Міністерство освіти і науки України

Луцький національний технічний університет
(повне найменування закладу вищої освіти)

Факультет комп'ютерних та інформаційних технологій
(повне найменування факультету)

Кафедра комп'ютерної інженерії та кібербезпеки
(повне найменування кафедри)

КВАЛІФІКАЦІЙНА РОБОТА ЗА СТУПЕНЕМ ВИЩОЇ ОСВІТИ «БАКАЛАВР»

РОБОТ З ДИСТАНЦІЙНИМ КЕРУВАННЯМ НА ПЛАТФОРМІ ARDUINO UNO

ROBOT WITH REMOTE CONTROL ON THE ARDUINO UNO PLATFORM

спеціальність 123 Комп'ютерна інженерія
(шифр і назва спеціальності)

освітня програма Комп'ютерна інженерія
(назва освітньої програми)

Виконав: здобувач вищої освіти
групи КІс-21
Гриценюк Віталій Віталійович

(підпис)

Керівник:
к.т.н., старший викладач
Гринюк Сергій Васильович

(підпис)

Кваліфікаційну роботу
допущено до захисту
« _____ » червня 2023 р.
Гарант освітньої програми:
к.т.н., доцент
Лавренчук Світлана Василівна

(підпис)

Луцьк – 2023 року

ЛУЦЬКИЙ НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ

Факультет *комп'ютерних та інформаційних технологій*
Кафедра *комп'ютерної інженерії та кібербезпеки*
Ступінь вищої освіти: *бакалавр*
Галузь знань: *12 Інформаційні технології*
Спеціальність: *123 Комп'ютерна інженерія*
Освітня програма: *«Комп'ютерна інженерія»*

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ проф. Н.Чернящук
«_____» _____ 2023 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ ЗДОБУВАЧУ ВИЩОЇ ОСВІТИ

Гриценюку Віталію Віталійовичу
(прізвище, ім'я, по батькові)

1. Тема кваліфікаційної роботи *Робот з дистанційним керуванням на платформі Arduino UNO*

Керівник роботи *Гринюк Сергій Васильович, к.т.н., старший викладач*

затверджені наказом закладу вищої освіти від «28» грудня 2022 року № 982/01-02
2. Строк подання здобувачем вищої освіти кваліфікаційної роботи 01.06.2023р.

3. Вихідні дані до роботи *Джерелом розробки є науково-технічна література та публікації в періодичних виданнях з даного питання, опубліковані зарубіжні та вітчизняні роботи в даній області та різні інтернет-ресурси технічного спрямування*

4. Зміст пояснювальної записки (перелік питань, які потрібно розробити):
Вступ
Аналіз предметної галузі
Вибір апаратної платформи та середо вища розробки програмного забезпечення
Розробка програмно-апаратних засобів керування роботом
Висновки

5. Перелік графічного (ілюстративного) матеріалу:
Плата Arduino UNO R3
Інтегроване середовище розробки Arduino(IDE)
Модуль Bluetooth HC-05
Макетна схема підключення елементів робота
Блок-схема дистанційного керування роботом
Інтерфейс програми для дистанційного управління

АНОТАЦІЯ

Гриценюк В.В. Робот з дистанційним керуванням на платформі Android UNO. Рукопис.

Кваліфікаційна робота бакалавра ОП «Комп'ютерна інженерія» спеціальності 123 Комп'ютерна інженерія. Луцький національний технічний університет. Луцьк, 2023.

Кваліфікаційна робота складається з вступу, трьох розділів, висновків, списку використаних джерел, одного додатку.

У першому розділі розглядається один з актуальних об'єктів дослідження в галузі робототехніки і штучного інтелекту – мобільний робот. Проаналізовано технологій, що дозволяють керувати рухами і функціями робота з віддаленої точки. Це може бути здійснено за допомогою різних методів, таких як бездротове зв'язку, інтернет, супутникові з'єднання, радіочастоти та інші.

У другому розділі проведено опис апаратної платформи та комплектуючих, а саме Arduino UNO та середовища розробки програмного забезпечення – Arduino IDE.

У третьому розділі було проведено безпосередню розробку апаратних та програмних засобів робота та тестування засобів керування ним..

Об'єкт дослідження – робот з дистанційним керування на базі Arduino Uno.

Предмет дослідження – технології створення та керування мобільними роботами.

Метою роботи є розробка робота з дистанційним керуванням з використанням сучасних підходів в методах керування та програмування робототехнічних систем.

Ключові слова: мобільний робот, Arduino UNO, Bluetooth, мікроконтролер, MIT App Inventor 2, дистанційне керування, Android.

ANNOTATION

Hrytsenyuk V.V. Robot with remote control on the Android UNO platform. Manuscript.

Bachelor's qualifying thesis of the OP "Computer Engineering" specialty 123 Computer Engineering. Lutsk National Technical University. Lutsk, 2023.

The qualification work consists of an introduction, three sections, conclusions, a list of used sources, and one appendix.

The first chapter examines one of the current objects of research in the field of robotics and artificial intelligence - a mobile robot. The technologies that allow controlling the movements and functions of the robot from a remote point are analyzed. This can be done using various methods such as wireless communication, internet, satellite connections, radio frequencies and others.

The second chapter describes the hardware platform and components, namely Arduino UNO and the software development environment - Arduino IDE.

In the third section, the direct development of the robot's hardware and software and testing of its controls were carried out.

The research object is a robot with remote control based on Arduino Uno.

The subject of research is the technology of creating and managing mobile robots.

The purpose of the work is to develop a robot with remote control using modern approaches in methods of control and programming of robotic systems.

Keywords: mobile robot, Arduino UNO, Bluetooth, microcontroller, MIT App Inventor 2, remote control, Android.

ЗМІСТ

ВСТУП

Актуальність теми дослідження полягає в тому, що в сучасному світі все більше зазнають широке впровадження комп'ютеризовані та робототехнічні системи в усіх сферах людської діяльності.

Системи дистанційного управління виконують важливу роль для зменшення ризиків, які виникають під час людської праці. Їх застосування у робототехніці дозволяє знизити ступінь участі людини при проведенні робіт у небезпечних умовах. Вони дозволяють отримувати безпечний доступ до важкодоступних або небезпечних зон, проводити оперативне втручання мобільних роботів (МР) у виробничі процеси з високим рівнем небезпеки або для оптимізації умов праці. Також, треба відзначити користь використання роботів під час проведення вибухотехнічних робіт (пошук, транспортування, знешкодження або знищення вибухонебезпечних предметів і боєприпасів). Тому розробка та удосконалення сучасної системи управління мобільними роботам за допомогою мобільного додатку – є актуальною.

Основними переваги використання МР для вищезазначених задач є:

- зручність органів управління щодо можливості швидкого здійснення будь-яких маневрів;
- простота навчання операторів, які мають керувати МР;
- суттєвий час автономної роботи МР, що важливо для довготривалих задач;
- значний радіус дії та можливість використання надійного зв'язку, що є критичним параметром під час вибухотехнічних операцій;
- економічність системи, як у цілому, так і в обслуговуванні, зважаючи на те, що електронні компоненти МР можуть виходити з ладу через фізичні пошкодження або під впливом часу.

З кожним роком зростає потреба використання МР у різних сферах праці [1]. Що, у свою чергу, обумовлює актуальність завдань розробки або удосконалення систем управління МР, в тому числі на виробництві [2].

Для організації ліній зв’язку об’єкта управління і оператора використовують різноманітні канали зв’язку. Наприклад, такі як GSM, Bluetooth та Wi-Fi, які входять у сукупність технологій Internet of Things. Ці технології дуже поширені, гнучкі у використанні, енергоефективні та економічні, що спрощує процес проектування та розробки системи дистанційного управління.

Метою роботи є розробка робота з дистанційним керуванням з використанням сучасних підходів в методах керування та програмування робототехнічних систем.

Об'єкт дослідження – робот з дистанційним керування на базі Arduino Uno.

Предмет дослідження – технології створення та керування мобільними роботами.

Були сформовані наступні цілі на кваліфікаційну роботу:

– проведення аналізу сучасного стану проблеми дистанційного управління МР;

– вибір технології для реалізації зв’язку системи дистанційного управління з МР;

– розробка програмної реалізації системи дистанційного управління МР.

– розробити прототип МР на дистанційному управлінні.

РОЗДІЛ 1
АНАЛІЗ ПРЕДМЕТНОЇ ГАЛУЗІ

1.1 Мобільний робот як об’єкт дослідження

Мобільний робот (МР) – це робот, який може пересуватися і переміщатися в просторі [3]. Розрізняюсь три класи мобільних роботів, а саме:

– наземні роботи;

– повітряні;

– морські.

Серед наземних роботів виділяють наступні класи:

– колісні роботи;

– гусеничні роботи;

– крокуючі роботи.

Одним з найпоширеніших та найдешевших конструкцій як у виконанні та програмуванні є колісні роботи. Колісні МР використовуються для переміщення по більшості поверхонь, таких як гладкі та нерівні поверхні, наприклад: асфальт, ґрунтові дороги, бетонні покриття, пересічна місцевість та інші. Окремо можна зауважити що гусеничні роботи мають також немало переваг при пересуванні по різним видам поверхонь, але висока пляма контакту гусениць призводить до ускладненого використання при переміщеннях на рівних гладких поверхнях та килимах. Колісні роботи (КР) розрізняють за кількістю коліс. Найчастіше зустрічаються двоколісні, триколісні чотириколісні та шестиколісні роботи.

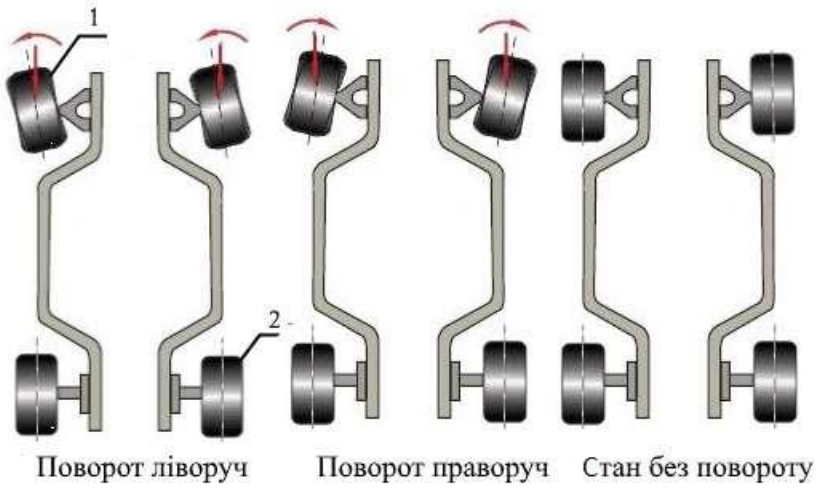
Розглянемо колісних роботів на прикладі вже існуючих.

Micro:bit Smart – найпростіший приклад колісного робота, який має три колеса, з яких два є ведучими, а також за їх рахунок відбувається керування, тому що вони є незалежними один від одного, а третє колесо є опорним [4]. Управління роботом здійснюється за рахунок контролю потужності яка поступає на колекторні мотор-редуктори. Тобто подаючи на відповідний двигун більшу потужність, колесо до якого поступає більший круговий момент крутиться

швидше і відбувається поворот конструкції. Проте така конструкція має недоліки, найбільші з яких не фіксоване положення опорного ролику, через що конструкції складно триматися заданої траєкторії, а також нерівномірний розподіл маси, через що робот може легко перевернутись. На рисунку 1.1 зображено колісного робота Micro:bit Smart.

Рисунок 1.1 – Загальний вигляд робота Micro:bit Smart

Найбільш розповсюдженим прикладом КР є роботи на чотирьох колесах, два з яких ведучі, інші два направляючі. У таких конструкціях за напрям руху відповідають направляючі колеса. В залежності від напрямку повороту направляючих коліс відбувається рух КР у заданий напрям. Данні конструкції направляючі зустрічаються людині у повсякденному житті. Наприклад, легкові автомобілі та іграшкові машинки на радіо-керуванні. Узагальнений вигляд чотириколісного шасі з наведеним принципом роботи направляючих коліс приведено на рисунку 1.2.

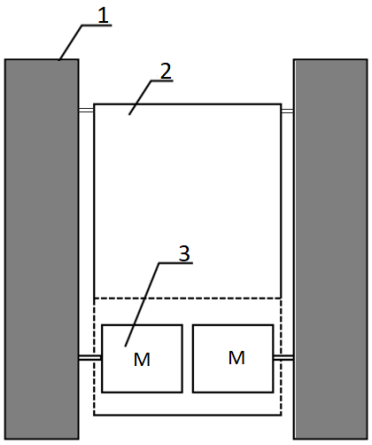


1 – направляюче колесо; 2 – ведуче колесо

Рисунок 1.2 – Загальний вигляд чотириколісного шасі

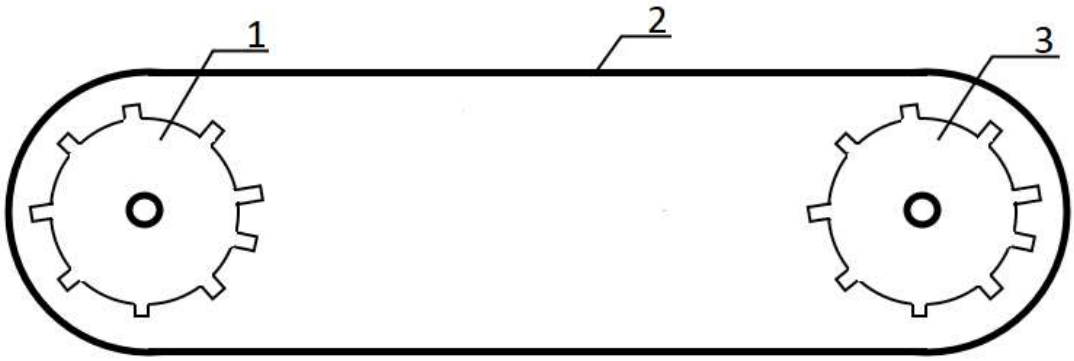
Порівняно з чотири колісними шасі, значну нішу серед роботів займають МР на базі гусеничних шасі (ГШ). До руху такі шасі приводять зазвичай два електромотори, які прокручують гусеничні стрічки. Усього у ГШ є дві гусеничні стрічки.

На відміну від колісних шасі, ГШ має як мінімум два ролики, два з яких виконують роль направляючих та привідних одночасно, як це виконується на триколісному шасі, а два інших виконують роль опорних роликів, які також підтримують гусениці у натягнутому стані. Загальний вигляд ГШ та ланки приведено на рисунках 1.3 та 1.4.



1 – гусенична ланка; 2 – платформа; 3 – двигун

Рисунок 1.3 – Приклад загального вигляду гусеничного шасі



1 – опорний ролик; 2 – гусенична стрічка; 3 — привідний ролик

Рисунок 1.4 – Узагальнена схема гусеничної ланки

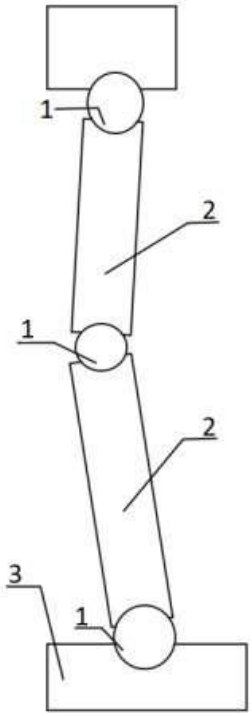
Перевагою ГШ є підвищена прохідність по пересічній місцевості, особливо де потребується найменший тиск на ґрунт. Зменшення тиску досягається через велику площа дотику гусениць з поверхнею. У разі проїзду робота через м’які ґрунти зменшується глибина занурення у поверхню.

Гнучкість гусениць значно підвищує прохідність МР через кам'янисті та рельєфні скалисті поверхні. Однією з найбільший переваг ГШ є можливість розвороту як на місті, так і відносно однієї з осей гусениці.

Основними недоліками ГШ є те що вони потребують більшу потужність двигунів, тому що потрібно набагато більше кругового моменту сили щоб прокручувати гусениці, також ГШ не є ефективним рішенням у випадку використання роботів на гладких рівних поверхнях.

Останній вид наземних роботів це крокуючі роботи. Даний тип є дорогим у виготовлені та складним у програмуванні, тому, що потребує високої точності руху ланок. Для цього використовують сервоприводи, вони же – крокові двигуни (КД), вони дешеві під час обслуговування та експлуатації. Окремо можна виділити пневмопривідні та гідропривідні механізми, але вони не надають високої точності.

На рисунку 1.5 приведена схема ноги крокуючого робота.



1 – кроковий двигун; 2 – ланка ноги; 3 – опора ноги

Рисунок 1.5 – Схематичне зображення ноги крокуючого робота

Окрім складності та високого кошторису виділяють важливий недолік крокуючої конструкції робота, це стійкість. Для підвищення стійкості потрібно додаткові датчики нахилу, або гіроскоп, постійно коригувати положення робота, що призводить до більших витрат енергії під час роботи крокуючого роботу.

1.2 Дистанційне керування роботом

Дистанційне керування МР здійснюється різними типами зв’язку.

Наприклад, радіохвилями різних діапазонів частот, довжини хвилі, та інше. До таких методів можна віднести засоби концепції Internet of Things (IoT). IoT, або інтернет речей – це концепція обчислювальної мережі фізичних предметів (тобто власне, речей), які оснащені деякими технологіями для взаємодії один з одним [5]. Зазвичай взаємодія робота з оператором відбувається за допомогою електронно-обчислювальної машини (ЕОМ). Зважаючи на те, що управління МР за допомогою технологій доступних на мобільному пристрої:

– Bluetooth;

– Wi-Fi;

– GSM.

Для розуміння розглянемо детальніше кожну з технологій.

Bluetooth – виробнича специфікація бездротових персональних мереж Bluetooth забезпечує обмін інформацією між такими пристроями, як персональні комп'ютери, смартфони, принтери, та іншими пристроями, на надійній, повсюдно доступній радіочастоті для ближнього зв'язку. Bluetooth дозволяє цим пристроям утримувати зв’язок, коли вони знаходяться в радіусі видимості один від одного [6]. У таблиці 1.1.

приведено дальність зв’язку відповідно до версії специфікації.

Таблиця 1.1 – Дальність передачі даних специфікацій Bluetooth

Версія Bluetooth	Дальність зв’язку, м
3.0	8 – 10
4.0	10 – 20
5.0	40 – 1500

Дальність зв’язку значно змінюється при появі перешкод і завад. Принцип дії заснований на використанні радіохвиль у вільному від ліцензування діапазоні від 2,4 ГГц до 24,835 ГГц. В наш час така топологія використовується у різноманітних побутових пристроях та бездротових мережах.

Wi-Fi – технологія бездротової локальної мережі з пристроями на основі стандартів IEEE 802.11 [11]. IEEE 802.11 – це набір стандартів зв’язку в бездротових мережах, які працюють на частотах 0,9 ГГц, 2,4 ГГц, 3,6 ГГц а також 5 ГГц. Технологія Wi-Fi працює на частотах 2,4 ГГц та 5 ГГц та має декілька режимів роботи, в залежності від яких змінюється швидкість передачі даних у локальній мережі. На сьогоднішній день найпоширенішим режимом роботи є Wi-Fi 802,11n. Дальність звя’зку до 50 м, проте сильно змінюється залежно від якості комплектуючих маршрутизатору, від завад та перешкод [7].

GSM – технологія сотового зв’язку з розподіленням каналу та високим рівнем безпеки за рахунок шифрування з відкритим ключем [8]. За рахунок

ретрансляції такий тип з’єднання дозволяє отримати зв’язок по усій планеті земля, проте він має значну кількість недоліків для дистанційного керування.

Серед недоліків можна виділити:

– неможливість зв’язку P2P;

– значна затримка для миттєвого реагування;

– високий кошторис використання.

РОЗДІЛ 2
ВИБІР АПАРАТНОЇ ПЛАТФОРМИ ТА СЕРЕДОВИЩА РОЗРОБКИ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

2.1 Огляд Arduino UNO

Arduino/Genuino Uno – це плата мікроконтролера на базі процесора ATmega328P. Максимальна довжина і ширина друкованої плати Uno становить 6.9 см і 5.4 см відповідно, з урахуванням роз'єму USB і роз'єму живлення, які виступають за межі плати. Вона має 14 цифрових входів / виходів (з яких 6 можна використовувати як ШІМ-виходи), 6 аналогових входів, чіп з частотою роботи 16 МГц, два роз'єми: силовий і USB, роз'єм ISCP і кнопку перезавантаження пристрою (рис. 2.1). Для стабільної роботи плату необхідно підключити до живлення або через вбудований USB-роз'єм, або підключивши роз'єм живлення до джерела від 7 до 12В [9].

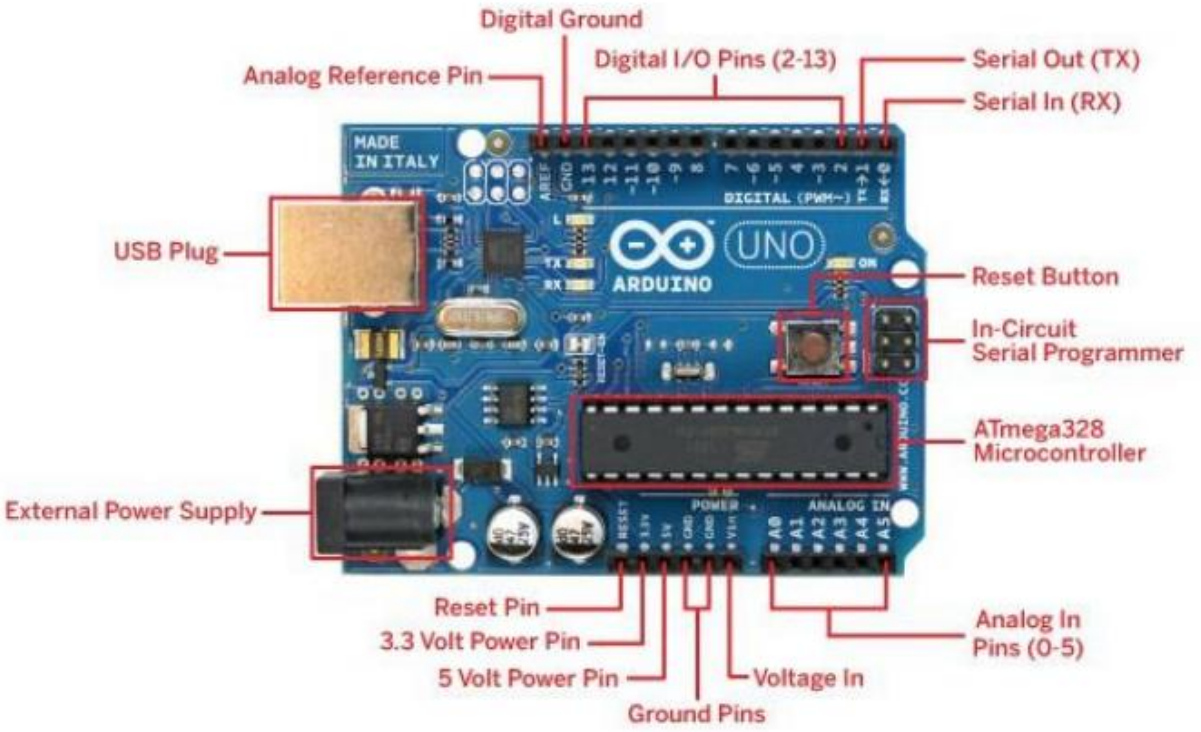


Рисунок 2.1 – Плата Arduino UNO R3

На відміну від всіх попередніх плат Ардуіно, Uno в якості перетворювача інтерфейсів USB-UART використовує мікроконтролер ATmega16U2 (ATmega8U2 до версії R2) замість мікросхеми FTDI [10].

На рисунку 2.2 зображено найменування входів/виходів Arduino Uno

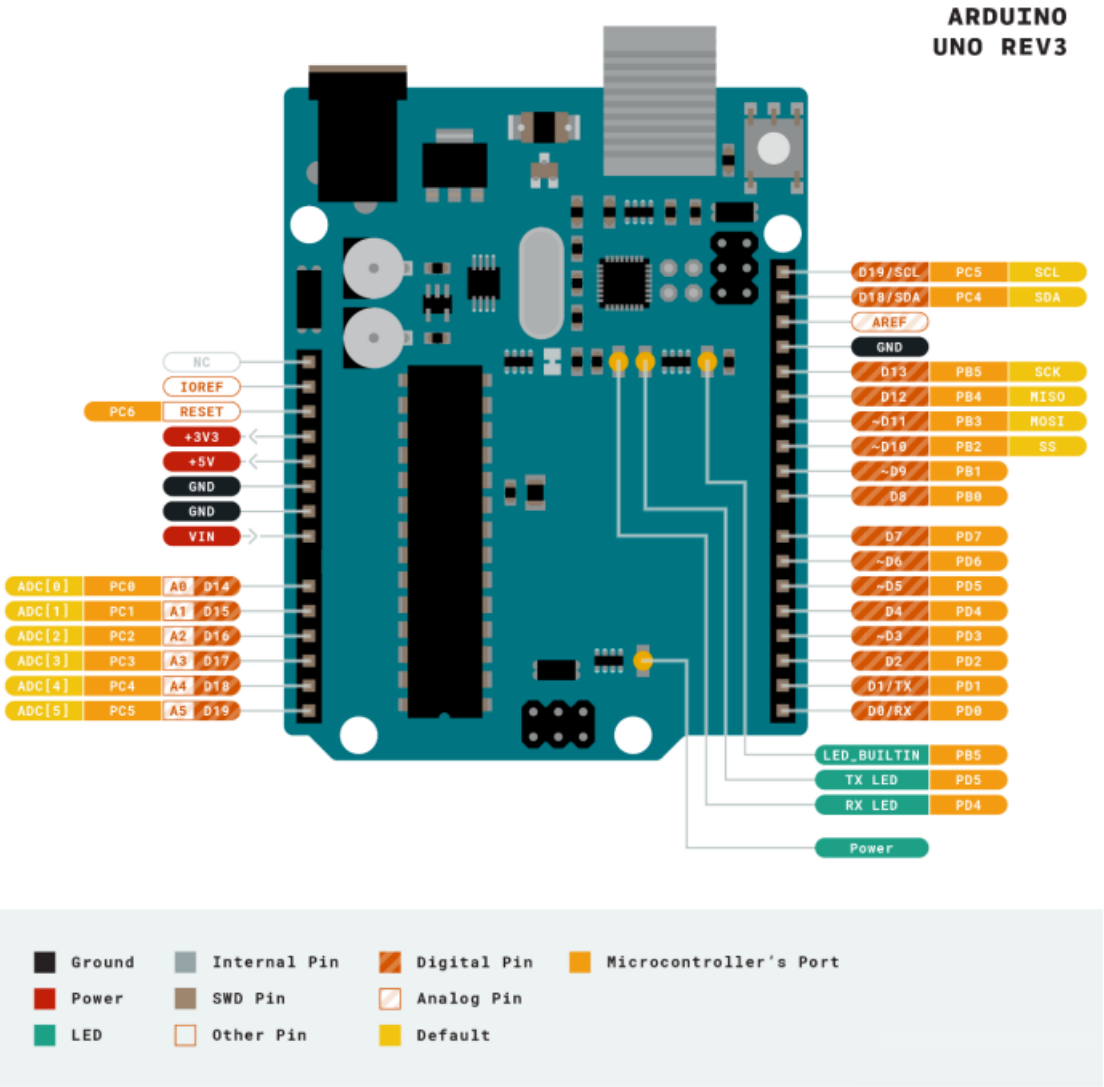


Рисунок 2.2 – Найменування входів та виходів на платі Arduino Uno

2.1.1 Живлення

На платі передбачено кілька портів, що дозволяють живити від неї підключені датчики, сенсори і актуатори. Всі ці порти позначені:

– Vin. Вхід живлення. Використовується для отримання живлення від зовнішнього джерела. Через даний порт відбувається тільки подача живлення на плату, отримати звідти живлення для зовнішніх пристроїв неможливо

– 5V. Джерело 5В напруги для живлення зовнішніх пристроїв. При отриманні живлення платою з будь-яких інших джерел (USB, роз'єм живлення або Vin) на цьому контакті ви завжди зможете отримати 25 стабільну напругу 5 вольт. Його можна вивести на макетну плату або подати безпосередньо на свій пристрій.

– 3V3. Джерело 3.3 В напруги для живлення зовнішніх пристроїв. Працює за таким-же принципом, що і контакт 5V. З даної ніжки також можна вивести напруга на макетну плату, або подати на необхідний датчик / сенсор напряму. Максимальний обсяг струму становить 50 мА.

– GND. Контакт для підключення заземлення. Необхідний для створення замкненого кола при підключенні до контактів Vin, 5V або 3V3. У всіх випадках ніжку GND необхідно виводити як мінус, інакше ланцюг не буде замкнутий і живлення не буде подаватися.

– IOREF. Цей порт на платі Arduino/Genuino забезпечує еталон напруги, з яким працює мікроконтролер. Залежно від напруги на порту IOREF, плата може переключитися на відповідне джерело живлення або задіяти перетворювачі рівнів, що дозволить їй працювати як з 5В, так і з 3.3В пристроями.

Для захисту USB порту комп'ютера від зворотних струмів, короткого замикання і перенавантаження, на платформі Arduino Uno вбудований автоматичний самовідновлюваний запобіжник. При проходженні струму харчування більше 500 мА через USB порт, запобіжник автоматично спрацьовує і розмикає ланцюг живлення до тих пір, поки значення струму не повернутися до нормальних.

2.1.2 Пам'ять

Плата Arduino Uno за замовчуванням підтримує три типи пам'яті:

– Flash-пам'ять об'ємом 32 кБ. Це основне сховище для команд. Коли ви прошиває контролер своїм скетчем, він записується саме сюди. 0.5кб з даного

пулу пам'яті відводиться на bootloader - програму, яка займається ініціалізацією системи, завантаження через USB і запуску скетчу.

– Оперативна SRAM пам'ять об'ємом 2 кБ. Тут по-замовчуванню зберігаються змінні і об'єкти, створювані в ході роботи програми. Пам'ять ця енерго-залежна, при виключенні живлення всі дані, зрозуміло, зітруться.

– Незалежна пам'ять (EEPROM) обсягом 1кб. Тут можна зберігати дані, що не зітруться при виключенні контролера. Але процедура запису і зчитування EEPROM вимагає використання додаткової бібліотеки, яка доступна в Arduino IDE за замовчуванням. Також ніжно пам'ятати про обмеження циклів перезапису, властивих технології EEPROM.

2.1.3 Порти

Порти з номерами від 0 до 13 є цифровими. Це означає, що ви можете зчитувати і подавати на них тільки два види сигналів: HIGH (логічна 1) і LOW (логічний 0). HIGH сигнал розпізнається Arduino як струм напругою 5В, а LOW сигнал, відповідно 0В. Кожен з 14 цифрових контактів на Uno може використовуватися як вхід або вихід. Кожен порт може забезпечувати або приймати 20 мА, як рекомендовано, і має внутрішній підтягуючий резистор (відключений за замовчуванням) 20-50 кОм. Максимум 40 мА - це значення, яке не можна перевищувати на жодному контакті вводу-виводу, щоб уникнути постійних пошкоджень мікроконтролера.

Arduino Uno має на своїй платформі 6 аналогових входів з дозволом 10 Біт на кожен вхід. Даний дозвіл говорить нам про те, що сигнал, що приходить на нього, можна оцифрувати в діапазоні від 0 до 1024 умовних значень.

Окрім цього деякі порти можуть виконувати додаткові функції. 0(RX) і 1(TX) цифрові порти послідовного інтерфейсу використовуються для надсилання та отримання інформації через послідовний інтерфейс (засіб зв'язку з іншими пристроями). Порти 2 та 3 можуть використовуватися для виклику переривань в роботі мікроконтролера при низькому рівні сигналу на цих контактах. Цифрові контакти 3, 5, 6, 9, 10, 11 можуть виводити 8-бітні аналогові

значення у вигляді ШІМ- 27 сигналу. На порту 13 є вбудований світлодіод, який включається при отриманні HIGH і відключається при отриманні LOW сигналу.

2.1.4 Зв'язок

Arduino Uno надає ряд можливостей для здійснення зв'язку з комп'ютером, ще одним Arduino або іншими мікроконтролерами. У ATmega328 є приймач UART, що дозволяє здійснювати послідовний зв'язок за допомогою цифрових портів 0 (RX) і 1 (TX). Мікроконтролер ATmega16U2 на платі забезпечує зв'язок цього приймача з USB-портом комп'ютера, і при підключенні до ПК дозволяє Ардуіно визначатися як віртуальний COM-порт. Прошивка мікросхеми 16U2 використовує стандартні драйвера USB-COM, тому установка зовнішніх драйверів не потрібна. У пакет програмного забезпечення Ардуіно входить спеціальна програма, що дозволяє зчитувати і відправляти на Ардуіно прості текстові дані. При передачі даних через мікросхему-перетворювач USB-UART під час USB-з'єднання з комп'ютером, на платі будуть мигати світлодіоди RX і TX. Також є вбудована бібліотека SoftwareSerial дозволяє реалізувати послідовний зв'язок на будь-яких цифрових висновках Arduino Uno, якщо вбудовані порти 0 та 1 будуть використовуватись для інших потреб.

2.2 Програмування плати Arduino

Для написання програм (скетчів) для контролера Ардуіно вам потрібно встановити середовище програмування. Найпростішим варіантом буде установка безкоштовної Arduino IDE, скачати її можна з офіційного сайту. Сам скетч найчастіше представляє собою нескінченний цикл, в якому регулярно перевіряються порти з приєднаними до них датчиками і за допомогою спеціальних команд формується керуючий вплив на зовнішні пристрої (вони включаються або вимикаються). У програміста Ардуіно є можливість підключити готові бібліотеки, як вбудовані в IDE, так і доступні на численних сайтах і форумах [11].

Написана і скомпільована програма завантажується через USB-з'єднання (UART- Serial). З боку контролера за цей процес відповідає bootloader.

Arduino програмується на мові програмування C/C ++ з відповідним йому синтаксисом. Вбудований складальник, препроцесор і компілятор виправляють велику кількість помилок і робить багато за користувача автоматично, ми навіть про це не знаємо і не замислюємося. Базові функції для управління портами і інтерфейсами мікроконтролера, математика і деякі інші функції/макроси взяті з відкритого фреймворка для роботи з мікроконтролерами під назвою Wiring. Саме з нього складається базовий набір інструментів Ардуіно. У зв'язку з цим самі розробники Arduino називають мову «спрощеним C++», і навіть дали йому окрему назву – Arduino Wiring. З самого початку в Arduino IDE нам доступна величезна купа різних функцій і інструментів: Всі можливості мови C ++, які надає компілятор: типи даних, оператори і взагалі весь неосяжний синтаксис. Ми програмуємо на тому ж C ++, на якому можна програмувати в будь-якому іншому місці. «Ядро» Arduino – бібліотека Arduino.h, яка автоматично підключається в код. У ній містяться функції для управління пінами, інтерфейсами, а також є набір всяких корисних функцій і інструментів. А ще воно відповідає за ініціалізацію і настройку периферії мікроконтролера при запуску. В ядрі, до речі, лежать стандартні бібліотеки для Serial, Wire, SPI і EEPROM. В папці з програмою лежить набір стандартних бібліотек: для LCD дисплея, крокового мотора, сервоприводу і деяких інших приладів. З компілятором йде набір низькорівневих бібліотек для AVR. Компілятор дозволяє працювати з мікроконтролером «безпосередньо» за допомогою регістрів і читання даташіта. Також ми можемо писати на асемблері, взявши під контроль кожен такт роботи мікроконтролера [12].

При запуску Arduino IDE (рис. 2.3) дає нам заготовку у вигляді двох обов'язкових функцій: setup() і loop(). Код в блоці setup() виконується один раз при кожному запуску мікроконтролера. Код в блоці loop() виконується «по колу» на всьому протязі роботи мікроконтролера, починаючи з моменту завершення виконання setup(). Також програма може містити підключення бібліотек.

Бібліотека є файлом (набором 29 файлів), що містить такий самий C ++ код, на якому ми пишемо скетч (іноді зустрічаються і асемблерні вставки). Ми можемо підключити бібліотеку в свій код і використовувати можливості, які вона дає, а варіантів там дуже багато: готові "інструменти" для роботи з зовнішніми датчиками і модулями, для роботи з внутрішньою периферією мікроконтролера (таймери, АЦП, пам'ять), бібліотеки різних математичних інструментів і багато чого іншого.

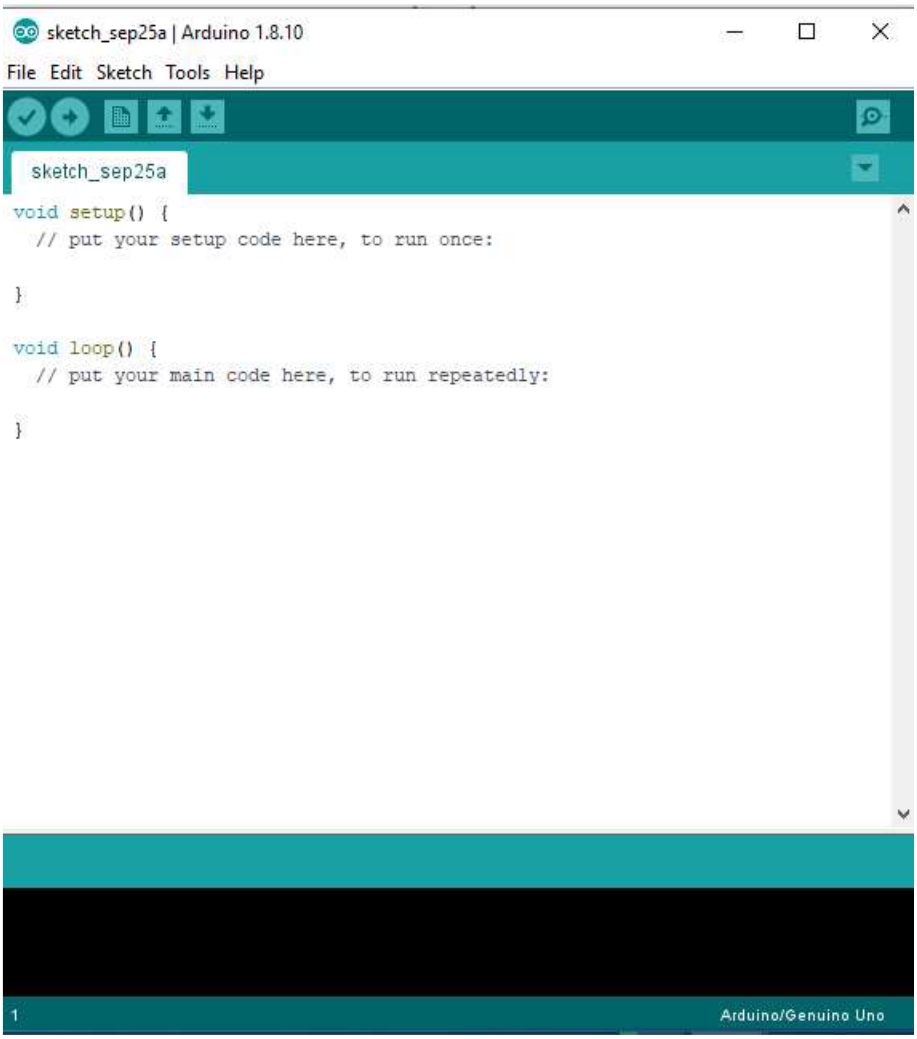


Рисунок 2.3 – Інтегроване середовище розробки Arduino (IDE).

2.3 Bluetooth модуль для Arduino HC-05

Одне з кращих рішень для організації двостороннього зв'язку по Bluetooth вашого Arduino-пристрої з планшетом, ноутбуком або іншим Bluetooth-

пристроєм – Bluetooth-модуль HC-05 (рис. 2.4), який може працювати як master (здійснювати пошук Bluetooth-пристроїв і ініціювати установку зв'язку), так і slave (пристрій – відомий) [13].

Рисунок 2.4 – Модуль Bluetooth HC-05

Основні характеристики модуля HC-05:

– Діапазон частот радіозв'язку: 2,4-2,48 ГГц

– Потужність передачі: 0,25-2,5 мВт

– Чутливість: -80 dBм

– Напруга живлення: 3,3-5 В

– Струм: 50 мА

– Радіус дії: до 10 метрів

– Інтерфейс: послідовний порт

Контакти: VCC – (живлення 3,6 – 6 В); GND - (земля); TXD (надсилання даних), RXD (отримання даних) – UART інтерфейс; STATE – індикатор стану; KEY (або EN) – контакт для входу в режим програмування.

2.4 Плата драйвера моторів L298P

Найпростіший електродвигун працює тільки на постійному струмі (від батарейки). Струм проходить по рамці, розташованої між полюсами постійного магніту. Взаємодія магнітних полів рамки з струмом і магніту змушує рамку

повертатися. Після кожного півоберту колектор перемикає контакти рамки, які підходять до батарейці, і тому рамка обертається.

До мікроконтролера мотори напряму краще не під'єднувати, адже, в основному, мотори працюють зі струмами, напруга яких більша 5В, і в момент виключення або включення мотори створять пікові перепади струму, що може негативно вплинути на плату. Тому для кращого управління моторами використовується такий модуль як драйвер мотора. Драйвер мотора – пристрій, який дозволяє легко та зручно керувати швидкістю та напрямом обертання мотора за допомогою цифрових та ШІМ сигналів [14].

L298P Motor Shield – це плата драйвера двигунів постійного струму, що використовує мікросхему потужного драйвера електродвигунів L298P, яка може безпосередньо управляти двома двигунами постійного струму; струм через навантаження – до 2 ампер. Вихідні інтерфейси управління двигунами використовують вісім високошвидкісних діодів в якості захисту. Дана плата може бути встановлена безпосередньо на плату Arduino (рис. 2.5).

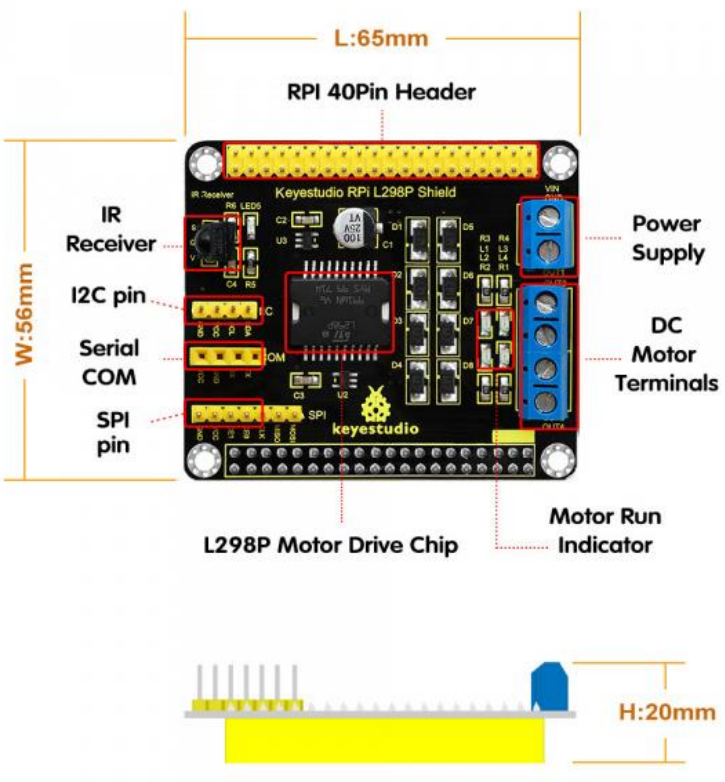


Рисунок 2.5 – L298P Motor Shield

Основні характеристики L298P Motor Shield:

– Вхідна напруга логічної частини: 5В;

– Вхідна напруга рушійної частини: 4,8 ~ 24 В;

– Робочий струм логічної частини Iss: ≤36mA;

– Робочий струм провідної частини Io: ≤ 2A;

– Максимальна розсіювана потужність: 25 Вт (T=75 ℃);

– Робоча температура: -25℃～＋130℃.

Мотор керується трьома портами – двома цифровими та одним ШІМ. Мотор може обертатися за годинниковою стрілкою або проти неї. За це відповідають цифрові порти, які на драйвері мотора підключаються до контактів IN (рис.2.6). За швидкість відповідає ШІМ порт і він підключається до EN контакту і може задавати діапазон значень 0-255.

IN1 - **високий**, IN2 - **низький** = обертання в одну сторону;
IN1 - **низький**, IN2 - **високий** = обертання в іншу сторону;
IN1 - **низький**, IN2 - **низький** = обертання немає;
IN1 - **високий**, IN2 - **високий** = обертання немає.

Рис. 2.6 – Комбінації сигналів для мотора

На рисунку 2.7 показано принципову схему плати L298P

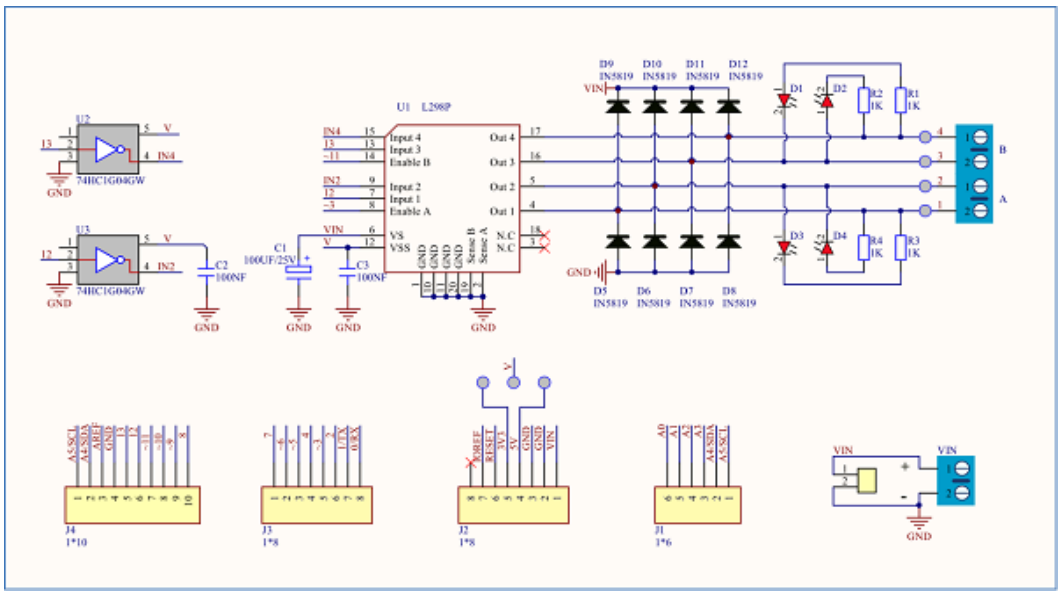


Рис. 2.7 – Принципова схема плати L298P

2.5 Плата розширення Sensor Shield V5

На платах розширення встановлюються усі необхідні електронні компоненти, а взаємодія з мікроконтролером, або іншими елементами основної плати відбувається через стандартні виводи. Зазвичай живлення на плату розширення подається з основної плати Arduino, однак в багатьох випадках існує можливість живлення з інших джерел.

На ринку готових рішень існують декілька пропозицій, що з першого погляду підходять під потреби даної роботи. Однією з таких плат розширення є плата Keyestudio Sensor Shield V5 [15]. Вона призначена для підключення до неї різних пристроїв Arduino, або аналогів таких приладів через стандартні інтерфейси. Зовнішній вигляд цієї плати розширення представлено на рисунку 2.8.

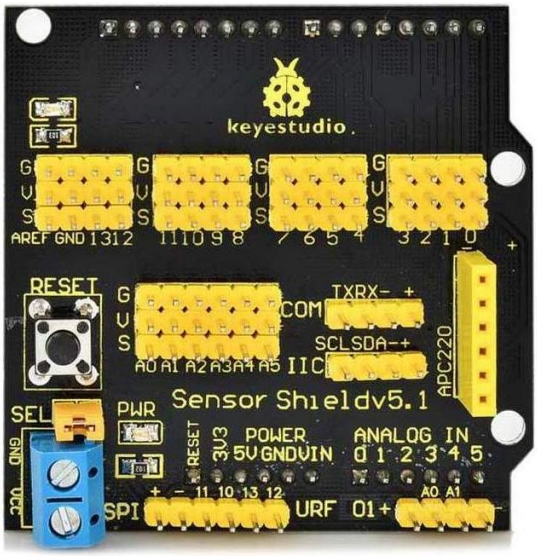


Рисунок 2.8 – Keyestudio Sensor Shield V5

Для використання даної плати розширення необхідно зібрати на її основі макет, підключаючи зовнішні пристрої до відповідних інтерфейсів плати. Далі до плати необхідно підключити Arduino контролер, або інший керуючий мікроконтролер. Та на завершення подати живлення на плату або від плати

прототипування, яка використовується, або ж від зовнішнього джерела живлення.

Керування платою розширення відбувається відповідною платформою, яка використовується. На платі знаходиться кнопка скидання параметрів, або перезавантаження RESET. Також на платі розміщені інтерфейси для підключення запам’ятовуючих пристроїв, виводи для цифрових пристроїв, аналогових, послідовна шина I2C, інтерфейс живлення для зовнішнього джерела та для живлення з плати прототипування, інтерфейси підключення модулів безпровідного зв’язку стандартів Wi-Fi та Bluetooth, а також два інтерфейсу підключення LCD дисплеїв – один послідовний та один паралельний.

На рисунку 2.9 показано можливі підключення периферії на Sensor Shield V5.

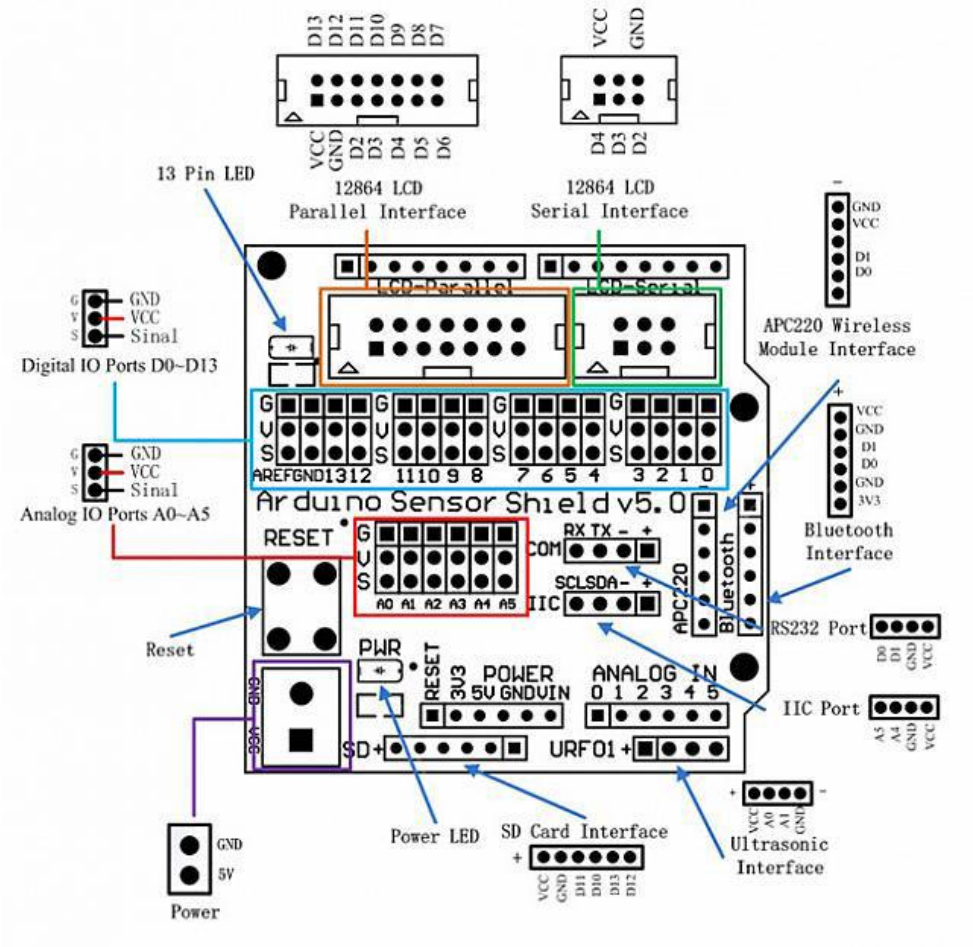


Рисунок 2.9 – Схема підключення периферії на Keyestudio Sensor Shield V5

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис	
		завдання видав	завдання прийняв
Дослідження предметної області	*Гринюк С.В.*		
Вибір апаратної платформи та середовища розробки	*Гринюк С.В.*		
Розробка програмно-апаратних засобів керування роботом	*Гринюк С.В.*		
Висновки	*Гринюк С.В.*		

7. Дата видачі завдання 01.11.2022 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів кваліфікаційної роботи	Строк виконання етапів роботи	Примітка
1.	*Обгрунтування теми кваліфікаційної роботи*	10.11.2022 р.	Виконано
2.	*Огляд літературних джерел*	22.11.2022 р.	Виконано
3.	*Дослідження предметної області*	12.01.2023 р.	Виконано
4.	*Вибір апаратної платформи та середовища розробки*	14.02.2023 р.	Виконано
5.	*Розробка програмно-апаратних засобів керування роботом*	26.03.2023 р.	Виконано
6.	*Оформлення матеріалів роботи*	03.04.2023 р.	Виконано
7.	*Нормоконтроль*	17.05.2023 р.	Виконано
8.	*Інструментальна перевірка на академічний плагіат*	01.06.2023 р	Виконано
9.	*Представлення кваліфікаційної роботи бакалавра до захисту*	15.06.2023 р.	Виконано

Здобувач вищої освіти _____ (Гриценюк В.В.)
(підпис) (прізвище, ініціали)

Керівник кваліфікаційної роботи _____ (Гринюк С.В.)
(підпис) (прізвище, ініціали)

РОЗДІЛ 3
РОЗРОБКА ПРОГРАМНО-АПАРАТНИХ ЗАСОБІВ КЕРУВАННЯ РОБОТОМ

3.1 Розробка апаратних засобів

Модель робота складається з наступних компонентів:

1. Мікроконтролера Arduino UNO R3;
2. Плати розширення Sensor Shield V5.0;
3. Модуль драйвера мотора L298P;
4. LED-панель 8x16;
5. 2-х електромоторів (5V);
6. Сервопривід;
7. Двох акумуляторів (3.7 V);
8. Bluetooth модуль HC-05.

За допомогою електромоторів буде здійснюватись рух моделі вперед-назад. Bluetooth модуль необхідний для прийому даних від додатка Драйвер мотора необхідний для управління напрямком руху моторів та швидкістю їх обертання. За допомогою мікроконтролера буде відбуватися керування всіма компонентами.

На рисунку 3.1 показана макетна схема підключення живлення та електромоторів.

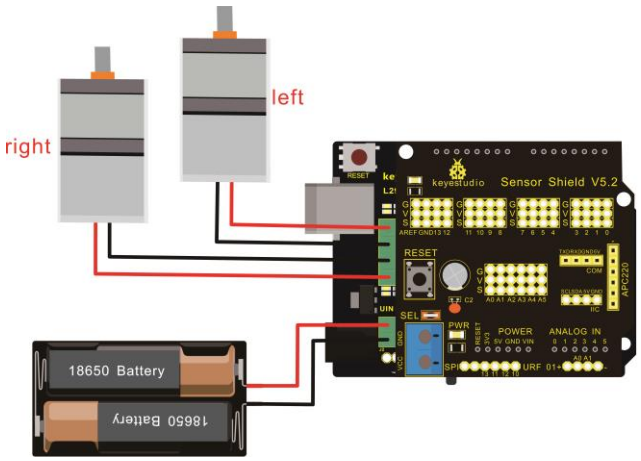


Рисунок 3.1 – Макетна схема підключення живлення та електромоторів

На рисунку 3.2 показана макетна схема підключення світлодіодної панелі.

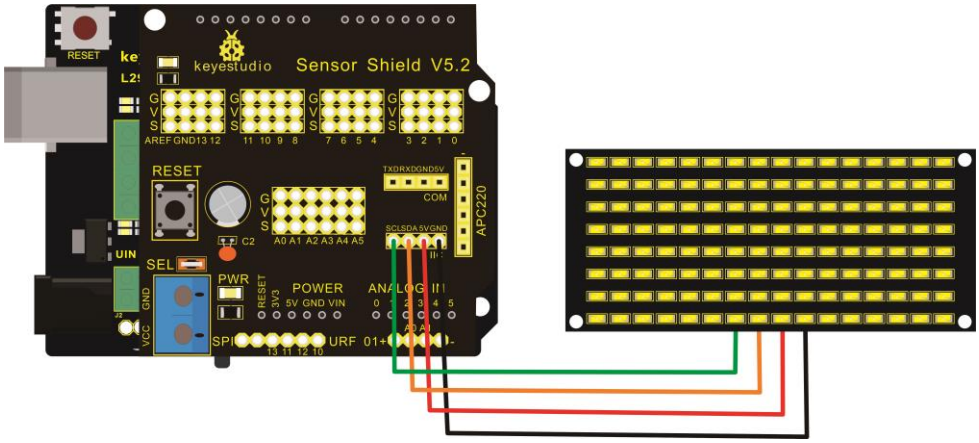


Рисунок 3.2 – Макетна схема підключення світлодіодної панелі

В табл. 3.1 представлено спосіб підключення LED панелі до плати.

Таблиця 3.1 – Підключення LED панелі до плати розширення Sensor Shield V5.0

LED панель	**Sensor Shield V5.0**
GND	-(GND)
VCC	+(VCC)
SDA	SDA
SCL	SCL

На рисунку 3.3 показана макетна схема підключення Bluetooth.

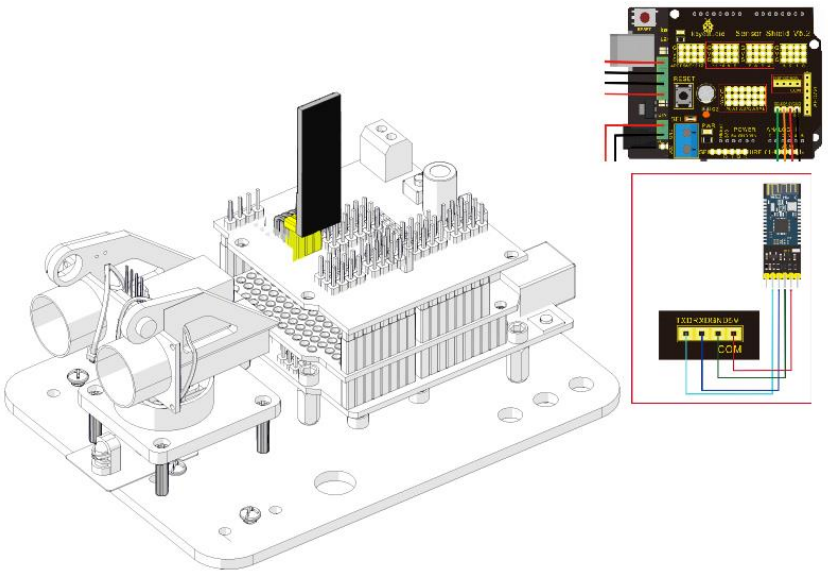


Рисунок 3.3 – Макетна схема підключення Bluetooth

В табл. 3.2 показано спосіб підключення модуля Bluetooth до плати.

Таблиця 3.2 – Підключення Bluetooth HC-05 до плати розширення Sensor Shield V5.0

Bluetooth HC-05	**Sensor Shield V5.0**
RXD	TX
TXD	RX
GND	–(GND)
VCC	+(VCC)

На рисунку 3.4 показано загальну схему підключення всіх елементів робота для дистанційного керування через Bluetooth.

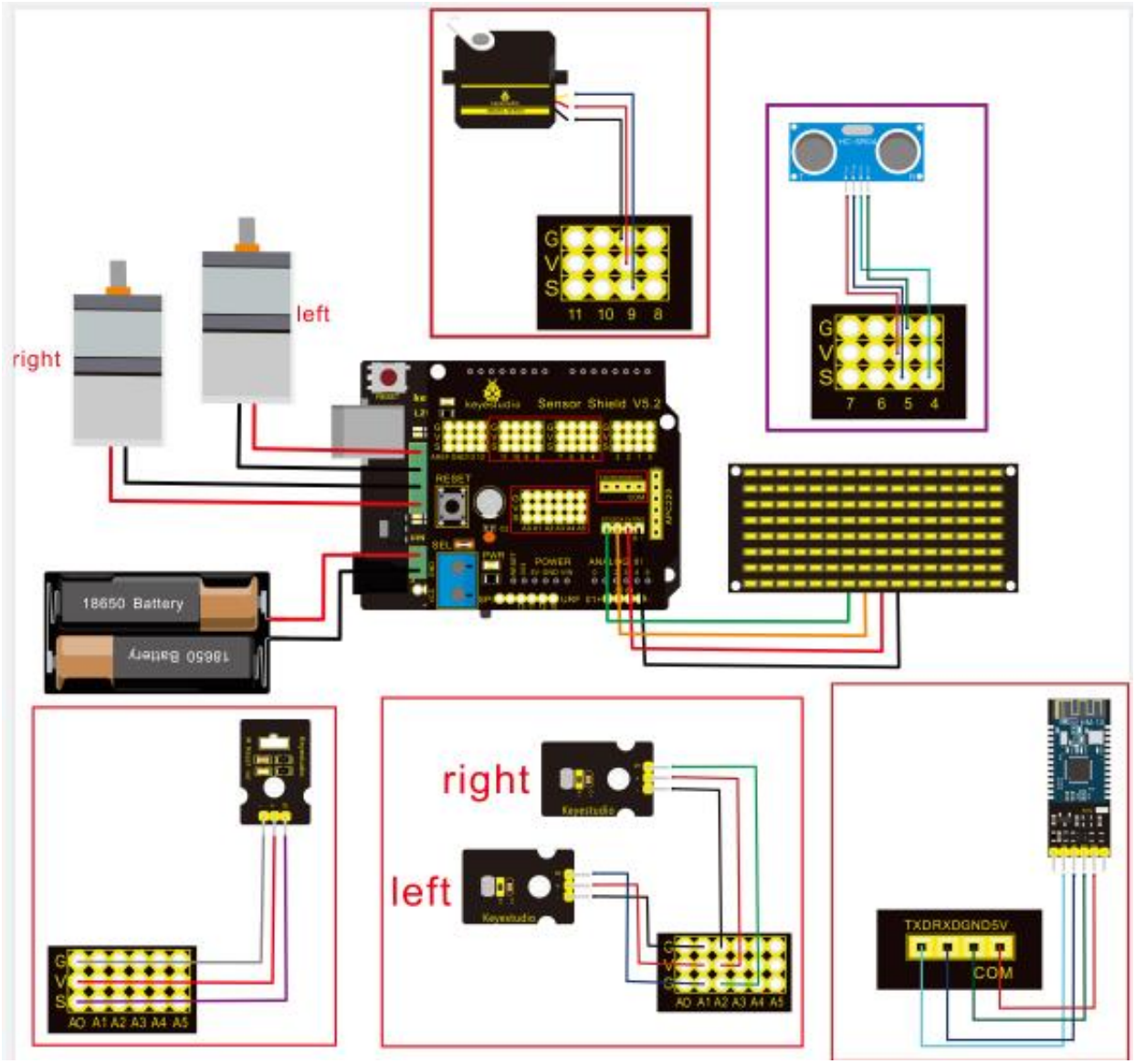


Рисунок 3.4 – Макетна схема підключення елементів робота

На рисунку 3.5 зображена блок-схема, за якою працює робот-машина на Arduino.

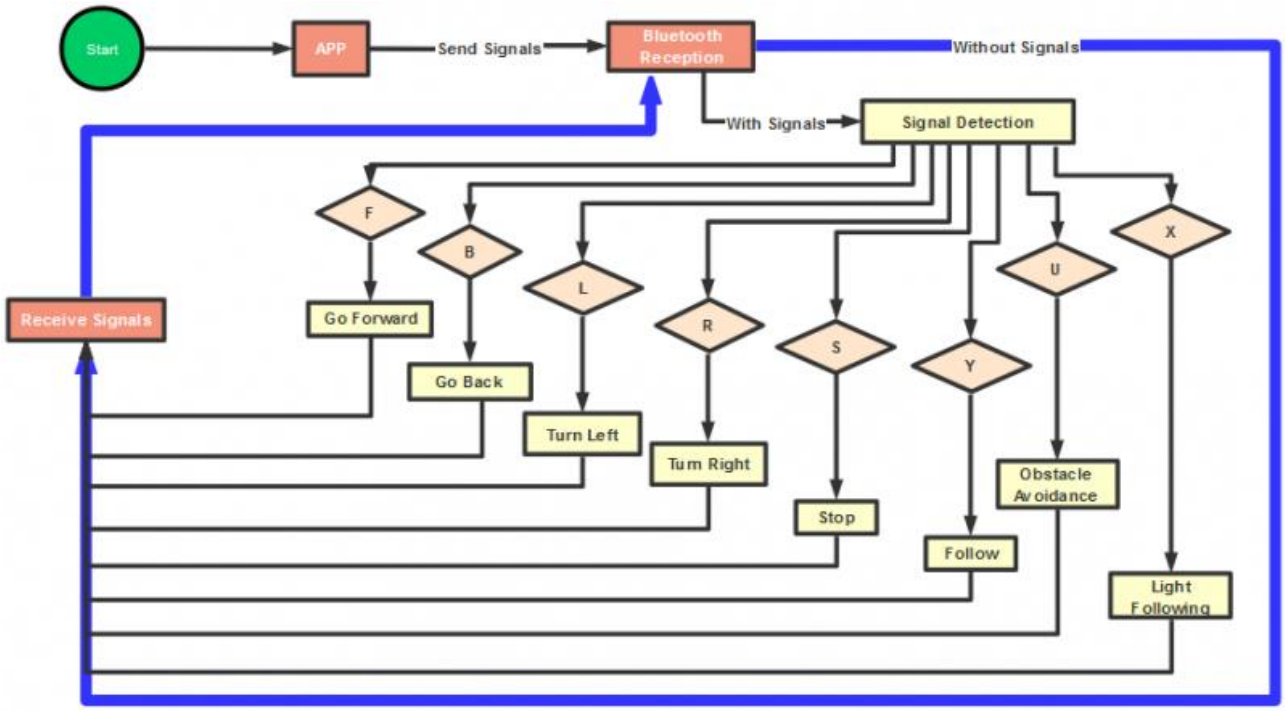


Рисунок 3.5 – Блок-схема дистанційного керування роботом

На рисунку 3.6 показано зібраний міні-робот

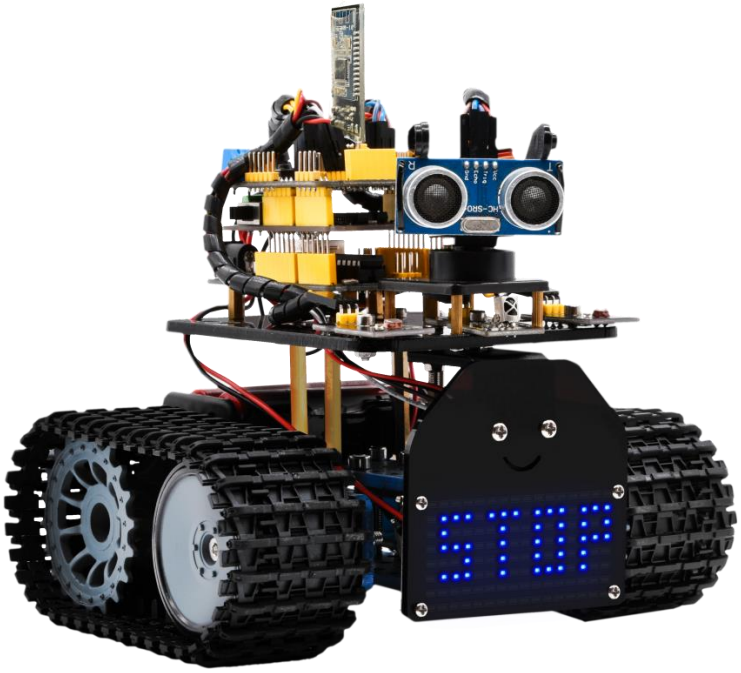


Рисунок 3.6 – Міні-танк робот

3.2 Розробка програмних засобів

Розробка програмних засобів відбувається у два етапи.

Перший етап – створення програмного інтерфейсу управління для взаємодії між телефоном та моделлю на основі мікроконтролера. Оскільки ми хочемо керувати моделлю робота, який буде здійснювати рух в двох площинах, тобто вверх-вниз та вліво-вправо, тому доцільно зробити інтерфейс, з якого буде управлятися платформа, подібним до джойстика.

Для розробки графічного інтерфейсу використали середовище розробки веб-додатків MIT App Inventor 2.

MIT App Inventor – це інтегроване середовище розробки веб-додатків Він пропонує веб-редактор «Що ви бачите – те і отримуєте» для створення програм для мобільних телефонів, орієнтованих на операційні системи Android та iOS. Він використовує блочну мову програмування, побудовану на Google Blockly і подібну на такі мовами, як StarLogo TNG та Scratch, надаючи можливість кожному створити додаток для мобільних телефонів, щоб задовольнити потреби.

Інтерфейс користувача MIT App Inventor включає два основних редактори: редактор дизайну та редактор блоків. Редактор дизайну або дизайнер (рис. 3.7) – це інтерфейс перетягування для викладення елементів користувальницького інтерфейсу програми (UI).

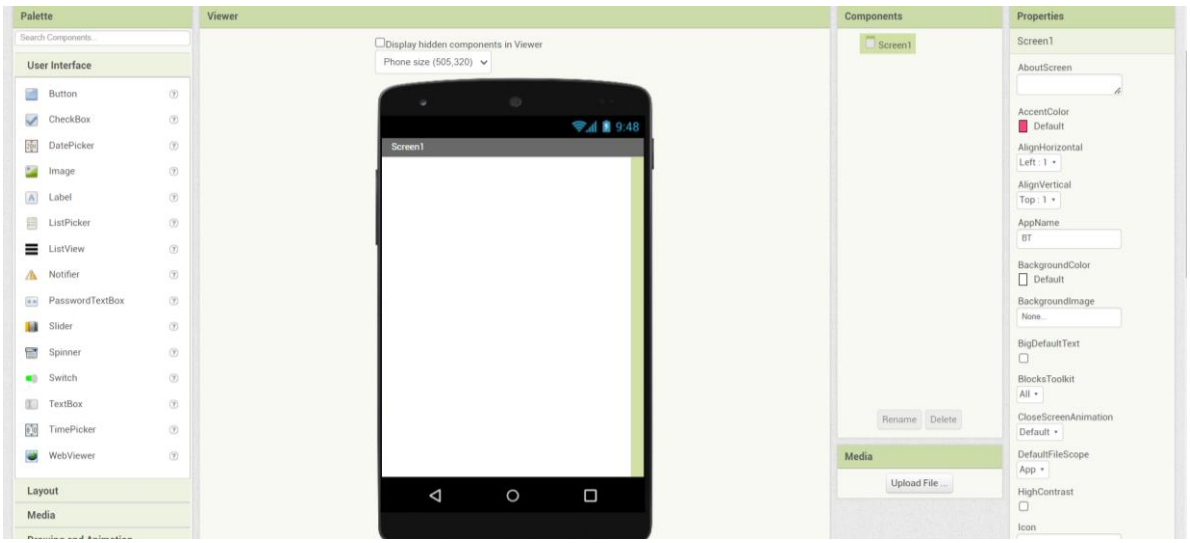


Рисунок 3.7 – Інтерфейс користувача MIT App Inventor

Розробники додатків перетягують компоненти з палітри (вкрай ліворуч) до засобу перегляду (ліворуч по центру), щоб додати їх до програми. Винахідники можуть змінювати властивості компонентів (крайній праворуч). Також відображається огляд компонентів екрану та носіїв проекту (в центрі праворуч). Редактор блоків – це середовище, в якому винахідники програм можуть візуально викласти логіку своїх додатків, використовуючи кольорові блоки, які з'єднуються, як шматочки головоломки, для опису програми. Код блоків зазвичай читається зліва направо, зверху вниз.

Таким чином, кожен може швидко створити мобільний додаток і негайно почати ітерацію та тестування.

Найперше розташовуємо кнопки для з'єднання з пристроями Bluetooth, відключення з'єднання та показу статусу з'єднання.

Другим кроком створюємо дизайн додатку та розміщуємо кнопки, за допомогою яких буде відбуватись управління рухом. Всі елементи, які ми використали, підписуємо для зручності. На цьому дизайн готовий (рис. 3.8).

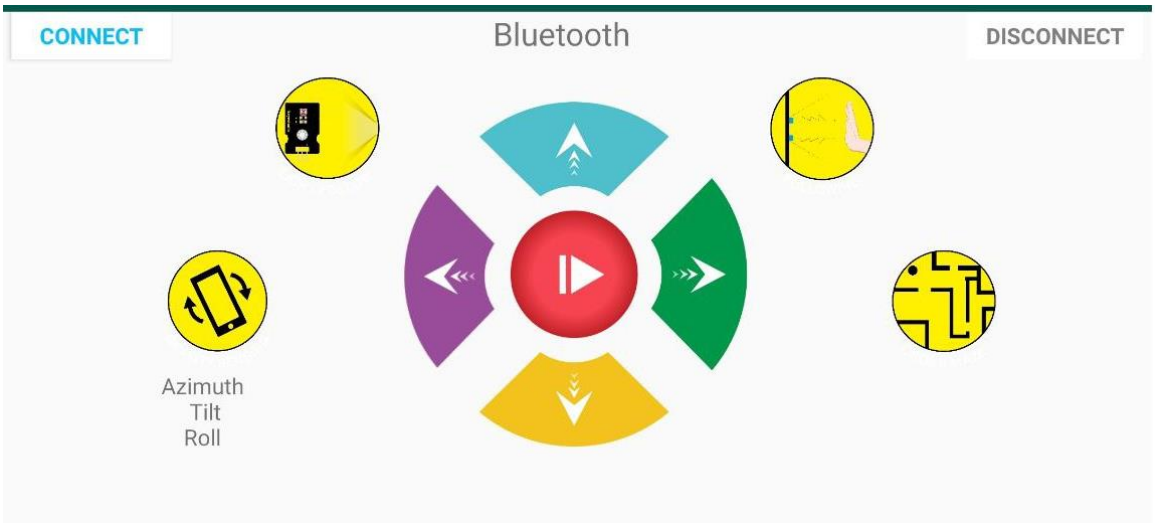


Рисунок 3.8 – Інтерфейс програми для дистанційного управління

Далі за допомогою режиму блоків ми програмуємо поведінку нашого інтерфейсу. Для початку задамо змінні в яких будуть зберігатися початкові і поточні координати кнопок. Запрограмуємо відключення при нажаті на кнопку відключення або при збої Bluetooth. Програмуємо вибір Bluetooth та зміну

надпису зі статусом. Останній крок – програмуємо поведінку кнопок, щоб він надсилав через Bluetooth координати свого положення. На цьому програмування поведінки програми закінчено (рис. 3.9).

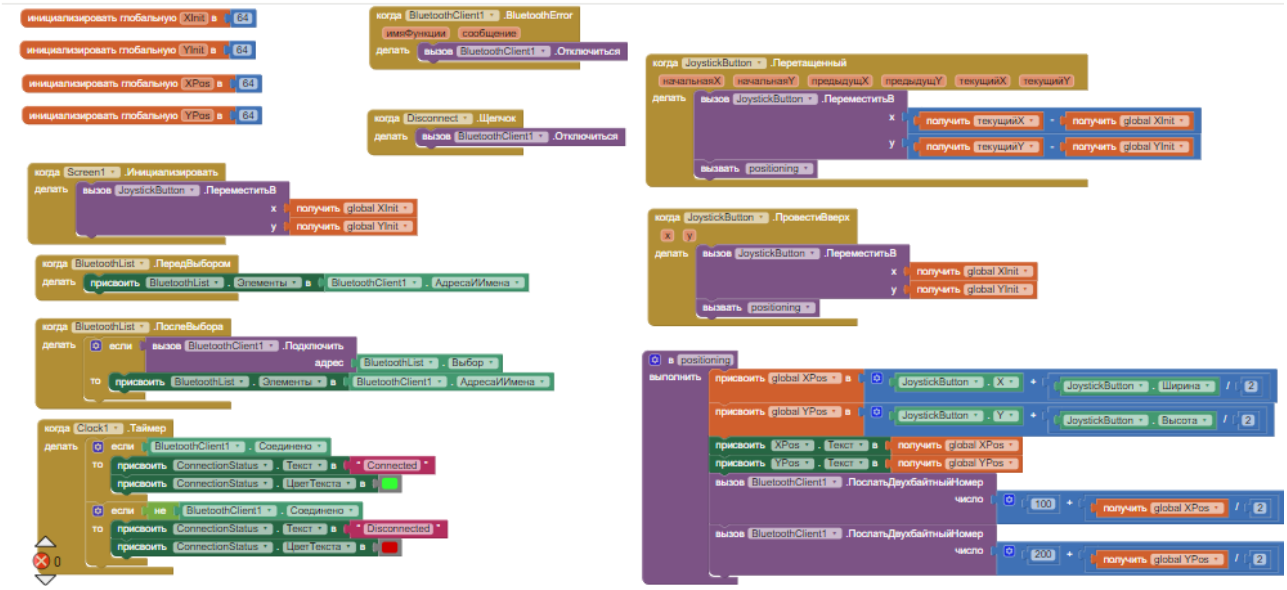


Рисунок 3.9 – Блоки програми для дистанційного управління

Другий етап – програмування мікроконтролера, який буде керувати всією схемою.

На наступному кроці за допомогою середовища програмування Arduino IDE пишемо скетч для роботи мікроконтролера. Коли код програми буде готовий, зберігаємо («Файл» – «Зберегти», або натиснути на зелений значок «Зберегти» зі стрілкою вниз в лівому верхньому куті інтерфейсу програми). Для того, щоб завантажити програму на плату, потрібно вгорі на панелі інструментів обрати пункт «Інструменти» та задати значення для двох підпунктів:

1. обрати плату, на яку потрібно завантажувати код програми, в нашому випадку це Arduino/Genuino UNO;

2. обрати послідовний порт, через який буде завантажуватись код програми (плата з мікроконтролером повинна бути підключеною до комп'ютера, через який буде завантажуватись код).

Останній кроком буде завантаження програми. Для цього в лівому верхньому куті потрібно натиснути на кнопку «Завантажити». В разі успішного

завантаження програма виведе повідомлення про успішне завершення завантаження в консольне вікно та дані про об'єм пам'яті мікроконтролера, яку завантажений скетч використовує (рис. 3.10).

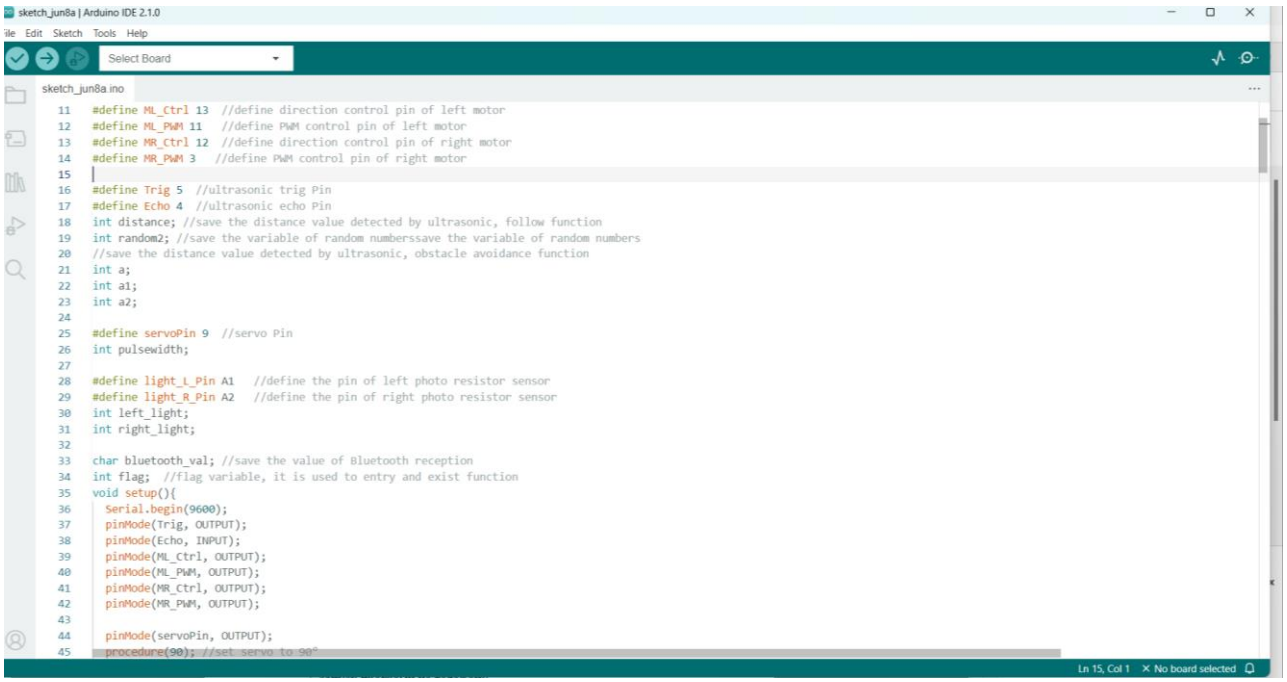


Рисунок 3.10 – Алгоритм завантаження скетчу в плату

3.3. Тестування засобів керування роботом

Тестування проводилось за наступним алгоритмом:

1.Завантаження програмного коду в мікроконтролер;

2. Ввімкнення живлення робота;

3.Завантаження програми на телефоні;

4. При переході в меню вибору Bluetooth відображався список доступних пристроїв Bluetooth;

5. Підключення до модуля Bluetooth (зазвичай він має стандартну назву HCSoft);

6. Керування роботом. Платформа приймає сигнали від телефона, та рухається правильно та без затримок. При перевірці дальності встановлено, що робот може приймати сигнали в радіусі 30 метрів.

ВИСНОВКИ

В кваліфікаційній роботі розглянуто задачі розробки системи дистанційного управління міні-роботом за допомогою мобільного додатку. Розроблена система дозволяє: розширити коло використання мобільних роботів, включаючи задачі, пов’язані із забезпеченням безпеки управлінського персоналу при виконанні особливо важливих завдань та підвищити економічність та динамічні характеристики безпосередньо міні-роботів.

У першому розділі зроблено огляд сучасного стану проблеми дистанційного управління МР, а саме, організацію систем, методи передачі даних, структурні елементи МР. Для розв’язання задачі дистанційного управління МР розроблено структуру апаратної та програмної частини системи.

За результатами огляду сучасних рішень з розробки систем дистанційного управління обрано технологію IoT. В якості протоколу передачі даних було обрано технологію Bluetooth. За типом організації системи зв’язку було обрано метод point-to-point.

Розроблено схеми підключення електронних компонентів системи керування міні-робота, здійснено розрахунок блоку живлення. Розроблено схеми алгоритмів роботи програми міні-робота. Розроблено програмний засіб для дистанційного управління, а саме, мобільний додаток, який відправляє команди мережею Bluetooth на приймач міні-робота, та програму керування для контролера, яка приймає дані від Bluetooth модулю HC-06, та обробляє їх.

Практичне використання результатів роботи дозволить застосовувати систему дистанційного управління під час наукової діяльності, проводити дослідження у важкодоступних або небезпечних місцях, наприклад під час вибухотехнічних операцій.

Напрямками подальшого удосконалення розробки можуть бути: додавання заздалегідь запрограмованих інтерфейсів керування для існуючих роботів, або робототехнічних систем; додавання можливості керування пристроями за допомогою мережевого зв’язку через мережу Internet; розробка більш гнучкого

інтерфейсу, для надання можливості користувачу підлаштовувати його під свої цілі; розробка більш досконалої системи команд; додавання програмного протоколу передачі даних, для унеможливлення втрати пакетів та більшої безпеки під час передачі даних.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Медведєв А. М. Аналіз стану систем управління роботизованими системами. Автоматизація та приладобудування («Automation and Development of Electronic Devices» ADED-2020): збірник студентських наукових статей / Харківський національний університет радіоелектроніки [редкол.: І.Ш. Невлюдов та ін.]. Харків : ХНУРЕ, 2020. Вип. 2. С. 262-265.

2. Сезонова І. К. Автоматизація технології управління виробничим процесом за допомогою мобільних додатків [Текст] / Виробництво&Мехатронні Системи 2019: Програма III-ої Міжнародної конференції, Харків, 24-25 жовтня 2019 р.: тези доповідей. Харків, 2019. С. 19-21.

3. Поліщук М. М., Ткач М. М. Робототехнічні системи: проєктування і моделювання: навч. Посіб. для студ. спеціальності 126 «Інформаційні системи та технології», Київ: КПІ ім. Ігоря Сікорського, 2021. 112 с.

4. Micro Smart Bit Robot Car V2 URL: https://arduino.ua/prod3279-microbit-smart-robo-platforma-ot-keyestudio-ks0426 (дата звернення: 14.10.2022).

5. Що таке інтернет речей і навіщо він потрібен? URL: https://techno.nv.ua/popscience/chto-takoe-internet-veshchej-1326653.html (дата звернення: 05.11.2022).

6. Що таке Bluetooth? URL: http://technoportal.ua/goodies/glossary/23.html (дата звернення: 20.11.2022).

7. Wi-Fi URL: http://www.uk.wikipedia.org/wiki/Wi-Fi (дата звернення: 20.11.2022).

8. GSM URL: http://www.uk.wikipedia.org/wiki/GSM (дата звернення: 20.11.2022).

9. Arduino UNO URL: https://store.arduino.cc/arduino-uno-rev3 (дата звернення: 15.01.2023).

10. ARM URL: https://uk.wikipedia.org/wiki/ARM (дата звернення: 15.01.2023).

11. Arduino IDE URL: https://www.arduino.cc/en/software (дата звернення: 24.10.2020).

12. Microcontroller (MCU) URL: https://internetofthingsagenda.techtarget.com (дата звернення 20.02.2023).

13. HC-05-BLUETOOTH URL:http://www.kosmodrom.com.ua/el.php?name=HC-05-BLUETOOTH (дата звернення: 25.02.2023).

14. Драйвер двигунів L298N URL: https://3d-diy.ru/wiki/arduino-moduli/drayver-dvigatelya-l298n/#Techno (дата звернення: 25.02.2023).

15. Sensor Shield v5 0 URL: https://forum.arduino.cc/t/arduino-sensor-shield-v5-apc220-manual/223457 (дата звернення: 01.03.2023).

ДОДАТКИ

Додаток А

Лістинг коду файлу Bluetooth.ino

```
//Array, used to store the data of the pattern, can be calculated by yourself or obtained from the modulus tool
unsigned char start01[] = {0x01,0x02,0x04,0x08,0x10,0x20,0x40,0x80,0x80,0x40,0x20,0x10,0x08,0x04,0x02,0x01};
unsigned char front[] = {0x00,0x00,0x00,0x00,0x00,0x24,0x12,0x09,0x12,0x24,0x00,0x00,0x00,0x00,0x00,0x00};
unsigned char back[] = {0x00,0x00,0x00,0x00,0x00,0x24,0x48,0x90,0x48,0x24,0x00,0x00,0x00,0x00,0x00,0x00};
unsigned char left[] = {0x00,0x00,0x00,0x00,0x00,0x00,0x44,0x28,0x10,0x44,0x28,0x10,0x44,0x28,0x10,0x00};
unsigned char right[] = {0x00,0x10,0x28,0x44,0x10,0x28,0x44,0x10,0x28,0x44,0x00,0x00,0x00,0x00,0x00,0x00};
unsigned char STOP01[] =
{0x2E,0x2A,0x3A,0x00,0x02,0x3E,0x02,0x00,0x3E,0x22,0x3E,0x00,0x3E,0x0A,0x0E,0x00};
unsigned char clear[] = {0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00};
#define SCL_Pin  A5  //Set clock pin to A5
#define SDA_Pin  A4  //Set data pin to A4

#define ML_Ctrl 13  //define direction control pin of left motor
#define ML_PWM 11   //define PWM control pin of left motor
#define MR_Ctrl 12  //define direction control pin of right motor
#define MR_PWM 3   //define PWM control pin of right motor

#define Trig 5  //ultrasonic trig Pin
#define Echo 4  //ultrasonic echo Pin
int distance; //save the distance value detected by ultrasonic, follow function
int random2; //save the variable of random numberssave the variable of random numbers
//save the distance value detected by ultrasonic, obstacle avoidance function
int a;
int a1;
int a2;

#define servoPin 9  //servo Pin
int pulsewidth;

#define light_L_Pin A1   //define the pin of left photo resistor sensor
#define light_R_Pin A2   //define the pin of right photo resistor sensor
int left_light;
int right_light;

char bluetooth_val; //save the value of Bluetooth reception
int flag;  //flag variable, it is used to entry and exist function
void setup(){
  Serial.begin(9600);
  pinMode(Trig, OUTPUT);
  pinMode(Echo, INPUT);
  pinMode(ML_Ctrl, OUTPUT);
  pinMode(ML_PWM, OUTPUT);
  pinMode(MR_Ctrl, OUTPUT);
  pinMode(MR_PWM, OUTPUT);

  pinMode(servoPin, OUTPUT);
  procedure(90); //set servo to 90°
 pinMode(SCL_Pin,OUTPUT);
  pinMode(SDA_Pin,OUTPUT);
  matrix_display(clear);    //Clear the display
  matrix_display(start01);  //display start pattern
  pinMode(light_L_Pin, INPUT);
  pinMode(light_R_Pin, INPUT);
}

void loop(){
  if (Serial.available())
  {
```

```
    bluetooth_val = Serial.read();
    Serial.println(bluetooth_val);
  }
  switch (bluetooth_val)
  {
    case 'F':  //Forward instruction
      Car_front();
      matrix_display(front);  //display forward pattern
      break;
    case 'B':  //Back instruction
      Car_back();
      matrix_display(back);  // display back pattern
      break;
    case 'L':  //left-turning instruction
      Car_left();
      matrix_display(left);  //show left-turning pattern
      break;
    case 'R':  //right-turning instruction
      Car_right();
      matrix_display(right);  //show right-turning pattern
      break;
    case 'S':  //stop instruction
      Car_Stop();
      matrix_display(STOP01);  //display stop pattern
      break;
   case 'Y':
      matrix_display(start01);  //show start pattern
      follow();
      break;
   case 'U':
      matrix_display(start01);  //show start pattern
      avoid();
      break;
   case 'X':
      matrix_display(start01);  //show start pattern
      light_track();
      break;
  }}
/*******************Obstacle Avoidance Function***************/
void avoid()
{
  flag = 0;  //the design that enter obstacle avoidance function
  while (flag == 0)
  {
    random2 = random(1, 100);
    a = checkdistance();  //assign the front distance detected by ultrasonic sensor to variable a

    if (a < 20) //when the front distance detected is less than 20cm
    {
      Car_Stop();  //robot stops
      delay(200); //delay in 200ms

      procedure(160);  //Ultrasonic platform turns left
      for (int j = 1; j <= 10; j = j + (1)) { ///for statement, the data will be more accurate if ultrasonic sensor detect a few
times.
        a1 = checkdistance();  //assign the left distance detected by ultrasonic sensor to variable a1
      }
      delay(200);
      procedure(20); //Ultrasonic platform turns right
      for (int k = 1; k <= 10; k = k + (1)) {
        a2 = checkdistance(); //assign the right distance detected by ultrasonic sensor to variable a2
      }
```

```
    if (a1 < 50 || a2 < 50)  //robot will turn to the longer distance side when left or right distance is less than 50cm.if the
left or right distance is less than 50cm, the robot will turn to the greater distance
    {
     if (a1 > a2) //left distance is greater than right
     {
      procedure(90);  //Ultrasonic platform turns back to right ahead ultrasonic platform turns front
      Car_left();  //robot turns left
      delay(500);  //turn left 500ms
      Car_front(); //go forward
     }
     else
     {
      procedure(90);
      Car_right(); //robot turns right
      delay(500);
      Car_front();  //go forward
     }
    }
    else  //both distance on two side is greater than or equal to 50cm, turn randomly
    {
     if ((long) (random2) % (long) (2) == 0)  //when the random number is even
     {
      procedure(90);
      Car_left(); //robot turns left
      delay(500);
      Car_front(); //go forward
     }
     else
     {
      procedure(90);
      Car_right(); //robot turns right
      delay(500);
      Car_front(); ///go forward
    } } }
  else  //If the front distance is greater than or equal to 20cm, robot car will go front
  {
     Car_front(); //go forward
  }
  // receive the Bluetooth value to end the obstacle avoidance function
  if (Serial.available())
  {
   bluetooth_val = Serial.read();
   if (bluetooth_val == 'S')  //receive S
   {
    flag = 1;  //when assign 1 to flag, end loop
   }}}}
/********************Follow****************/
void follow() {
  flag = 0;
  while (flag == 0) {
   distance = checkdistance();  //assign the distance detected by ultrasonic sensor to distance
   if (distance >= 20 && distance <= 60) //the range to go front
   {
    Car_front();
   }
   else if (distance > 10 && distance < 20)  //the range to stop
   {
    Car_Stop();
   }
   else if (distance <= 10)  // the range to go back
   {
    Car_back();
   }
```

```
    else  //other situations, stop
    {
      Car_Stop();
    }
    if (Serial.available())
    {
      bluetooth_val = Serial.read();
      if (bluetooth_val == 'S')
      {
        flag = 1;  //end loop
      }}}}
//The function to control ultrasonic sensor the function controlling ultrasonic sensor
float checkdistance() {
  digitalWrite(Trig, LOW);
  delayMicroseconds(2);
  digitalWrite(Trig, HIGH);
  delayMicroseconds(10);
  digitalWrite(Trig, LOW);
  float distance = pulseIn(Echo, HIGH) / 58.00;  //58.20 means 2*29.1=58.2
  delay(10);
  return distance;
}
//The function to control servo the function controlling servo
void procedure(int myangle) {
  for (int i = 0; i <= 50; i = i + (1)) {
    pulsewidth = myangle * 11 + 500;
    digitalWrite(servoPin,HIGH);
    delayMicroseconds(pulsewidth);
    digitalWrite(servoPin,LOW);
    delay((20 - pulsewidth / 1000));
  }}

/*****************Light Follow******************/
void light_track() {
  flag = 0;
  while (flag == 0) {
    left_light = analogRead(light_L_Pin);
    right_light = analogRead(light_R_Pin);
    if (left_light > 650 && right_light > 650) //the value detected by photo resistor, go forward
    {
      Car_front();
    }
    else if (left_light > 650 && right_light <= 650)  //the value detected by photo resistor, turn left
    {
      Car_left();
    }
    else if (left_light <= 650 && right_light > 650) //the value detected by photo resistor, turn right
    {
      Car_right();
    }
    else  //other situations, stop
    {
      Car_Stop();
    }
    if (Serial.available())
    {
      bluetooth_val = Serial.read();
      if (bluetooth_val == 'S') {
        flag = 1;
      }}}}
/*****************Dot Matrix ******************/
// this function is used for dot matrix display
void matrix_display(unsigned char matrix_value[])
```

```
{
  IIC_start();
  IIC_send(0xc0);  //Choose address

  for(int i = 0;i < 16;i++) //pattern data has 16 bits
  {
    IIC_send(matrix_value[i]); //convey the pattern data
  }
  IIC_end();   //end the transmission of pattern data
  IIC_start();
  IIC_send(0x8A);  //display control, set pulse width to 4/16
  IIC_end();
}
//The condition starting to transmit data
void IIC_start()
{
  digitalWrite(SCL_Pin,HIGH);
  delayMicroseconds(3);
  digitalWrite(SDA_Pin,HIGH);
  delayMicroseconds(3);
  digitalWrite(SDA_Pin,LOW);
  delayMicroseconds(3);
}
//convey data
void IIC_send(unsigned char send_data)
{
  for(char i = 0;i < 8;i++)  //each byte has 8 bits
  {
      digitalWrite(SCL_Pin,LOW);  //pull down clock pin SCL Pin to change the signals of SDA
      delayMicroseconds(3);
      if(send_data & 0x01)  //set high and low level of SDA_Pin according to 1 or 0 of every bit
      {
        digitalWrite(SDA_Pin,HIGH);
      }
      else
      {
        digitalWrite(SDA_Pin,LOW);
      }
      delayMicroseconds(3);
      digitalWrite(SCL_Pin,HIGH); //pull up clock pin SCL_Pin to stop transmitting data
      delayMicroseconds(3);
      send_data = send_data >> 1;  // detect bit by bit, so move the data right by one
  }}
//The sign that data transmission ends
void IIC_end()
{
  digitalWrite(SCL_Pin,LOW);
  delayMicroseconds(3);
  digitalWrite(SDA_Pin,LOW);
  delayMicroseconds(3);
  digitalWrite(SCL_Pin,HIGH);
  delayMicroseconds(3);
  digitalWrite(SDA_Pin,HIGH);
  delayMicroseconds(3);
}
/**********the function to run motor**********/
void Car_front()
{
  digitalWrite(MR_Ctrl,LOW);
  analogWrite(MR_PWM,200);
  digitalWrite(ML_Ctrl,LOW);
  analogWrite(ML_PWM,200);
}
```

```
void Car_back()
{
  digitalWrite(MR_Ctrl,HIGH);
  analogWrite(MR_PWM,200);
  digitalWrite(ML_Ctrl,HIGH);
  analogWrite(ML_PWM,200);
}
void Car_left()
{
  digitalWrite(MR_Ctrl,LOW);
  analogWrite(MR_PWM,255);
  digitalWrite(ML_Ctrl,HIGH);
  analogWrite(ML_PWM,255);
}
void Car_right()
{
  digitalWrite(MR_Ctrl,HIGH);
  analogWrite(MR_PWM,255);
  digitalWrite(ML_Ctrl,LOW);
  analogWrite(ML_PWM,255);
}
void Car_Stop()
{
  digitalWrite(MR_Ctrl,LOW);
  analogWrite(MR_PWM,0);
  digitalWrite(ML_Ctrl,LOW);
  analogWrite(ML_PWM,0);
}
void Car_T_left()
{
  digitalWrite(MR_Ctrl,LOW);
  analogWrite(MR_PWM,255);
  digitalWrite(ML_Ctrl,LOW);
  analogWrite(ML_PWM,180);
}
void Car_T_right()
{
  digitalWrite(MR_Ctrl,LOW);
  analogWrite(MR_PWM,180);
  digitalWrite(ML_Ctrl,LOW);
  analogWrite(ML_PWM,255);
}
```