3. Алгоритм роботи програми

Алгоритм роботи світлофора зазвичай базується на стандартних послідовних діях, які виконуються для керування рухом транспорту. Контролер відпрацьовує такі послідовності дій:

1) Конфігурування системи – налаштування задіяних ліній, визначення напрямку роботи пінів
2) Початок роботи. Вимкнення світлодіодів – всі лінії, до яких вони підключені, переводяться в стан з низькою напругою.
3) Перехід в нескінчений цикл.
4) Перевіряємо положення перемикача. Якщо не ввімкнуто, то викликаємо функцію changeLights, де прописана поведінка ліній мікроконтролера:
    a. зелений напрямку 1 – в низький рівень;
    b. жовті в обох напрямках – у високий;
    c. утримання стану на протязі 7 секунд;
    d. жовті в обох напрямках – у низький рівень;
    e. червоний напрямку 1 та зелений напрямку 2 – у високий рівень;
    f. утримання стану на протязі 7 секунд;
    g. жовті в обох напрямках – у високий;
    h. зелений напрямку 2 – в низький рівень;
    i. утримання стану на протязі 3 секунд;
    j. жовті в обох напрямках – у низький рівень;
    k. зелений напрямку 1 та червоний напрямку 2 – у високий рівень;
    l. червоний напрямку 1 – у низький рівень;
    m. утримання стану на протязі 7 секунд;
5) Якщо перемикач увімкнено, то встановимо блимаючий жовтий:
    a. згасимо всі, окрім жовтих;
    b. вмикаємо жовтий в обидві сторони;
    c. утримання стану на протязі півсекунди;

d. вимикаємо жовтий в обидві сторони;
e. утримання стану на протязі півсекунди;
f. повторюємо, поки перемикач увімкнений;

6) Повертаємся на початок головного циклу.

3.4. Опис програмних блоків

Код, наданий нижче, є програмою для Arduino, що реалізує керування світлофором з двох сторін.

Визначення пінів контролера для підключення світлодіодів

```
int r1 = 2;  // Червоний світлодіод сторони 1
int y1 = 4;  // Жовтий світлодіод сторони 1
int g1 = 6;  // Зелений світлодіод сторони 1

int r2 = 3;  // Червоний світлодіод сторони 2
int y2 = 5;  // Жовтий світлодіод сторони 2
int g2 = 7;  // Зелений світлодіод сторони 2
```

Тут визначаються змінні для зберігання номерів пінів, на які підключені світлодіоди кожного кольору з обох боків світлофора.

Визначення піна для перемикача миготливого жовтого світла:

```
int s_pin = 13;  // Пін для перемикача миготливого
                 //жовтого світла
```

У функції setup() відбувається ініціалізація пінів для роботи з підключеними світлодіодами та перемикачем. Піни, що використовуються для світлодіодів, встановлюються в режим виведення (OUTPUT), а пін перемикача – в режим введення (INPUT).

Основна функція loop().

```
void loop() {
int s = digitalRead(s_pin);  // Зчитуємо стан перемикача
```

```
  if (s == LOW) {
    // Блок коду для роботи світлофора без миготливого
        жовтого світла
  } else {
    // Блок коду для роботи світлофора з миготливим
       жовтим світлом
  }
}
```

Блок коду для роботи світлофора без миготливого жовтого світла.

```
digitalWrite(g1, LOW);
digitalWrite(y1, HIGH);
digitalWrite(y2, HIGH);
delay(7000);

digitalWrite(y1, LOW);
digitalWrite(r1, HIGH);
digitalWrite(y2, LOW);
digitalWrite(r2, LOW);
digitalWrite(g2, HIGH);
delay(7000);

digitalWrite(y1, HIGH);
digitalWrite(y2, HIGH);
digitalWrite(g2, LOW);
delay(3000);

digitalWrite(g1, HIGH);
digitalWrite(y1, LOW);
digitalWrite(r1, LOW);
digitalWrite(y2, LOW);
digitalWrite(r2, HIGH);
delay(7000);
```

У цьому блоці коду реалізована послідовність команд для управління світлофором без миготливого жовтого світла. Кожна команда digitalWrite() встановлює стан відповідного світлодіода, а функція delay() затримує виконання програми на вказану кількість мілісекунд.

Блок коду для роботи світлофора з миготливим жовтим світлом.

```
digitalWrite(r1, LOW);
digitalWrite(r2, LOW);
digitalWrite(g1, LOW);
digitalWrite(g2, LOW);

do {
  digitalWrite(y1, HIGH);
  digitalWrite(y2, HIGH);
```

```
    delay(500);
    digitalWrite(y1, LOW);
    digitalWrite(y2, LOW);
    delay(500);
  } while (digitalRead(s_pin) == HIGH);
```

У цьому блоці коду реалізована послідовність команд для роботи світлофора з жовтим світлом, що миготить. Спочатку всі світлодіоди, окрім жовтих, вимикаються. Потім за допомогою циклу do-while миготливі жовті світлодіоди включаються протягом 0,5 секунд, потім вимикаються на 0,5 секунд. Цикл продовжується доти, доки перемикач не буде переведений у положення "вимкнено" (зчитується стан LOW).

Таким чином, код реалізує логіку роботи світлофора з можливістю вибору між звичайним і миготливим жовтим режимом за допомогою перемикача.

ВИСНОВКИ

Напружений трафік сучасних транспортних магістралей потребує впровадження інтелектуальних систем управління дорожнім рухом, що повинні взаємодіяти між собою для створення комплексної системи. Такі системи можна будувати на широко росповсюджених сучасних мікроконтролерних платформах.

В роботі реалізовано повнофункціональний модуль управління випромінювальним блоком світлофора, що може бути бути доповнений довільною силовою частиною, а також дозволяє просту інтеграцію в сімейство подібних систем. Модуль побудовано з використанням контролера Arduino Nano, що дозволяє реалізувати систему з мінімальними затратами, скороченим часом програмування та досить простим обслуговуванням.

В проекті реалізовано керування транспортними потоками у двох перпендикулярних напрямках, а також реалізовано два режима роботи: звичайний та економічний (блимаючий жовтий).

Всі з’єднання з платою контролера роз’ємні, що дозволяє у разі потреби швидко замінити плану без необхідності використання додаткового інструмента.

Проект "Розробка мікропроцесорної системи регулювання дорожього руху на перехресті на базі Arduino Nano" демонструє можливості використання мікроконтролерів та розумних систем для покращення регулювання дорожнього руху. Він може бути застосований у містах, щоб зменшити затори та покращити безпеку на дорогах.

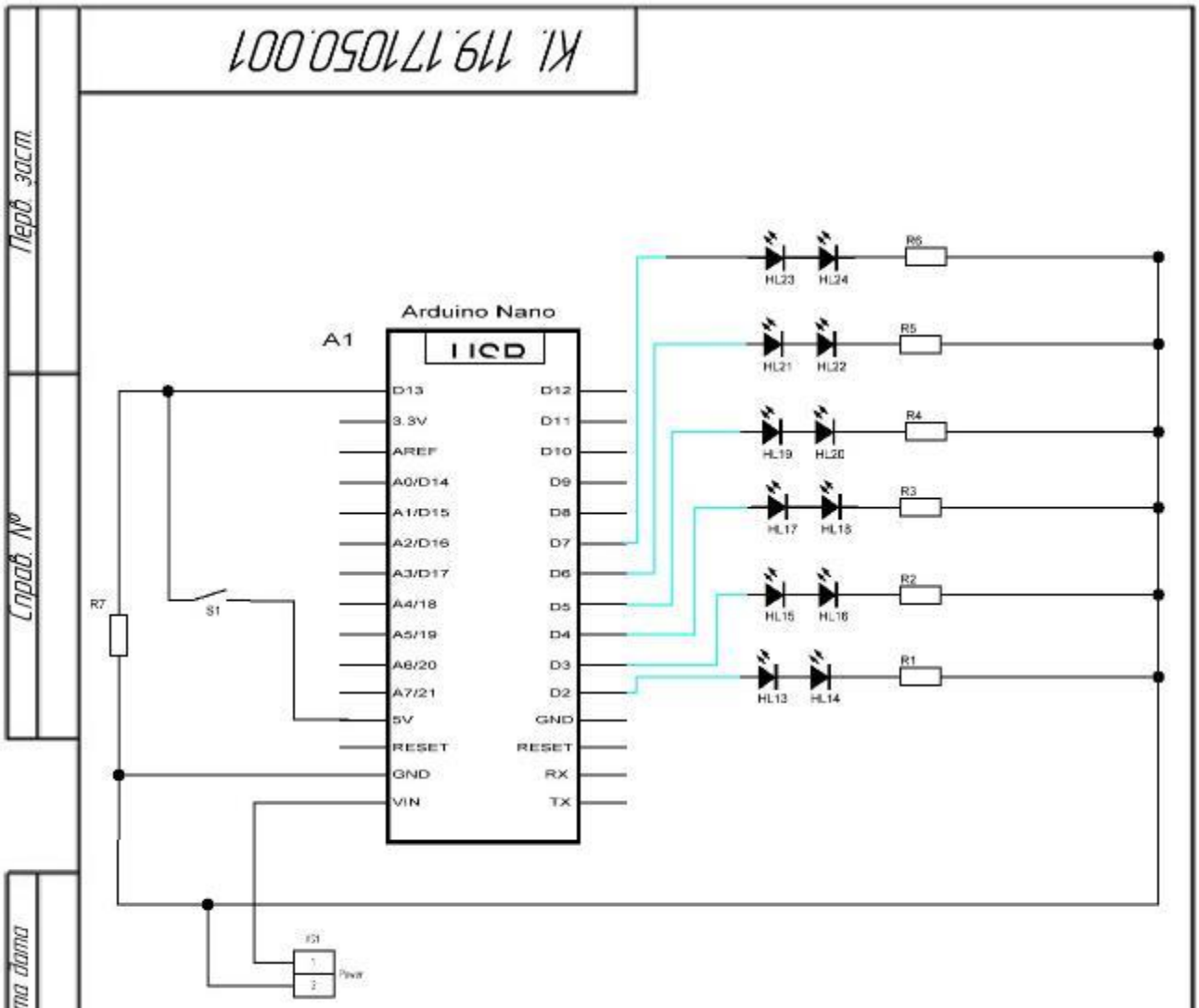
ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Система керування дорожнім рухом [Електронний ресурс] / Режим доступу: www. URL: https://pidru4niki.com/81363/tehnika/avtomatizovani_sistemi_keruvannya_dorozhnim_ruhom – 01.04.23р.
2. Вимоги до управління дорожнім рухом) [Електронний ресурс] / Режим доступу: www. URL: https://vlp.com.ua/node/18558 –01.04.23р.
3. SEA TCS (Traffic Control System) [Електронний ресурс] / Режим доступу: www. URL: https://www.sea.com.ua/ua/smart-city/kompleksnaa-sistema-upravlenia-doroznym-dvizeniem-sea-ksudd/ – 01.04.2023 р.
4. Интеллектуальное управление дорожным движением. [Електронний ресурс] / Режим доступу: www. URL: https://www.mallenom.ru/resheniya/po-otrasliam/road-manager/ – 01.04.2023 р.
5. Росток-ЕЛЕКОМ. [Електронний ресурс] / Режим доступу: www. URL: http://rostok-elekom.com/ – 02.04.2023р.
6. Urban Traffic Management by SWARCO [Електронний ресурс] / Режим доступу: www. URL: https://www.swarco.com/products/software/urban-traffic-management – 02.04.2023 р.
7. Welcome to Traficon! [Електронний ресурс] / Режим доступу: www. URL: https://traficon.net/ – 02.04.2023 р.
8. Ардуіно Нано [Електронний ресурс] / Режим доступу: www. URL: https://arduino.ru/Hardware/ArduinoBoardNano – 02.04.23р.
9. Особливості реалізації Arduino [Електронний ресурс] / Режим доступу: www. URL: https://tproger.ru/curriculum/arduino-quick-start/ – 03.04.23р.
10. Базовий мікроконтролер [Електронний ресурс] / Режим доступу: www. URL: https://gyrator.ru/mikrokontroller-osnovy – 06.04.23р.

11. Технічні характеристики платформи [Електронний ресурс] / Режим доступу: www. URL: https://arduino.ru/Hardware/ArduinoBoardUno – 08.04.23р.
12. Ресурси платформи, що використані в проекті [Електронний ресурс] / Режим доступу: www. URL: https://arduino.ru/Arduino_environment – 12.04.23р.
13. Середовище розробки Arduino IDE [Електронний ресурс] / Режим доступу: www. URL: https://doc.arduino.ua/ru/guide/Environment – 15.04.23р.
14. Реалізація багатозадачності в Arduino [Електронний ресурс] / Режим доступу: www. URL: https://alexgyver.ru/lessons/how-to-sketch/ – 18.04.23р.
15. Створення об'єктів і класів в Arduino [Електронний ресурс] / Режим доступу: www. URL: https://robotraffik.ru/page/roborace-project-3/ – 21.04.23р.

Додаток А

Принципова електрична схема



					КІ. 119.171050.001		
Змн.	Лист	№ докум.	Підп.	Дата	Схема електрична принципова	Літ.	Маса	Масштаб
Розроб.		Попова К.О.		27.05		У		
Перев.		Сабанов С.О.		27.05		Лист	Листов 1	
Т. контр.								
Н. контр.						ЗІЕІТ КІ-119		
Затв.		Сабанов С.О.		27.05				

Перед. заст.
Справ. №
Підп. та дата
Інв. № дубл.
Взам. інв. №
Підп. та дата
Інв. № подл.

Додаток Б

Перелік елементів

Визн.	Найменування	К-сть	Примітка
	Резистори		
R1-6	33 Ом, 0.125W, 5%	6	
R7	10 kOm, 0.125W, 5%	1	
	Діоди		
HL1-4	CZ-5014SURC, червоний	4	
HL5-8	CZ-5014SUYC, жовтий	4	
HL2-12	CZ-5014SPGC, зелений	4	
	Контролер		
A1	Arduino Nano (ATmega328)	1	
	Перемикач		
S1		1	

Перев. заст.

Справ. №

Підп. та дата

Інв. № дубл.

Взам. інв. №

Підп. та дата

Інв. № подл.

Зм.	Лист	№ докум.	Підп.	Дата
Розроб.		Попова К.О.		27.05
Перев.		Сабанов С. О		27.05
Т. контр.				
Н. контр.				
Затв.		Сабанов С. О		27.05

КІ. 119.171050.001

Перелік елементів

Літ	Маса	Масштаб
У		
Лист	Листів 1	

ЗІЕІТ КІ-119

Додаток В

Код проекту

```
//r - red, y - yellow, g - green
//r1,y1,g1 - кольори боку 1
//r2,y2,g2 - кольори сторони 2
// Виводи Ардуїно для сторони 1
int r1 = 2;
int y1 = 4;
int g1 = 6;

// Виводи Ардуїно для сторони 2
int r2 = 3;
int y2 = 5;
int g2 = 7;

//Переключатель миготливого жовтого - вивод 8
int s_pin = 8;

void setup() {
// Трафік з боку 1
pinMode (r1, OUTPUT);
pinMode (y1, OUTPUT);
pinMode (g1, OUTPUT);

// Трафік з боку 2
pinMode (r2, OUTPUT);
pinMode (y2, OUTPUT);
pinMode (g2, OUTPUT);

//Пін перемикача встановлюємо на вхід
pinMode (s_pin, INPUT);
}
void loop() {
   //Перевіримо положення перемикача:
   int s = digitalRead (s_pin);
   //Якщо не вкл, то:
   if (s == LOW) {
// Загораються обидва жовті індикатори
digitalWrite(g1, LOW);
digitalWrite(y1, HIGH);
digitalWrite(y2, HIGH);
delay (7000);

// Вимикає жовтий і вмикає r1 і g2
digitalWrite(y1, LOW);
digitalWrite(r1, HIGH);
digitalWrite(y2, LOW);
digitalWrite(r2, LOW);
digitalWrite(g2, HIGH);
```

```
delay (7000);

// Вмикаються обидва жовті індикатори
digitalWrite(y1, HIGH);
digitalWrite(y2, HIGH);
digitalWrite(g2, LOW);
delay (3000);

// Вимикає жовте світло і вмикає g1 і r2
digitalWrite(g1, HIGH);
digitalWrite(y1, LOW);
digitalWrite(r1, LOW);
digitalWrite(y2, LOW);
digitalWrite(r2, HIGH);
delay (7000);
}
//Якщо перемикач увімкнений, то встановимо миготливий жовтий:

   else {
// Вимикаємо усе, крім жовтого
digitalWrite(r1, LOW);
digitalWrite(r2, LOW);
digitalWrite(g1, LOW);
digitalWrite(g2, LOW);
do {
//Вмикаємо жовтий в обидві сторони
digitalWrite(y1, HIGH);
digitalWrite(y2, HIGH);
delay(500);
digitalWrite(y1, LOW);
digitalWrite(y2, LOW);
delay(500);
}
//Продовжуємо цикл миготливого жовтого, поки перемикач замкнуть:
while (digitalRead (s_pin) == HIGH) ;

}
}
```