КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

ІМЕНІ ТАРАСА ШЕВЧЕНКА

Факультету радіофізики, електроніки та комп’ютерних систем

Кафедра комп’ютерної інженерії

Розробка макета рухомої платформи з можливістю дистанційного керування

Дипломна робота бакалавра

студента 4 року навчання

Владислава ГОЛУБЦОВА

Спеціальність: 123 «Комп’ютерна інженерія»

Науковий керівник

канд. фіз.-мат. наук Олександр БАУЖА,

доцент кафедри кафедра комп'ютерної інженерії

Рецензент

канд. фіз.-мат. наук Богдан СУСЬ,

доцент кафедри нанофізики конденсованих середовищ

Навчально наукового інституту високих технологій

До захисту допускаю: Завідувач кафедрою Юрій БОЙКО

Ухвалено на засіданні кафедри “____”_____2022 р., протокол № ____

Київ 2022

РЕФЕРАТ

Дипломна робота: 66 с., 28 рис., 6 табл., 11 джерел, 1 додаток.

Метою даної роботи є розробка демонстраційного макета керованої рухомої платформи з розширеним функціоналом.

У даній бакалаврській роботі розроблено демонстраційний макет рухомої платформи з можливістю використання віддаленого керування. Макет розроблений на базі апаратно-обчислювальної платформи Arduino з використанням допоміжних модулів та драйверів для реалізації відповідної логіки роботи. Керування рухомою платформою здійснюється за допомогою створеного мобільного застосунку, який взаємодіє з ним за технологією бездротового зв'язку Bluetooth.

У першому розділі розглянуто технічну документацію компонентів для розробки рухомої платформи з дистанційним керуванням. Також виокремлено фізичну складову мікроконтролера та відповідних модулів для подальшої модернізації макета.

У другому розділі розглядається практична частина дипломної роботи, у якій проведено аналіз вибору інтегрованого середовища розробки, а також логіка роботи мікроконтролера та мобільного застосунку. Описано схематичне конструювання макета розробленої платформи та продемонстровано результати роботи.

Розроблений макет може використовуватись для демонстрації у курсах «Периферійні пристрої», «Мікропроцесорна техніка», «Схемотехніка», «Програмування вбудованих систем».

ПЕРЕДАЧА ДАНИХ, ДИСТАНЦІЙНЕ КЕРУВАННЯ, ШИРОТНО-ІМПУЛЬСНА МОДУЛЯЦІЯ, ДАТЧИК, ДВИГУН, СВІТЛОДІОД, ЗАСТОСУНОК, МІКРОКОНТРОЛЕР, АПАРАТНО-ОБЧИСЛЮВАЛЬНА ПЛАТФОРМА ARDUINO

Зміст

Перелік умовних скорочень

МП — мікропроцесор

МК — мікроконтролер

ПЗП — постійний запам'ятовувальний пристрій

АЛП — арифметико-логічний пристрій

РЗП — регістри загального призначення

ЦП — центральний процесор

ОЗП — оперативно-запам'ятовувальний пристрій

АЦП — аналого-цифровий перетворювач

ШІМ — широтно-імпульсна модуляція

ЕОМ — електронно-обчислювальна машина

RISC (Reduced Instruction Set Computer) — архітектура процесора, збільшення швидкодії здійснюється внаслідок спрощення інструкцій

CISC (Complex Instruction Set Computer) — архітектура процесора, більшість команд якої є комплексними

AVR (**A**lf and **V**egard's **R**ISC) — сімейство мікроконтролерів, що раніше випускалися фірмою Atmel, нині Microchip

PIC (Peripheral Interface Controller) — сімейство мікроконтролерів Microchip

DSP (Digital Signal Processor) — спеціалізований програмований мікропроцесор, призначений для маніпулювання в реальному часі потоком цифрових даних

SRAM (Static Random Access Memory) — статична оперативна пам'ять

ROM (Read-Only Memory) — енергонезалежна пам'ять, із якої може проводитись тільки зчитування даних

IDE (Integrated Development Environment) — інтегроване середовище розробки

Вступ

Сучасні технології бездротової передачі даних із кожним днем стають все кращими, збільшується діапазон покриття та швидкість передачі даних тощо. Важко уявити життя без електронних пристроїв сьогодні. Для покращення комфорту, багато приладів побуту намагаються об’єднати в одну мережу та керувати нею дистанційно. Прикладом можуть бути розумні будинки.

Для розробки будь-якого пристрою знадобиться платформа розробки. Платформа Arduino є зручною платформою для розробки багатофункціональних пристроїв. Вона має велику кількість плат із функціоналом, що забезпечить виконання будь-якої цілі. Під платформу Arduino випускається понад 500 модулів, датчиків та адаптерів, що своєю чергою відкриває широкий діапазон шляхів для розробки різноманітних приладів та пристроїв.

Розвиток інтегральних мікросхем включає велику кількість електронних елементів, і застосування їх в ЕОМ. Починаючи з третього покоління й до сьогодення, дозволяє збільшувати швидкодію та спростити процедуру комунікації людини з ЕОМ. А також надзвичайно наближує її до об'єкта управління та контролю. Наступні зростання ступеня інтеграції дозволяють розміщувати на кристалі мікросхем вже не вузли чи фрагменти пристроїв ЕОМ, а цілі ЕОМ.

У результаті цього розвитку були створені мікроконтролери, вироби мікроелектроніки та обчислювальної техніки принципово нового рівня, здатні обробляти та зберігати інформацію в одній мікросхемі. Якщо використовувати мікроконтролери у розробках, це не тільки підвищить технічні й економічні показники надійності, розмірів, вартості, споживчої потужності, а також дозволить зменшити час на розробки виробів та зробить їх придатними для подальшої модифікації.

Сучасна мікроелектроніка спонукає до розвитку та об'єднання мікроконтролерів із новітніми пристроями, як-от смартфони, планшети, ноутбуки чи будь-які інші сучасні комунікаційні пристрої. Розробка електричних машин із додатковими модифікаціями підвищує ефективність та спрощує використання за допомогою розробленого додатка.

Налагоджування зв'язку між пристроєм та мікроконтролером можливе бездротовим видом з’єднання Bluetooth, а саме за допомогою Bluetooth модулів. Характерними особливостями цього виду є: доступність та простота у використанні. Також він не вимагає прямої видимості між пристроями для синхронізації. Переваги цієї технології перекривають будь-які негативні аспекти, тому незабаром люди використовуватимуть цей різновид з’єднання в усіх електронних пристроях свого будинку.

1 Мікроконтролери

Мікроконтролер (англ. microcontroller) — це компактна інтегральна схема, призначена для керування певною операцією у вбудованій системі. Вбудована система є комбінацією апаратного та програмного забезпечення. Мікроконтролер містить у собі мікропроцесор, постійну та оперативну пам’ять для зберігання коду, що виконується та даних, периферійні пристрої, блоки зі спеціальними функціями, що розміщені на одній інтегральній схемі. Працює як невеликий автономний комп’ютер малих розмірів. Компактність конструкції дозволяє зменшити енергоспоживання, і у результаті приведе до економії коштів. Також мікроконтролери можуть забезпечити функціональну безпеку і безпеку вбудованих систем. Для уявлення про те, як мікроконтролер взаємодіє з навколишнім світом розглянемо загальну структуру мікроконтролера.

Рисунок 1. Структура мікроконтролера

Мікроконтролер — це монокристальний комп'ютер, що здатний виконувати елементарні задачі. Їх використовують для керування електронними пристроями. Мікроконтролери зустрічаються у таких сферах: автомобілебудування, електроніка, побутова техніка, офісне приладдя тощо. Сучасні автомобілі оснащені великою кількістю мікроконтролерів, а саме: круїз-контролем, антиблокувальною системою, керуванням двигуна, системою кондиціювання та підігрівом сидінь. Електроніка

демонструє мікроконтролери у вигляді телевізорів, автовідповідачів, музикального приладдя та інших розумних пристроїв. У офісних приміщеннях МК зустрічаються в різних моніторах, лазерних принтерах та пристроях зчитування. Можемо побачити, що життя полегшується саме завдяки цим розробкам.

Умовно мікроконтролери поділені на такі класи: 8-розрядні МК для вбудованих програм, 16- і 32-розрядні МК та цифрові сигнальні процесори.

Основним представником сімейства мікроконтролерів є 8-розрядні прилади, що широко використовують у промисловості, комп'ютерній або побутовій техніці. Розвиток прямував від елементарних мікроконтролерів зі слабкою периферією до найсучасніших мультифункціональних МК, що забезпечують реалізацію роботи складних та довготривалих алгоритмів керування.

Особливістю 16- і 32-розрядних МК є використання тільки зовнішньої пам’яті, яка містить пам’ять програм та деякий обсяг пам’яті даних. Також n-бітовий мікроконтролер одночасно працює з n бітами.

Цифрові сигнальні процесори вважаються процесорами нової категорії. Мета DSP зазвичай полягає у вимірюванні, фільтрації чи стисканні безперервних аналогових сигналів. Більша частина МП загального призначення мають можливість виконувати алгоритми цифрової обробки сигналів, але з можливими проблемами. Безпосередньо DSP використовують на бортах літаків у мікрофонах з активним контролем шуму та для придушення роздвоєння зображення у телевізійних сигналах.

Також мікроконтролер може містити вбудовану незалежну пам'ять. Більша частина контролерів взагалі не має шин підключення зовнішньої пам'яті. Через відносно невелику вартість типи пам'яті використовують лише для одноразового запису. Мікроконтролери звичного вигляду мають багаторазовий перезапис зовнішньої пам'яті.

1.1 Принцип роботи мікроконтролера

Попри складний пристрій, принцип роботи мікроконтролера надзвичайно простий. МК відповідає аналоговому принципу дії. Електронна система розуміє лише дві команди: існує сигнал чи ні. Відповідно до цих сигналів у пам'ять вписується код певної команди. У кожному МК прописані свої базові набори команд і лише їх він здатний приймати та виконувати. Час від часу МК зчитує команди й виконує їх. Комбінуючи окремі команди між собою, можна написати унікальну програму, за якою працюватиме будь-який електронний пристрій.

Залежно від набору програм МК поділяють на архітектури:

- CISC — може виконувати одну складну інструкцію, яка виконує ті самі операції, всі відразу, безпосередньо в пам'яті.
- RISC — завантажує кожну інструкцію, виконує дії з нею та завантажує наступну.

RISC архітектура	CISC архітектура
1-байтові команди	Багатобайтові команди
Багато регістрів	Мало регістрів
Команди простого виконання	Команди складного виконання
Декілька команд за 1 цикл процесора	1 команда за 1 цикл процесора
Декілька виконавчих пристроїв	1 виконавчий пристрій

Таблиця 1. Основні риси архітектур процесорів

Більшість контролерів виконані в RISC архітектурі. Пояснюється це тим, що такі мікроконтролери простіше виготовляти, вони дешевші та мають більший попит у розробників електронної техніки.

1.2 Однокристальні мікроконтролери з RISC архітектурою

Особливість МК, які використовували RISC архітектуру, — виконання всіх команд проходить за один-три такти. А у мікроконтролерах із CISC архітектурою — за один-три машинних цикли, кожен із яких складається з кількох тактів. Відповідно до цього, у RISC мікроконтролерах значно більша швидкодія. Проте, завдяки більш наповненій системі команд CISC контролерів, у деяких випадках можна зекономити час виконання певних фрагментів програми, а також — пам'ять програм. Ще одна особливість МК RISC архітектури — у цих процесорах скорочується до мінімуму набір команд, що виконуються. МК AVR фірми Armel або МК PIC16 фірми Microchip — і є такими мікроконтролерами.

Типовими представниками RISC мікропроцесорів є:

- PIC-мікроконтролери;
- AVR-мікроконтролери.

1.2.1 PIC-мікроконтролери

Розробкою та виготовленням PIC-мікроконтролерів займається відома американська компанія MicroChip Technology Inc. МК побудовані за RISC архітектурою та застосовуються системами керування високошвидкісними двигунами, у приладах побутової електроніки та системах охорони. Шляхом скорочення кількості команд (до трьох десятків), місце для резервування у пам’яті складає «одне слово». Час проходження кожної команди відповідає чотирьом тактам одичного циклу, який триває 200 наносекунд за частоти 20 МГц; винятком є лише команди розгалуження. Оперативна пам'ять виконана за схемою довільної вибірки та можливості безпосередньої адресації у коді команди до будь-якого регістру. Реалізація стеку здійснена апаратно з глибиною два, вісім чи шістнадцять регістрів. Більша частина PIC-контролерів доповнена системою переривань, джерелом яких

може бути таймер. Особливістю PIC МК є спеціальний біт захисту, що передбачає нелегальне копіювання даних.

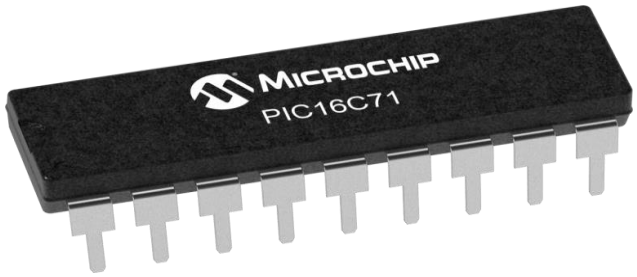

Рисунок 2. Мікроконтролер PIC16C71

В інтегральні схеми PIC16Cxx серії вбудовані ОЗП місткістю до 256 байтів і ПЗП місткістю до 10 Кбайт. Переважна більшість МК мають однократно програмований ПЗП. Особливі МК оснащені ПЗП з ультрафіолетовим або електричним стиранням. Контролер PIC16C71 має внутрішній 8-розрядний АЦП із пристроєм вибирання чи зберігання та вхідний аналоговий мультиплексор. Схожий МК PIC16C64 налічує додатковий вихід з ШІМ, а також послідовний двоспрямований синхронний порт з інтерфейсами.

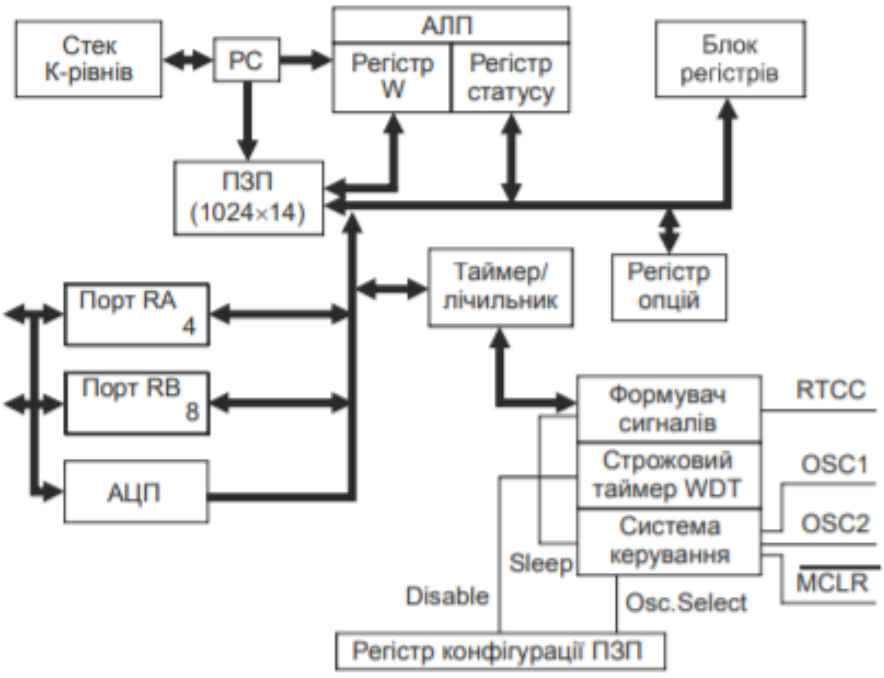

Рисунок 3. Архітектура PIC-мікроконтролерів PIC16C71

1.2.2 Однокристальні AVR-мікроконтролери

Розробники з норвезької фірми Atmel Corporation створили AVR-мікроконтролери. Їх ініціали й сформували бренд AVR. Особливість МК AVR — їх широкий вибір, що надає можливість користувачу обрати мікроконтролер із потрібними характеристиками та функціоналом. Наразі в серійному виробництві існує три різновиди AVR: Mega, Classic та Tiny. Вважається, що використання мікроконтролерів лінії Tiny є найбільш економічною і з найпростішою структурою, лінія Mega є найпотужнішою, а лінія Classic демонструє проміжну позицію між Tiny і Mega. Згідно з останніми даними припинений випуск мікроконтролерів серії Classic.

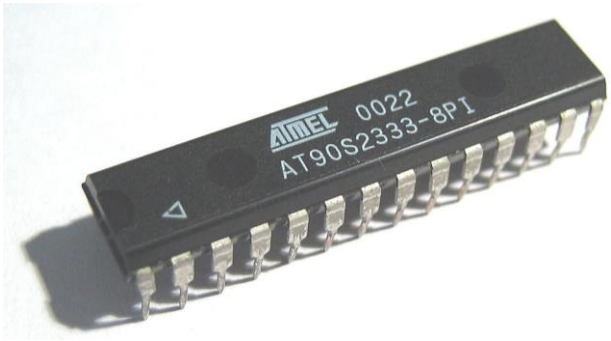

Рисунок 4. Мікроконтролер AT90S2333

Переваги AVR:

- Потужність периферії;
- Швидкодія;
- Простота інтерфейсів;
- Популярність.

Недоліки AVR:

- Складність взаємодії 8-розрядності — для 16-розрядних операцій;
- Великий набір команд;
- Мала кількість режимів.

AVR-мікроконтролер містить гарвардський процесор, регістровий файл, два типи енергонезалежної пам'яті, оперативну пам’ять, різну периферію тощо.

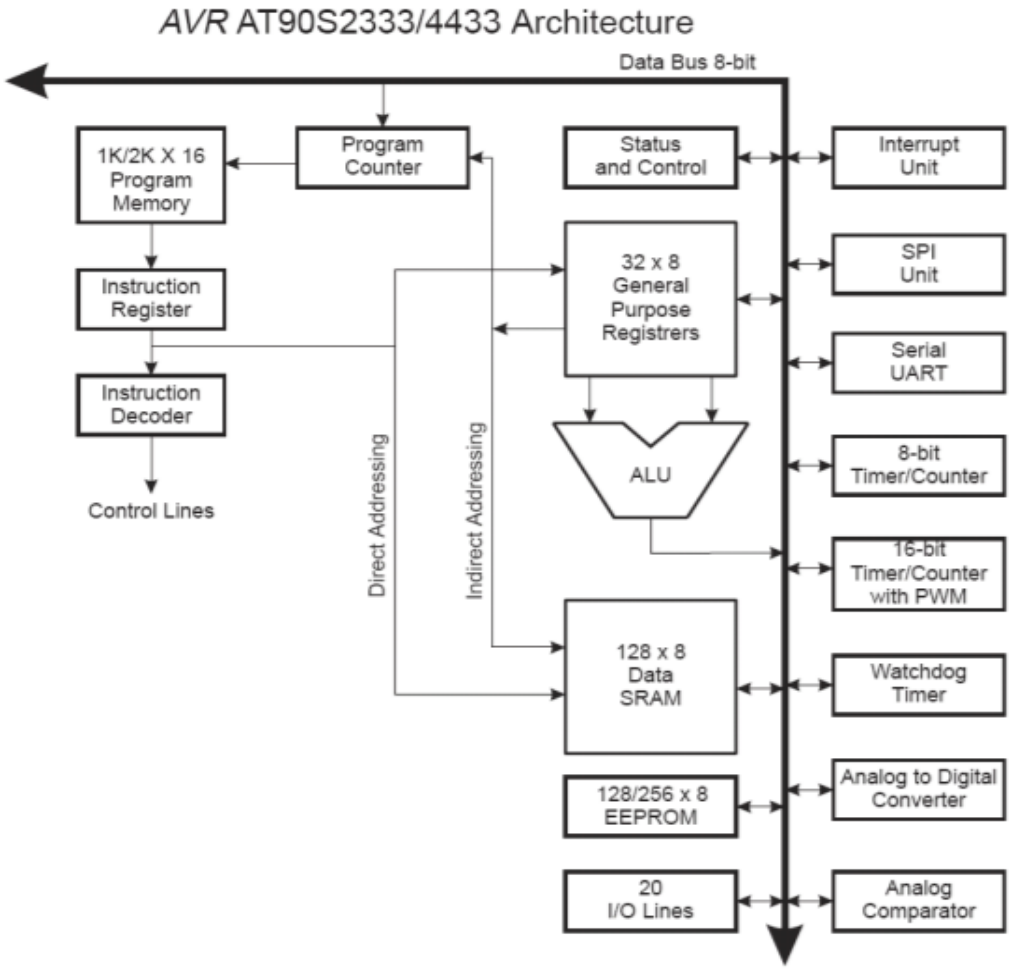

Рисунок 5. Архітектура AVR-МК AT90S2333/4433

1.2.2.1 Мікропроцесор

Головним елементом МК AVR вважається центральний процесорний пристрій (мікропроцесор), побудований за RISC архітектурою. Головною частиною цього модулю є арифметико-логічний пристрій. Виконання системного тактового сигналу з пам'яті програм до лічильника команд упорядковується кожною наступною командою, яка виконується АЛП. Коли команда вибирається з пам’яті програми, буде виконана раніше вибрана команда. Такий принцип дозволяє досягнути максимальної швидкодії.

До РЗП підключається арифметико-логічний пристрій. Всього існує 32 регістри загального призначення, вони мають байтовий формат, тобто кожен регістр складається з 8 бітів. РЗП розташований на початку адресного простору RAM, але не є його фізичною частиною. Існує два способи доступу до регістрів: пам'ять або регістри. Ця особливість підвищує ефективність і продуктивність роботи мікроконтролера.

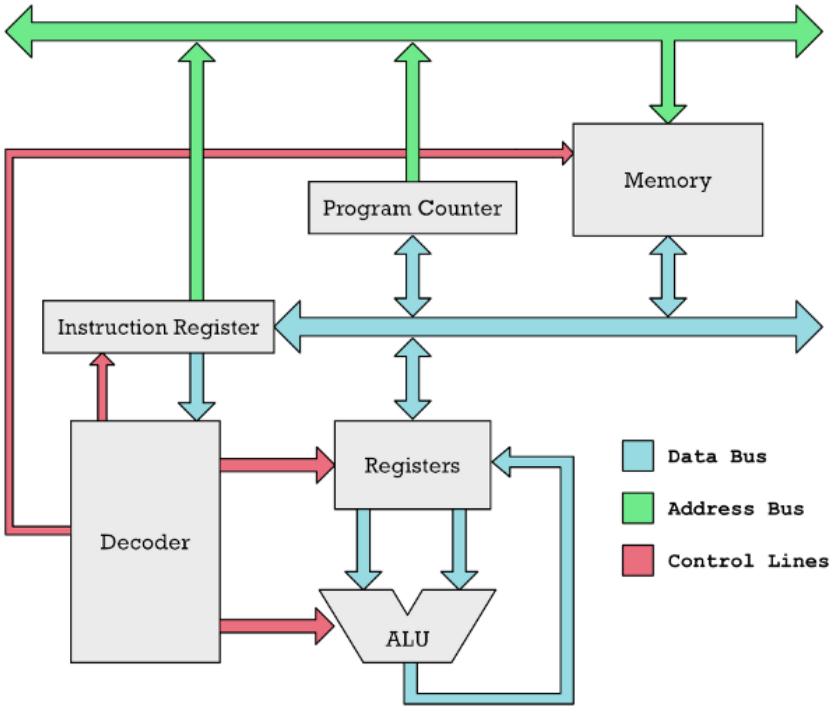

Рисунок 6. Структурна схема МП

Різниця між регістрами та пам’яттю з довільним доступом полягає в тому, що регістри можна використовувати для будь-яких операцій: арифметичних, логічних чи порозрядних, а в оперативну пам’ять можна записувати лише дані з регістрів.

1.2.2.2 Пам'ять

Мікроконтролери AVR розроблені з модифікованою Гарвардською архітектурою, яка не тільки розділяє адресний простір для пам'яті даних і даних, а й розділяє шини доступу до них. Є дві шини даних, перша має доступ до всіх даних, а

інша, шина введення/виведення даних, має обмежений доступ до певного обсягу пам’яті. Розроблена архітектура пам'яті дозволяє процесорам отримувати доступ одночасно до пам'яті програм і пам'яті даних. Це демонструє збільшення продуктивності МК в порівнянні з архітектурою CISC, де ЦП використовує одну і ту ж шину для доступу до пам'яті програм та пам'яті даних.

Можна виділити два основні види пам'яті, які використовуються в мікроконтролерах:

1. Пам'ять програм (FLASH ROM)

Зберігання послідовностей команд, які керують функціоналом МК, розділяється відповідно до пам'яті програм. Контролер AVR містить внутрішню системну перепрограмовану флешпам'ять для зберігання програм розміром від 0,5 до 256 Кбайт. Із метою підвищення ефективності програмного забезпечення простір пам’яті Flash-програм розділено на дві частини: завантажувальний розділ та розділ прикладних програм у пристроях. Перевага Flash-пам’яті полягає в тому, що вона електрично перепрограмована і інформацію можна переписувати тисячу разів знову і знову. Програми можна зберігати у Flash-пам'яті двома способами: через програматор або SPI-інтерфейс.

Різні AVR-мікроконтролери сімейства Mega здатні самостійно змінювати вміст пам'яті програм. Функція самопрограмування дозволяє розвивати гнучкі системи, які в майбутньому будуть змінюватись самим мікроконтролером на основі навколишнього середовища.

2. Пам'ять даних

Ключовою складовою AVR МК є 8-бітна шина даних, що зв'язує між собою RAM, лічильник команд, регістри стану й периферійних пристроїв, РЗП та АЛП мікроконтролера.

Пам'ять даних розділена на три частини:

a) Регістрова пам'ять

Вона складається з 32 РЗП, які є згрупованими в файл службовими регістрами вводу-виводу. Для регістрів вводу-виводу виділено 64 байти. Обидва розташовані в адресному просторі RAM, але не є його частиною.

В області регістрів вводу-виводу є різні регістри: службові, регістри стану, управління периферією тощо. Тобто управління мікроконтролером полягає в управлінні вищезгаданими регістрами.

b) Енергонезалежна пам'ять даних (EEPROM)

Це особливий тип внутрішньої пам'яті, виготовлений за Flash-технологією і призначений для довготривалого зберігання даних.

Більшість МК AVR мають EEPROM для напівпостійного зберігання даних. Подібно до флешпам'яті, EEPROM може зберігати свій вміст за умови відключення живлення. Розмір EEPROM коливається від 64 до 4096 байтів. Типи пам'яті, безпосередньо доступні програмам МК, зручні для зберігання проміжних даних, констант, ключів тощо.

У більшості версіях AVR ця пам'ять не відображається в адресному просторі ЦП, тому до адреси МК звертаються не так, як до решти видів пам'яті. Доступ можна здійснити лише за допомогою спеціальних регістрів-вказівників та інструкцій читання-запису, що робить доступ до EEPROM набагато повільнішим, ніж до інших внутрішніх ОЗП. Пам'ять EEPROM також має обмеження на кількість записів – Atmel вказує 100 000 записів. Реєстратор EEPROM порівнює вміст адреси EEPROM з потрібним вмістом і виконує фактичний запис, лише якщо вміст потрібно змінити.

c) Статична оперативна пам'ять (SRAM)

Внутрішня оперативна пам'ять у байтовому форматі для зберігання даних.

Залежно від серії розмір оперативної пам'яті варіюється від 64 байт до 32 КБ (найбільший розмір у серії ATxmega). Кількість циклів читання та запису в статичній ОЗП необмежена. Вимкнення напруги живлення призведе до втрати інформації без можливості відновлення.

1.2.2.3 Периферія

Периферія AVR мікроконтролерів містить у собі:

a) **У портів вводу-виводу** є певна кількість незалежні лінії вводу-виводу, що варіюються в межах від 3 до 53. Будь-яку з ліній порту можна запрограмувати на вхід або на вихід. Потужні вихідні драйвери забезпечують навантажувальну здатність до декількох десятків mA на лінію порту. Це дозволяє безпосередньо підключати світлодіоди та біполярні транзистори до МК. Максимально допустиме струмове навантаження на всі лінії порту не повинне перевищувати 80 mA за напруги 5 V. Особливістю побудови архітектури AVR портів вводу-виводу є те, що кожен фізичний вихід має 3 біти контролю-керування, хоча у більшості 8-розрядних МК відомих компаній їх налічується всього 2. Таке впровадження прибирає необхідність створення копій вмісту порту та підвищує продуктивність МК під час роботи із зовнішніми пристроями та зовнішніми електричними завадами.

b) **Таймери та лічильники** МК AVR мають у своєму складі від 1 до 4 таймерів або лічильників з 8-ми чи 16-ти бітовою розрядністю відповідно. Вони здатні працювати як таймери від внутрішнього джерела, так і як лічильники зовнішніх подій. Їх використовують для замірів точних часових інтервалів, формування послідовності імпульсів, а також імпульсів суми на виходах МК тощо. Широтно-імпульсний модулятор передує режиму ШІМ і генерує сигнал з програмною частотою і шпаруватістю. Таймери та

лічильники здатні обробляти запити переривань, перемикаючи процесор на їх обслуговування по подіях. А також звільняють його від необхідності періодично перевіряти стан таймерів. Саме завдяки використанню в реальному часі вони є найбільш важливими елементами.

c) **Сторожовий таймер** призначений для запобігання непередбачених проблем у ході виконання програми, а також від випадкових збоїв програми. У нього є власний RC генератор, що працює на частоті, наближеною до 1 МГц, і залежить від величини температури та напруги живлення. За правильної роботи програми, команда сторожового таймера регулярно скидається та запобігає скидання процесора.

d) **Аналоговий компаратор** призначений для того, аби порівнювати напругу на двох входах МК. У результаті його роботи буде логічне значення, яке може бути прочитати програма. За фронтом можна встановити спрацювання переривання.

e) **Переривання** в МК AVR реалізовані багаторівневою системою переривань. Вони зупиняють правильний хід програми для виконання пріоритетної задачі, що визначається внутрішньою чи зовнішньою подією. За виявленням події, яка викликає переривання, в МК збережено вміст лічильника команд, потім виконання ЦП переривання заданої програми й виконується обробку переривань підпрограми. За результатом виконання підпрограми переривання відбувається відновлення попередньо збереженого лічильника команд ЦП. І як наслідок, МК повертається до виконання перерваної програми.

f) **Аналого-цифровий перетворювач** призначений аби отримувати числові значення напруги на вході, що в результаті зберігається в регістр даних АЦП.

g) **Універсальний послідовний приймач** є зручним та простим послідовним інтерфейсом для організації інформаційного каналу обміну МК із зовнішнім світом. Одночасно приймає та передає дані. Аби з'єднати МК та персональний комп'ютер, знадобиться схема сполучення рівнів сигналів.

h) **Послідовний периферійний інтерфейс** призначений для організації обміну даних між двома пристроями. Завдяки йому зручно робити обмін даними між деякими МК AVR.

i) **Двопровідний послідовний інтерфейс** повний аналог двопровідної двоспрямованої шини фірми Philips. Він дозволяє з'єднати до 128 різних пристроїв разом та складаються з ліній даних та ліній тактового сигналу.

j) **Тактовий генератор** створює імпульси для синхронізації роботи всіх вузлів МК. Його запускають завдяки джерелам опорної частоти, а саме: RC-комірок, зовнішнього генератора або кварцового генератора. Мінімальної допустимої частоти не існує, а максимальна залежить від конкретного типу МК Armel та його характеристик. Можливі випадки розгону до більшої частоти генератора.

k) **Система реального часу** присутня в усіх МК серії Mega. У таймера-лічильника є окремий переддільник, який може бути ввімкнений програмним способом до джерела основної тактової частоти, або до додаткової асинхронної опорної частоти.

1.2.2.4 Живлення

Документація AVR свідчить, що мікроконтролери функціонують за напруги живлення в межах від 1,6 до 5,5 В. Споживання струму в активному режимі залежить від напруги живлення та частоти МК. Значення струму менше ніж 1 мА на частоті до 500 кГц, 5-6 мА на 5 МГц і 8-9 мА на 12 МГц.

Усі моделі мікроконтролерів сімейства підтримують лише два різні «сплячі» режими: режим очікування і режим мікроспоживання. Мікроконтролери серії ATtiny15L мають особливість та підтримують ще третій режим — режим зниження шумів АЦП.

Як і у всіх МК AVR, безумовно можна вибрати один з двох можливих «сплячих» режимів:

- **Режим очікування** припиняє роботу процесора і фіксує вміст пам'яті даних, тоді як внутрішній генератор сигналів синхронізації, таймери, система переривань та сторожовий таймер продовжують працювати. За частоти 12 МГц споживаний струм не перевищує 2,5 мА.
- **Режим мікроживлення** зберігає вміст реєстрового файлу, але внутрішній генератор синхронних сигналів зупиняється, тому всі функції припиняються, доки не буде отримано сигнал зовнішнього переривання або сигнал апаратного скидання. За напруги живлення 5 В споживаний струм сторожового таймера в режимі мікроживлення становить близько 80 мкА, і менше ніж 1 мкА, коли він вимкнений.

1.3 Широтно-імпульсна модуляція

Широтно-імпульсна модуляція (ШІМ) — це метод перетворення сигналу, основною ідеєю якого є зміна тривалості імпульсу (шпаруватість), а частота відповідно залишається константою. Сигнал, модульований по ширині імпульсу, формується двома способами: аналоговим та цифровим.

Основною причиною застосування ШІМ є прагнення до підвищення коефіцієнта корисної дії (ККД) при побудові вторинних джерел живлення електронної апаратури. Наприклад, її використовують для регулювання яскравості підсвічування LED-екранів та моніторів або коли налагоджують потужність швидкості моторів електродвигунів.

1.3.1 Принцип роботи ШІМ

Аналоговий ШІМ-сигнал формується аналоговим компаратором: на один вхід надходить опорний сигнал з частотою значно вище модулюючого сигналу. Сигнал матиме трикутну або пилкоподібну форму, а на іншій вхід — модулюючий безперервний сигнал. Частота вихідних імпульсів ШІМ відповідає частоті модулюючого сигналу. Протягом того часу, коли позитивний вхідний сигнал вищий за негативний вхідний сигнал, — вихід формує логічну «1», а в іншому випадку — «0». На виході виходить дискретний сигнал, частота якого відповідає частоті опорного сигналу, а тривалість імпульсу пропорційна рівню модулюючої напруги.

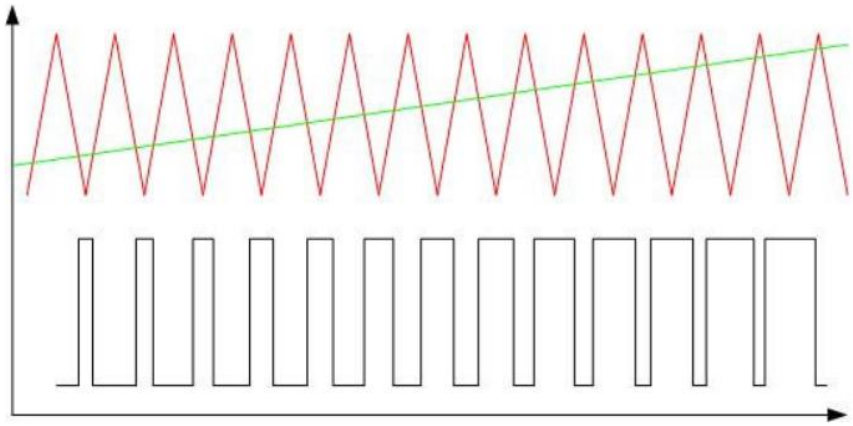

Рисунок 7. Модуляція по ширині імпульсу трикутного сигналу

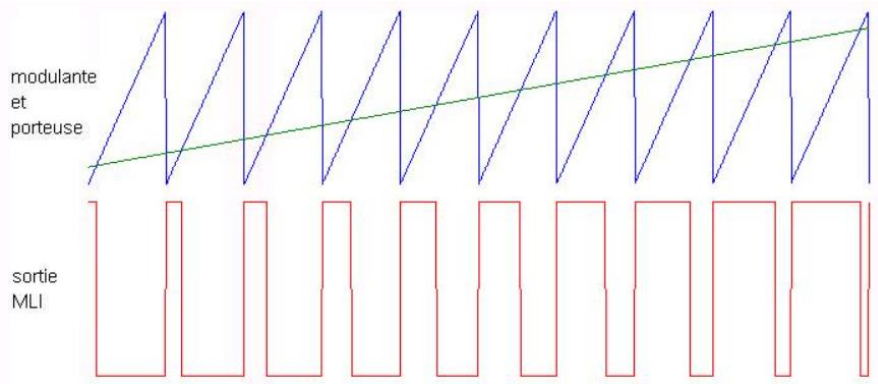

Рисунок 8. Модуляція по ширині імпульсу пилкоподібного сигналу

Аналогова ШІМ використовується у підсилювачах низької частоти D-класу.

У двійковій цифровій технології, де вихід може приймати лише одне з двох значень. Цілком природно використовувати ШІМ для наближення до бажаного середнього рівня вихідного сигналу. Неймовірно прості в експлуатації пилкоподібні або трикутні сигнали генеруються n-бітовим лічильником. Цикл поділено на ділянки, заповнені прямокутними підімпульсами. Середнє значення періоду залежить від кількості прямокутних підімпульсів.

Цифрова ШІМ є приближенням бінарного сигналу до неперервного сигналу середнього або максимально наближеного значення проміжку часу. Прямокутні підімпульси, що заповнюють період, можуть стояти будь-де в ньому. На середню

величину у періоді впливає лише їх кількість. Наприклад, при розбитті періоду на 8 частин послідовності 11110000, 11100100, 11100010, 11100001 та ін., дають однакову середню за період величину. Проте окремі одиниці погіршують режим роботи ключа.

Рисунок 9. Приклад цифрового ШІМ сигналу з n-частин підімпульсів

1.3.2 Переваги та недоліки ШІМ

Переваги використання:

- Висока ефективність сигналу;
- Мала втрата енергії;
- Висока надійність та стабільність роботи;
- Низька вартість використання.

Недоліки використання:

- За низької робочої частоти людське око відчуває мерехтіння;
- Поява комутаційного шуму.

1.3.3 Застосування ШІМ

ШІМ застосовують у вторинних джерелах живлення, енергетиці, регуляторах потужності, регуляторах яскравості джерел світла, швидкості обертання колекторних двигунів, мобільних пристроях тощо. У цих випадках використання ШІМ дозволяє значно підвищити ефективність роботи системи та спростити її реалізацію.

Крім цього, щоб отримати аналоговий сигнал завдяки цифровому виходу МК, аналогом є цифро-аналоговий перетворювач. Реалізація проста та вимагає мінімум зовнішніх компонентів, у більшості випадків достатньо RC-кола.

1.4 Драйвери, модулі та датчики

Існують додаткові можливості роботи мікроконтролера: вихід до мережі інтернету, зв'язок з іншими пристроями за допомогою технології Bluetooth, навчитися керувати двигунами, працювати з промисловим обладнанням, відстежувати зміну фізичних величин. Для їх реалізації варто додати драйвери, модулі та датчики. Прикладами компонентів є сервоприводи, датчики відстані або вологості, модулі керування або обслуговування тощо.

1.4.1 Bluetooth модуль HC-05

Технологія Bluetooth використовується для передачі даних між двома пристроями, які знаходяться поблизу один від одного. Ця технологія забезпечує хорошу стійкість до широкосмугових перешкод, дозволяючи багатьом пристроям, що знаходяться поблизу, спілкуватися один з одним, не заважаючи один одному. Технологія широко використовується в мобільних телефонах, планшетах, ноутбуках, гарнітурах та смарт-пристроях.

Одним з найкращих рішень для встановлення двостороннього зв'язку з планшетом, ноутбуком або іншим Bluetooth-пристроєм є модуль Bluetooth HC-05, який має два режими роботи: master (шукає Bluetooth-пристрої та ініціює зв'язок) і slave (відомий пристрій).

Рисунок 10. Bluetooth-модуль HC-05

Модуль має 6 виводів:

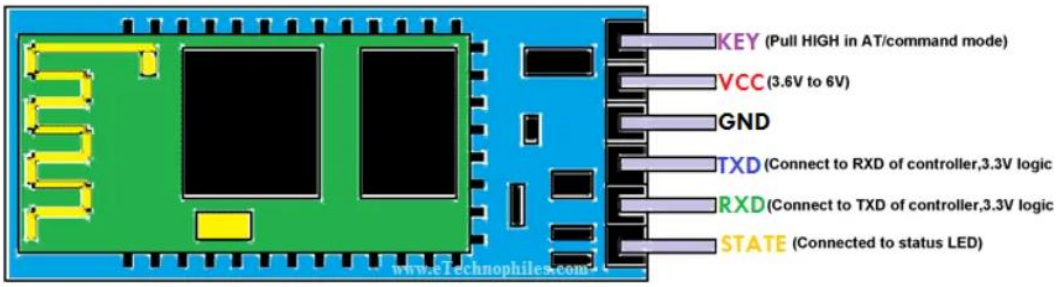

Рисунок 11. Розпінування Bluetooth модуля HC-05

Налаштування модуля здійснюється в режимі програмування надсиланням AT-команд по послідовному порту. Щоб увійти в режим програмування, на контакт KEY необхідно подати високий сигнал. Деякі модулі HC-05 сприймають AT-команди тільки при фізично натиснутій кнопці. Інтерфейсом модуля є послідовний порт (COM).

Характеристика	**Значення**
Діапазон частот радіозв'язку	2,4-2,48 ГГц
Чутливість	-80 dBm
Напруга живлення	3,3-6 V
Споживаний струм	40 mA
Радіус дії	До 10 м у закритому просторі; до 30 м на відкритому
Робочий діапазон температур	-25 ... 75 С
Розмір	27 х 13 х 2,2 мм

Таблиця 2. Технічні характеристики Bluetooth модуля HC-05

Принцип роботи полягає в взаємодії модуля зв'язку з пристроєм керування шляхом двосторонньої передачі даних. Для цього потрібно синхронізуватись з мобільним пристроєм за допомогою Bluetooth та стандартним паролем «1234».

1.4.2 Ультразвуковий датчик відстані HC-SR04

Цей далекомір може бути чудовим датчиком для платформи розробки, завдяки якому він зможе визначати відстані до об'єктів, об'їжджати перешкоди та будувати карту приміщення. Його можна також використовувати як датчик сигналізації, що спрацьовує при наближенні об'єктів.

Ультразвуковий модуль визначення дальність HC-SR04 забезпечує безконтактне вимірювання в межах від 2 до 400 см, точність дальності може досягати 3 мм. Модулі включають ультразвукові передавачі, приймач і схему управління.

Рисунок 12. Ультразвуковий датчик відстані HC-SR04

Датчик відстані працює подібно до ехолокації у дельфінів і кажанів — він визначає відстань до об'єкта перед собою або наявність перешкоди. Модуль виробляє звукові імпульси на частоті 40 кГц та прослуховує echo. За той час, коли звукова хвиля рухається вперед і назад, ви можете остаточно визначити відстань до об'єкта.

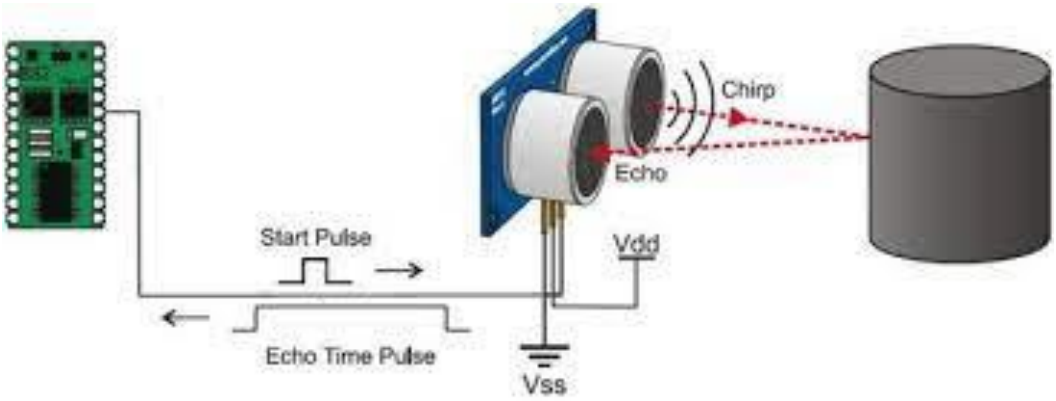

Рисунок 13. Принцип роботи ультразвукового датчика відстані HC-SR04

Таким чином, відстань до об'єкта можна розрахувати за формулою (1):

$$distance = \frac{time \times speed}{2}, \quad (1)$$

де distance — відстань до об’єкта;

time — виміряний час імпульсу;

speed — швидкість звуку (~340 м/с).

Датчик відстані HC-SR04 має певні технічні характеристики, знання цих характеристик дозволить правильно спроєктовувати пристрої з фізичної точки зору.

Характеристика	Значення
Робоча напруга DC	5 V
Робочий струм	15 mA
Робоча частота	40 Гц
Мінімальна дальність	2 см
Максимальна дальність	400 см
Кут вимірювання	15 градусів
Вхідний сигнал тригера, TTL імпульс	10 uS
Розмір	45*20*15 мм

Таблиця 3. Технічні характеристики датчика HC-SR04

1.4.3 Драйвер двигуна L298N

Драйвер двигуна виконує дуже важливу роль у проєктах, що використовують двигуни постійного струму або крокові двигуни. За допомогою мікросхеми драйвера L298N можна створювати роботів, автономні автомобілі та інші пристрої з механічними модулями.

Схема модуля складається з двох H-мостів, що дозволяють підключати двополярний кроковий двигун або два щіткові двигуни постійного струму одночасно.

Швидкість і напрямок обертання двигуна можна змінювати за допомогою ШІМ-сигналу. Управління здійснюється шляхом подачі відповідних сигналів на командні входи, які є керованими.

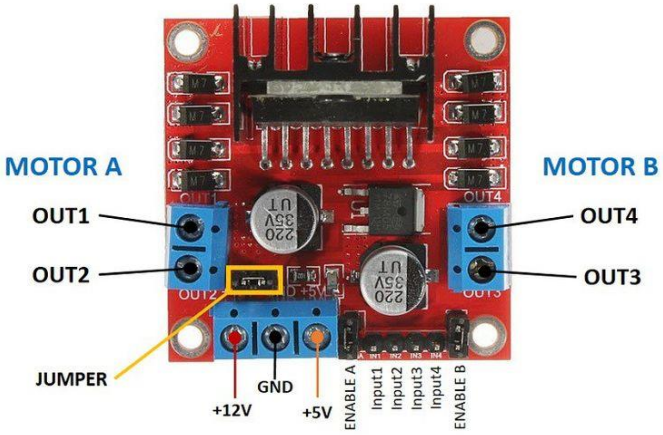

Рисунок 14. Драйвер двигуна L298N

Опис та призначення контактів:

- OUT1 та OUT2 – роз'єм для підключення першого щіткового двигуна або першої обмотки крокового двигуна;
- OUT3 та OUT4 – роз'єм для підключення другого щіткового двигуна або другої обмотки крокового двигуна;
- VSS(+ 12 В) – вхід для живлення двигунів, максимальна напруга 35 В;
- GND – загальний провід, бажано підключати також аналоговий вхід;
- Input1, Input2 – контакти керування першим щітковим двигуном або першою обмоткою крокового двигуна;
- Input3, Input4 – контакти керування другим щітковим двигуном або другою обмоткою крокового двигуна;
- JUMPER – режим , наявність клеми повідомляє про режим;
- Vs(+ 5 В) – вхід для живлення + 5 В. Через нього безпосередньо запитується сама мікросхема L298N після стабілізації;
- ENABLE A, ENABLE B – контакти для включення-вимкнення 1-го і 2-го двигунів. До цих контактів подається ШІМ-сигнал, щоб змінити швидкість

щіткового двигуна. Як правило, ці контакти перемикаються, щоб забезпечити постійне підживлення + 5 В.

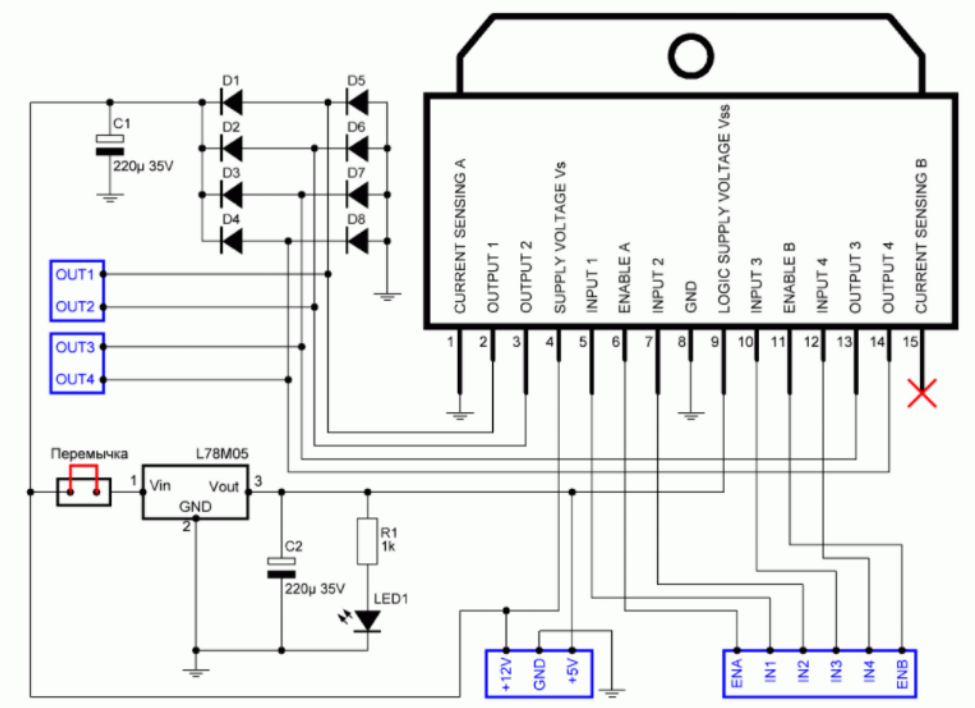

Рисунок 15. Електрична схема драйвера L298N

Драйвер двигуна L298N має певні технічні характеристики, і розуміння цих характеристик дозволить спроєктувати пристрій фізично правильно.

Характеристика	Значення
Напруга живлення логіки	5 V
Споживаний логікою струм	36 mA
Напруга живлення двигунів	Від 5 до 35 V
Робочий струм драйвера	2 A
Піковий струм драйвера	3 A
Максимальна потужність	20 Вт (при температурі 75 C)
Діапазон робочих температур	-25 C ... +135 C
Розмір	43.5 x 43.2 x 29.4 мм

Таблиця 4. Технічні характеристики драйвера L298N

1.5 Платформи розробки

Розробка демонстраційної моделі вимагає вибору платформи розробки, на основі якої буде розроблятись пристрій. Розділити платформи для розробки можна на дві частини:

- Плати на мікроконтролері
- Одноплатні комп'ютери

Можливі гібридні платформи, коли на платі розташований мікроконтролер і процесор. Ідея цілком зрозуміла, залишити потужні складні завдання – процесору, такі як: вихід до мережі, оброблення медіасигналів тощо. А мікроконтролеру функцію точного керування, наприклад, керування приводами чи датчиками.

	Плати на МК	**Одноплатний комп'ютер**
Продуктивність	1 ядро, сотні МГц, десятки КБ оперативної пам'яті, сотні КБ постійної пам'яті.	Від 1 ядра, тисячі МГц, сотні МБ оперативної пам'яті, ГБ-ти постійної пам'яті.
Багатозадачність	Ні.	Так.
Швидкість реагування	Повний контроль над часом і тривалості сигналів.	Можливі проблеми через багатозадачність.
Доступ до мережі	Потрібні додаткові модулі.	Завдяки вбудованому мережевому адаптеру.

Робота від батареї	Споживання десятків мА. Час роботи до місяця в економному режимі.	Споживання сотні-тисячі мА. Час роботи великого акумулятора до одного дня.
Мови програмування	Paskal, C / C++, Assembler.	Perl, C / C++, Java, JavaScript, Python та інші.

Таблиця 5. Порівняння параметрів мікроконтролера та одноплатного комп'ютера

МК може одночасно виконувати лише одну задачу, тому й відмінно працює. Одноплатні комп'ютери виконують програми в рамках операційної системи й мають велику продуктивність. Продуктивність мікроконтролерів не сильно поступаються, а в чомусь навіть перевершують одноплатні комп'ютери. Великим плюсом мікроконтролерів є велика тривалість роботи від батарейок коштом меншого енергоспоживання, а також краща швидкість реакції, менша вартість. Отже, для розробки проєктів з вищезазначеними особливостями доцільніше обирати плати на МК, ніж на одноплатному комп'ютері.

1.5.1 Мікроконтролер Arduino Uno

Платформа Arduino Uno побудована на основі мікроконтролері ATmega328, що показує гарний приклад плати МК. Щоб взаємодіяти з платою, вам знадобиться під'єднати платформу розробки до акумулятора або використати адаптер змінного/постійного струму, наприклад, живити від ноутбуку. Основою плати є кварцовий генератор з частотою 16 МГц, що живиться через роз'єм USB або роз'єм живлення. Також платформа має 6 аналогових входів та 14 цифрових входів-виходів з яких 6 підтримають ШІМ-сигнали. Чудовою дрібницею є вбудований світлодіод та кнопка скидання, що набагато спрощує роботу.

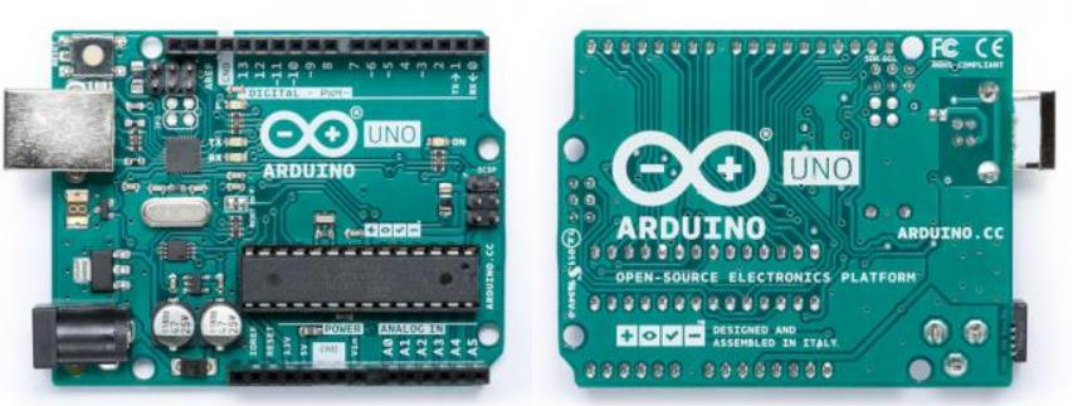

Рисунок 16.Плата Arduino Uno з обох боків

Мікроконтролер живлять через зовнішнє живлення чи USB-з'єднання Джерелом зовнішнього живлення є акумулятор або блок живлення Також при стабілізації напруги з мережі можна під'єднатися через роз'єм 2,1 мм з центральним плюсовим полюсом. Акумулятор напряму з'єднують до виходів Gnd і Vin, що продемонстровані на платі.

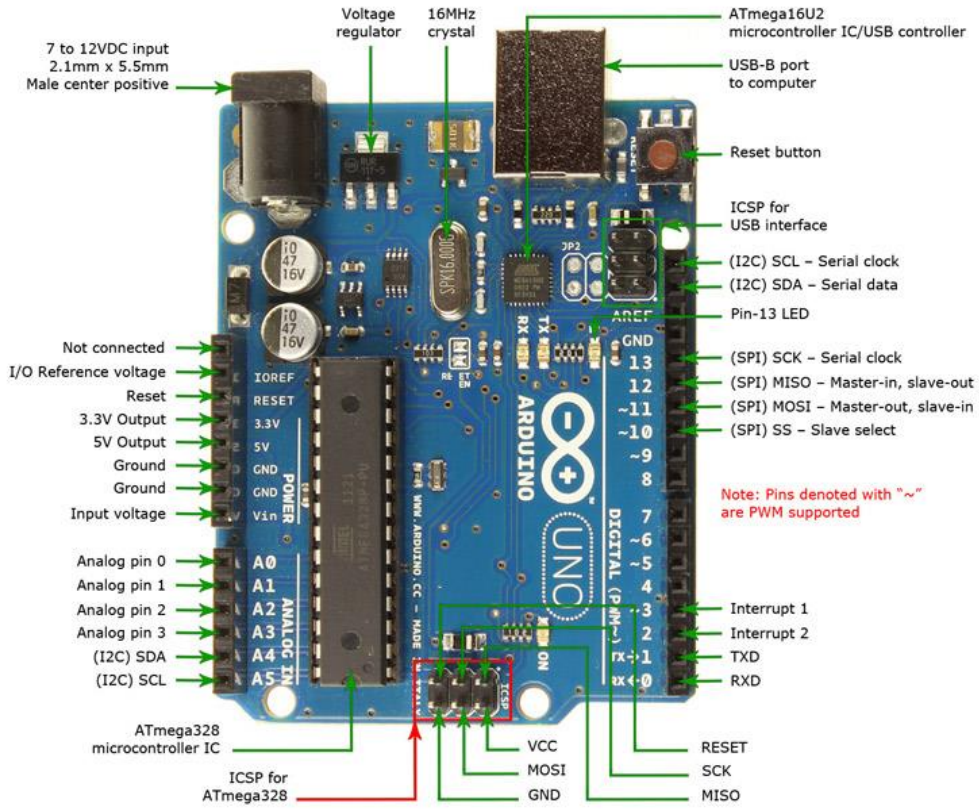

Рисунок 17. Розпінування плати Arduino Uno

Діапазон працездатної роботи платформи від зовнішнього живлення в межах 6-20 В, але умови стабільної роботи неможливе при граничних значеннях напруги. Для стабільної роботи рекомендований діапазон напруги живлення 7-12 В. Подання напруги вище як 12 В, може перегріти регулятор та завдати шкоди платі. Живлення нижче ніж 7 В, може видавати напругу на вихід нижче ніж 5 В, результатом чого МК працюватиме нестабільно.

Перелік входів-виходів живлення та їх означення:

- Вхід VIN призначений для живлення від зовнішнього джерела живлення;
- Вихід 5 V застосовується для живлення МК та інших компонентів. На виході напруга дорівнює значенню 5 В;
- Вихід 3,3 V видає вихідну напругу 3,3 В та максимальний струм споживання дорівнює 50 mA;
- Вихід IOREF повідомляє про робочу напругу. Залежно від напруги з виходу визначається сумісність, що дозволяє пристосуватися під напругу 5 або 3,3 В;
- Контакт GND відповідає за землю.

Головною складовою плати є мікроконтролер ATmega328, що має флешпам'ять розміром 32 КБ, з яких 2 КБ SRAM, 1 КБ EEPROM та 0,5 КБ для зберігання завантажувача.

Arduino Uno нараховує 14 цифрових контактів, що можна налаштувати як вхід або вихід. Налаштування контактів відбувається в інтегрованому середовищі розробки за допомогою функцій: digitalRead(), pinMode() та digitalWrite(). Виходи працюють при напрузі 5 В та мають внутрішні підтягувальні резистори опором в межах 20-50 кОм на кожному виході. Особливістю деяких контактів є додаткові функції:

- Послідовна шина відповідає за 0 і 1 пін. Задачею цих виходів є отримання RX і передачі TX даних TTL;
- Зовнішні переривання налаштовані на 2 і 3 піні. Ці виходи можна налаштувати на виклик переривання або на молодшому значенні, або на передньому чи задньому фронті, або при зміні значення. Для повного спектра можливостей варто звернутися до опису функції attachInterrupt();
- Виходи: 3, 5, 6, 9, 10, і 11, відповідають за ШІМ-сигнали. Кожен з виходів забезпечує ШІМ-сигнал з роздільною здатністю 8 бітів, що керується за допомогою функції analogWrite();
- Виходи 10, 11,12 та 13 здійснюють зв'язок SPI за допомогою інтерфейсу SPI.
- Цифровий вивід 13 з'єднаний з вбудований світлодіодом на платі. Значення сигналу на контакті відображається на світлодіоді;
- Наявність входу RESET допомагає швидкому перезавантаженню плати та скиданню робочого стану. Формування низького сигналу на виводі призводить до перезавантаження МК;
- Контакт AREF відповідає за опорну напругу аналогових входів. За використанням зовнішньої напруги необхідно використовувати функцію analogReference();
- Функції TWI можна виконувати на аналогових входах A4 та A5. Використанням бібліотеки Wire може здійснювати зв'язок по інтерфейсу TWI.

Плата оснащена 6 аналоговими входами із позначенням A0…A5, кожен може перевести аналогову напругу в 10-бітнє число.

Завантаження інтерфейсу відбувається через USB, а комп'ютер «спілкується» з платою через віртуальний COM-порт. Прошивка МК використовує стандартні драйвери USB COM, але для підключення в операційній системі Windows потрібен файл ArduinoUNO.inf, що є в будь-якому IDE. Послідовний монітор при підключенні

до плати за допомогою USB кабелю дозволяє надсилати та отримувати дані в текстовому вигляді. Більша частина розробників використовує послідовний монітор для аналізу роботи МК.

1.5.2 Одноплатний комп'ютер Raspberry Pi

Raspberry Pi — це компактний одноплатний комп'ютер розміром з банківську картку, розроблений Raspberry Pi Foundation у Великобританії. Основна мета — сприяти засвоєнню учнями елементарних комп'ютерних навичок на базі бюджетної платформи. Raspberry Pi заснований на процесорі з архітектурою ARM 11, який працює на частоті 700 МГц, але має потенціал для розгону до 1 ГГц. Висока продуктивність за низьким споживанням енергії дозволяє досягти максимальної ефективності у розробці.

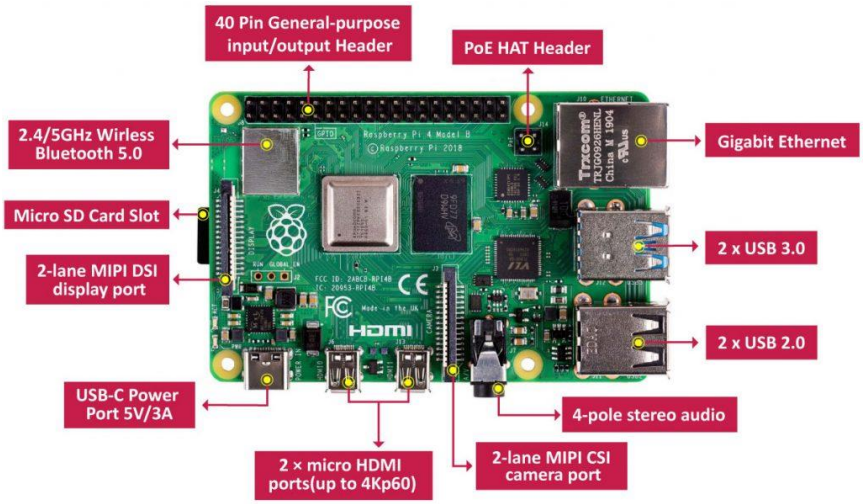

Рисунок 18. Одноплатний комп'ютер Rasberry Pi 4

Пристрій має функціональність справжнього персонального комп'ютера розміром з кредитну картку, при цьому має повноцінний «системний блок» з характеристиками, близькими до масового комп'ютерного століття початку XXI століття. Обладнання дозволяє встановлювати операційну систему Linux (або RiscOS), і ви можете завантажити близько трьох десятків інших операційних систем. У розробці офіційна установка Android, OS Chromium та Puppy Linux. За замовчення

цей одноплатний комп’ютер має операційну систему Debian Linux. Також низку безплатних програм, що за замовченням є на будь-якій операційній системі.

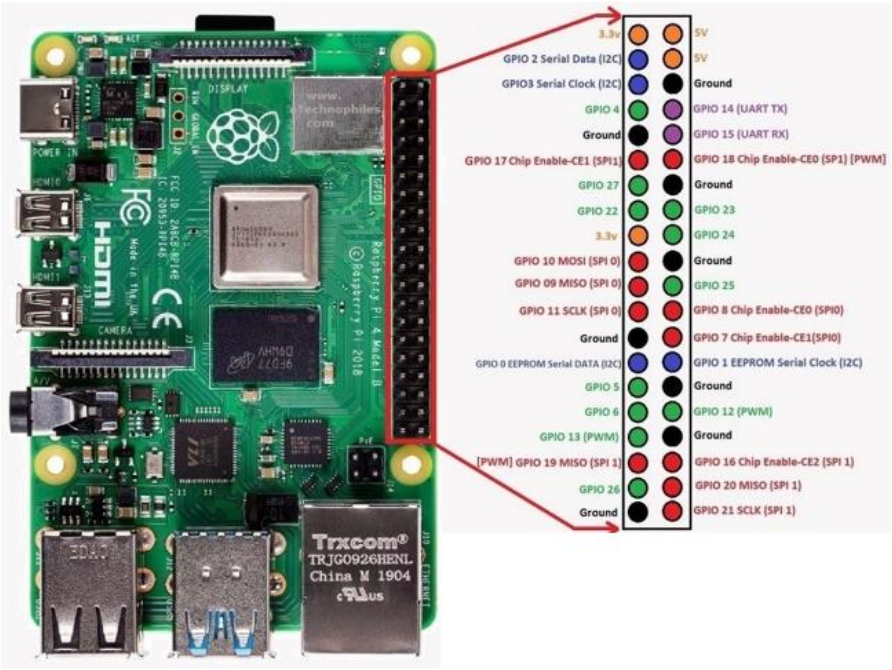

Рисунок 19. Розпінування Rasberry Pi 4 GPIO

Плати Raspberry Pi набули застосування у таких середовищах:

- в освіті: низька вартість та можливість встановлення операційної системи Linux дозволяють розв'язати проблеми навчання програмування. Цифрові входи дозволяють підключати інші елементи, що дозволяє створювати прості електронні пристрої, такі як smart пристрої в різних сферах.
- у системах розумного дому за допомогою сумісних модулів розширення новачки можуть придумати неймовірні проєкти.
- як сервер; Цей пристрій налаштовано так, щоб ви могли використовувати його як вебсервер особистої мережі. Наприклад, ви можете опублікувати власний вебсайт, але це досить повільно.
- у якості маршрутизатора мережі налаштовує дозвіл на маршрутизацію трафіку за використанням стандартних Linux застосувань.

2. Практична частина

2.1 Вибір інтегрованого середовища розробки мікроконтролерів

Кількість інтегрованих середовищ розробки для програмування мікроконтролерів налічується десятками різновидів. Обрання IDE залежить від переліку переваг: простота використання, доступність, мультиплатформність, вартість тощо. Основою складовою створеного макета рухомої платформи є плата Arduino Uno яка потребує середовища розробки. Платформа Arduino має своє власне середовище розробки Arduino IDE, яке є безплатним та абсолютно доступним та відкритим для користування. Іншим можливим варіантом є використання Processing IDE з підключенням бібліотеки Firmata, яка реалізує протокол Firmata наданням контролю плати Arduino. Після аналізу наявних інтегрованих середовищ розробки мікроконтролерів було обрано Arduino IDE для написання логіки МК.

2.1.1 Arduino IDE

Arduino IDE — кросплатформний застосунок розроблений мовою Java, що складається з компілятора, модуля передачі прошивки в плату та редактору коду. Створення середовища розробки відбулося мовою Processing. Можливе імпортування чи доповнення бібліотек за потребами. Обробка коду виконується препроцесором C, а потім відбувається компіляція за допомогою AVR-GCC.

Програми в Arduino IDE пишуть мовою C чи C++. Написана програма називається скетчем. Головною ідеєю є необхідність визначення хоча б двох основних функцій, щоб створити програму, що виконуватиме циклічне повторення коду програми:

- **setup ()**: функція задання початкових параметрів, що призначуються лише на початку роботи програми;
- **loop ()**: періодично виконання функції до вимкнення плати.

Рисунок 20. Стандартний скетч в Arduino IDE

Написання мінімальної логіки підпрограми будь-якої платформи розробки супроводжується щонайменше основними функціями та допоміжними бібліотеками.

2.1.2 Processing IDE

Мова Processing та IDE стали засновниками проєктів, таких як: Arduino, Wiring тощо. Processing використовує мову програмування Java та має додаткові спрощення: додаткові класи, псевдонімні математичні функції та операції. Також містить графічний інтерфейс для спрощення етапу компіляції та виконання. Програми обробки називають скетчами.

Однією з відмінностей від Arduino IDE є функція яку потрібно визначити, яка по суті є аналогічною до функції loop() в Arduino IDE:

• **draw ()**: функція виконує компіляцію скетчів та запускає його в окремому вікні.

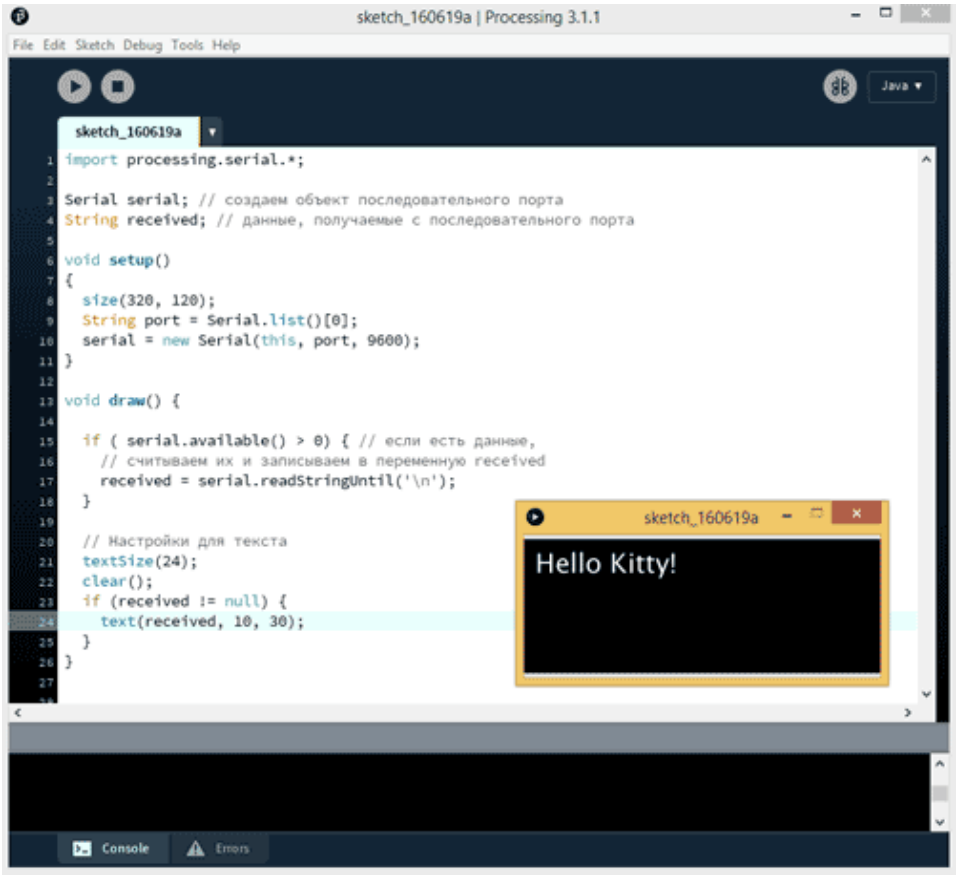

Рисунок 21. Приклад скетчу в Processing IDE

Однією з переваг Processing IDE є можливість розробки програм на мовах вищого рівня ніж C та C++. Основною мовою розробки в даному середовищі є мова Java, але також існує можливість написання скетчів на мові Python. Найбільш вагомою перевагою Processing IDE є можливість налагодження програм(debug), що значно полегшує розробку. Після налагодження скетч може бути завантажений безпосередньо на мікроконтролер.

2.2 Функції та бібліотеки для роботи з Arduino

2.2.1 Основні функції

Платформа Arduino IDE має велику кількість вбудованих функцій, яких цілком достатньо для реалізації стандартної розробки. Розглянемо далі стандартні функції, які були використані у даній дипломній роботі:

- **pinMode(pin, mode)** — налаштовує режим (mode) роботи вказаного контакту (pin). Значення режиму можливе лише одне з двох: INPUT або OUTPUT, встановлює на вхід або вихід відповідно контакту (pin). Виводи аналогового входу можна використовувати як цифрові контакти, які називаються A0, A1, A2 тощо.
- **analogWrite(pin, value)** — записує аналогову величину (value), ШІМ-сигнал, на контакт (pin). Функція набула широкого використання для керування яскравістю підключеного світлодіода або швидкісті обертання електродвигуна на різних швидкостях. Значення (value) відповідне періоду робочого циклу значення між 0 (повністю вимкнено) and 255 (сигнал поданий постійно). ШІМ-виходи на різних платах відповідають різним контактам та позначаються як PWM pins або символом «~».
- **analogRead(pin)** — зчитує значення з аналогового контакту (pin). Плати Arduino налічують від 6 до 16 контактів, 10-розрядний АЦП. Це означає, що напруга подана на аналоговий вхід від 0 до 5 вольтів буде перетворена в цілі значення у діапазоні від 0 до 1023, це дає 1024 кроки з шагом 0.0049 Вольтів (4.9мВ). Діапазон входу та роздільну здатність можна змінити за допомогою допоміжною вбудованою функцією analogReference(). Зчитування аналогового входу займає близько 100 мікросекунд.
- **digitalWrite (pin, value)** — записує значення (value) на цифровий контакт (pin). Значення (value) контакту можливе LOW чи HIGH. Якщо контакт був

налаштований як OUTPUT з pinMode (), його напруга буде встановлена: 5 В (або 3,3 В на платі з 3,3 В) для високого рівня, 0 В для для низького рівня. Якщо контакт налаштовано як INPUT, digitalWrite () ввімкне або вимкне внутрішнє підтягування на вхідному контакті.

- **delay (ms)** — зупиняє виконання програми на проміжок часу ms у мілісекундах.
- **Serial.print (val)** — друкує дані (val) на COM-порт у вигляді тексту ASCII, який може читати людина. Байти надсилаються як один символ. Символи та рядки надсилаються як є. Починаючи з версії 1.0, послідовна передача є асинхронною; Serial.print () повернеться до того, як будуть передані будь-які символи.
- **Serial.Begin(speed)** — ініціює послідовне з'єднання і задає швидкість передачі даних (speed) в біт/с (бодах). Для зв'язку з Serial Monitor обміну даними з комп'ютером використовує обов’язково хоча б одне число з наступних: 300, 1200, 2400, 4800, 9600, 14400, 19200, 28800, 38400, 57600 або 115200 бод. При з'єднанні через порти входів-виходів можуть використовуватись інші значення швидкості, необхідні пристрою для здійснення обмін даними.

Класифікація функцій: цифрові та аналогові функції з входами-виходами, математичні та тригонометричні, логічні, бітові та функції випадковості тощо.

2.2.2 Бібліотеки

Можливості середовища Arduino можна розширити за допомогою бібліотек, як і більшість платформ програмування. Бібліотеки надають додаткову функціональність у роботі та збільшують можливості. Ряд бібліотек встановлюється разом з IDE, також є можливість створювати власні чи завантажувати імпортовані.

Бібліотека	Призначення
Servo	керування сервомоторами.
Stepper	управління кроковими двигунами.
SPI	для зв'язку з пристроями за допомогою шини SPI.
Wire	інтерфейс (TWI/I2C) для надсилання та отримання даних через мережу пристроїв або датчиків.
SoftwareSerial	налагодження послідовного зв'язку на цифрових контактах.
ArduinoBLE	використання Bluetooth® Low Energy на певних платах.
Ethernet	підключення до Інтернету через Ethernet.
GSM	підключення до мережі GSM/GRPS за допомогою GSM.
WiFi	підключення до Інтернету через Wi-Fi.
EEPROM	читання та запис у «постійне» сховище.
SD	читання та запису SD-карт.
LiquidCrystal	керування рідкокристалічними дисплеями.

Таблиця 6. Опис стандартних бібліотек середовища Arduino IDE

Розглянемо допоміжні бібліотеки, які були використанні у роботі за допомогою імпортування, а саме:

Бібліотека RemoteXY є частиною проєкту RemoteXY. Ця бібліотека дозволяє керувати пристроєм за допомогою мобільного пристрою з операційною системою Android або iOS. Для підключення можна використовувати Bluetooth, USB OTG, WiFi, Ethernet або Cloud. Щоб створити графічний інтерфейс, треба перейти на офіційний вебсайт. Ця бібліотека сумісна з усіма архітектурами, тому ви зможете використовувати її на всіх платах Arduino. Використання RemoteXY допомагає заощадити багато часу та ресурсів. Прикладом є розробка нашого застосунку для керування платформою, додавши розроблену конфігурацію до застосунку.

Бібліотека CyberLib дає вагомий приріст швидкості (запису-зчитування цифрових портів у 20 разів) та зменшує розмір використаної пам'яті. Синтаксис максимально простий та буде зрозумілий навіть новачку. CyberLib взаємодії з платами платформи Arduino, таких як: Arduino Uno, Arduino Nano, Arduino Leonardo, Arduino Mega тощо. Максимальна швидкодія досягається також зчитуванням аналогових входів, роботою з Timer1 та SPI, взаємодією Serial з відправленням масиву та запису-зчитування Serial.

2.3 Написання логіки роботи МК

Написання логіки роботи будь-якого мікроконтролера починається з постановки мети та уявлення реалізації виконаних дій. Метою розробки стає розробити рухому платформу із низкою особливостей, що буде переважати схожі аналоги. Візуалізуємо можливості макета розробленої платформи:

1. Пересування за допомогою людського керування або самостійного пересування елементарним аналізом.
2. Регулювання швидкості двигунів.
3. Зчитування відстані до об'єкта, аналіз та інфографіка отриманої інформації.
4. Індикація різноколірними світлодіодами залежно від відстані до об'єкта.
5. Дистанційне керування відбувається за допомогою розробленого мобільного застосунку.

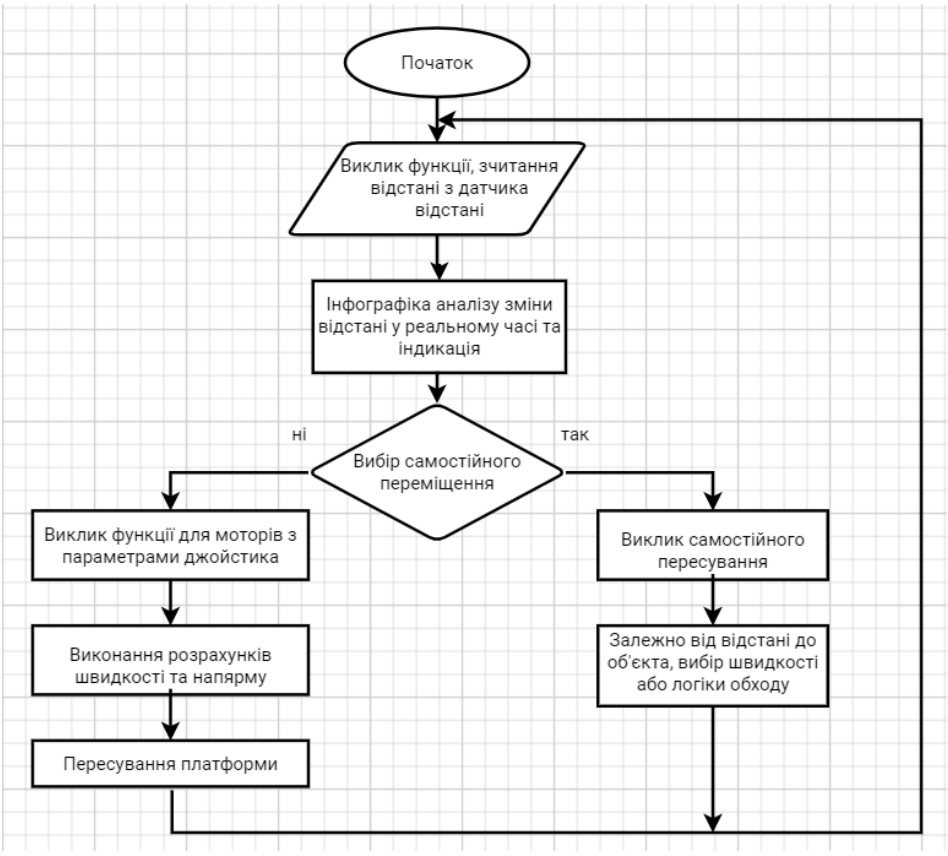

Рисунок 22. Блок схема основних кроки логіки роботи МК у платформі

Для розробки макета рухомої платформи за основу взято плату Arduino Uno побудовану на МК ATmega328. З проаналізованої документації підкреслено особливості роботи ШІМ виходів, що набагато спрощує розробку логіки роботи макета.

Основною ідеєю створення та розроблення макета є написання максимально оптимізованої та простої логіки взаємодії МК з допоміжними компонентами. Логіка роботи МК зачіпає також допоміжні елементи, а саме: двигуни (мотори) й драйвера керування, датчик розрахунку відстані, модуль зв'язку з МК та індикація світлодіодами.

По-перше, головною особливістю платформи є пересування. Напрямок та швидкість пересування регулюється застосунком. Метод SpeedMore містить отримання символьного масиву з відповідних змінних щодо моторів та значення, що використовується для визначення напряму та швидкості пересування. Принцип роботи джойстика та G-сенсора відповідає рисунку.

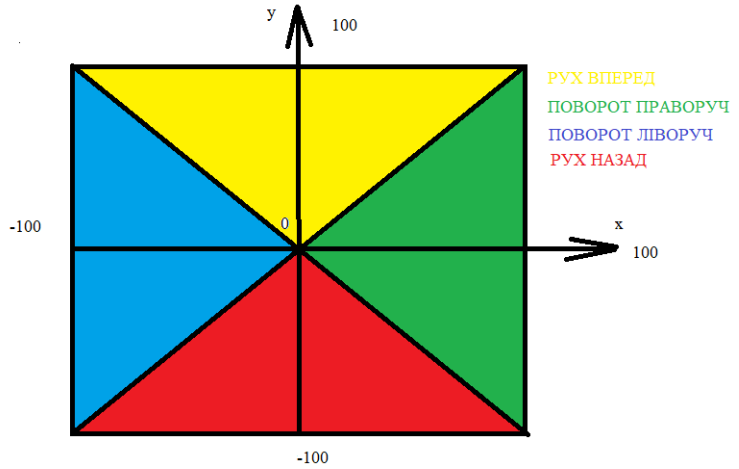

Рисунок 23. Логіка пересування платформи за допомогою джойстика

Відповідно логіки роботи при перебуванні джойстика у відповідній зоні розраховуються значення для кожного з моторів:

- для правих моторів: Y + X;
- для лівих моторів: Y – X.

Отримане значення відповідає напряму та ШІМ-сигналу швидкості, що подається на виходи МК. Драйвера мотора L298N отримує відповідні значення та платформа пересувається у заданому напрямку з вибраною швидкістю.

По-друге, для аналізу місцевості вирішено використати ультразвуковий датчик відстані HC-SR04. Призначений для вимірювання відстаней від 2 до 400 см з точністю до декількох мм. Розрахунок відстані для зручності виконується за використанням методу GetDistance з додатка А. Аналіз отриманої відстані використовується у самостійному режимі керування та інфографіки в застосунку. Максимальна точність наближення до реальних досягається збільшенням замірів та фільтрація результатів для запобігання помилок.

По-третє, написання індикація розробленого макета. Індикація прописана у методі Display, що розміщений в додатку А. Застерігати оточення про можливу заваду на шляху є доцільно. Принцип роботу схожий на роботу світлофора. Для роботи використовуємо світлодіоди різного кольору, а саме: червоного, жовтого та зеленого. Завжди горить лише один світлодіод, що відповідає певній умові. При відстані до об'єкта у сантиметрах:

- Більш як 100 — зелений світлодіод;
- в межах від 26 до 100 — жовтий світлодіод;
- менше ніж 26 — червоний світлодіод.

Підсумковим вважаю налаштування комунікації між МК та пристроєм керування. Для спілкування цих пристроїв доцільно використовувати модуль зв'язку HC-05, що використовує технологію бездротового зв'язку Bluetooth. Для прийняття та відправлення даних при зв’язуванні пристроїв проводиться відповідно через піни TX та RX мікроконтролера. Важливим є завантаження скетчу до макета, треба відключати від живлення модуль зв'язку за це відповідає VCC пін. Підключення схеми та розробка мобільного застосунку описано у наступних пунктах дипломної роботи.

2.3.1 Режим самостійного переміщення платформи

Реалізація самостійного пересування рухомої платформи також можна розробити за правильного використанням мікроконтролера та датчика відстані. Найбільш доцільно написати власну функцію, що буде виконувати переміщення з логічних міркувань. Розглянемо написаний метод SmartMode для самостійного пересування, що розміщений у додатку А.

Логіка метода полягає у прийнятті цілочисельного значення змінної LengthToCrimp, що дорівнює відстані до можливої перешкоди на шляху.

Далі на програмному рівні відбувається порівняння значень цієї змінної.

Якщо значення вище за 200, то платформа рухатиметься вперед зі 100% швидкістю, тому що перешкода досить далеко.

Якщо значення нижче ніж 200, то відбудеться така перевірка:

- значення вище за 50 — платформа рухатиметься вперед зі 75% швидкістю;

- значення нижче ніж 50 — відбувається перевірка, чи значення більше за 25, за такої умови платформа рухатиметься вперед зі 50% швидкістю.

У всіх інших варіантах платформа зупиняється та чекає одну секунду. Виконується остаточна перевірка методом SmartMode, що аналізує значення LengthToCrimp менше або дорівнює 10. Якщо умова виконується платформа протягом пів секунди рухається назад та зупиняється на секунду. За результатом виконується поворот платформи праворуч протягом пів секунди.

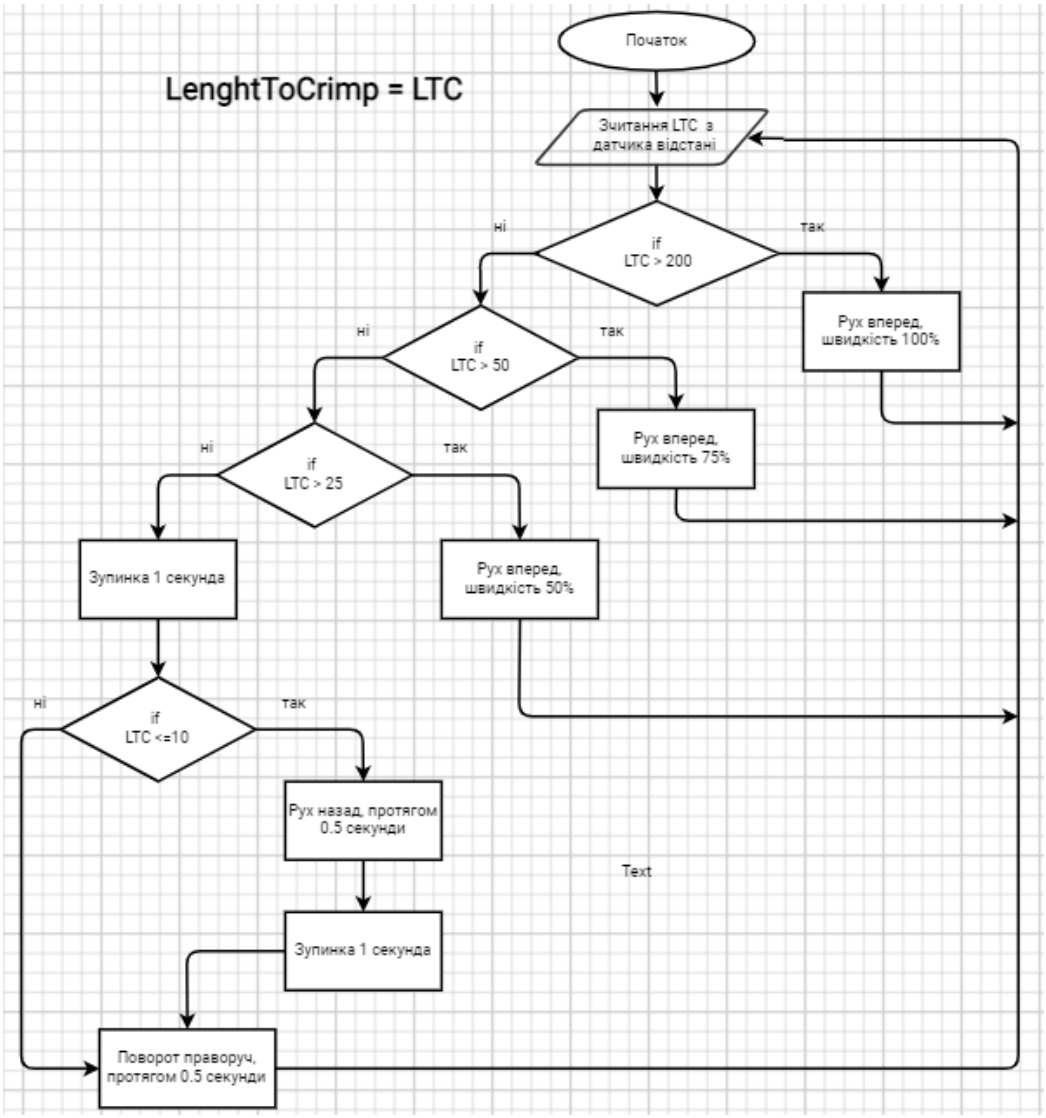

Рисунок 24. Блок схема логіки самостійного пересування платформи

Дії у методі виконуються послідовно. Виклик повторюється циклічно у реальному часі. Момент виходу з самостійного пересування платформи відбувається при зміні на ручне керування застосунку.

Самостійна логіка пересування супроводжується зміною яскравості зеленого світлодіода. Індикація залежить від відстані до об'єкта, значення ШІМ сигналу передається на вихід з від 0 до 255. Код розробленої логіки роботи МК наведено у додатку А.

2.4 Модель макета платформи розробки

Для складання макета платформи доцільно скористатися розробленою схемою, що розташована нижче.

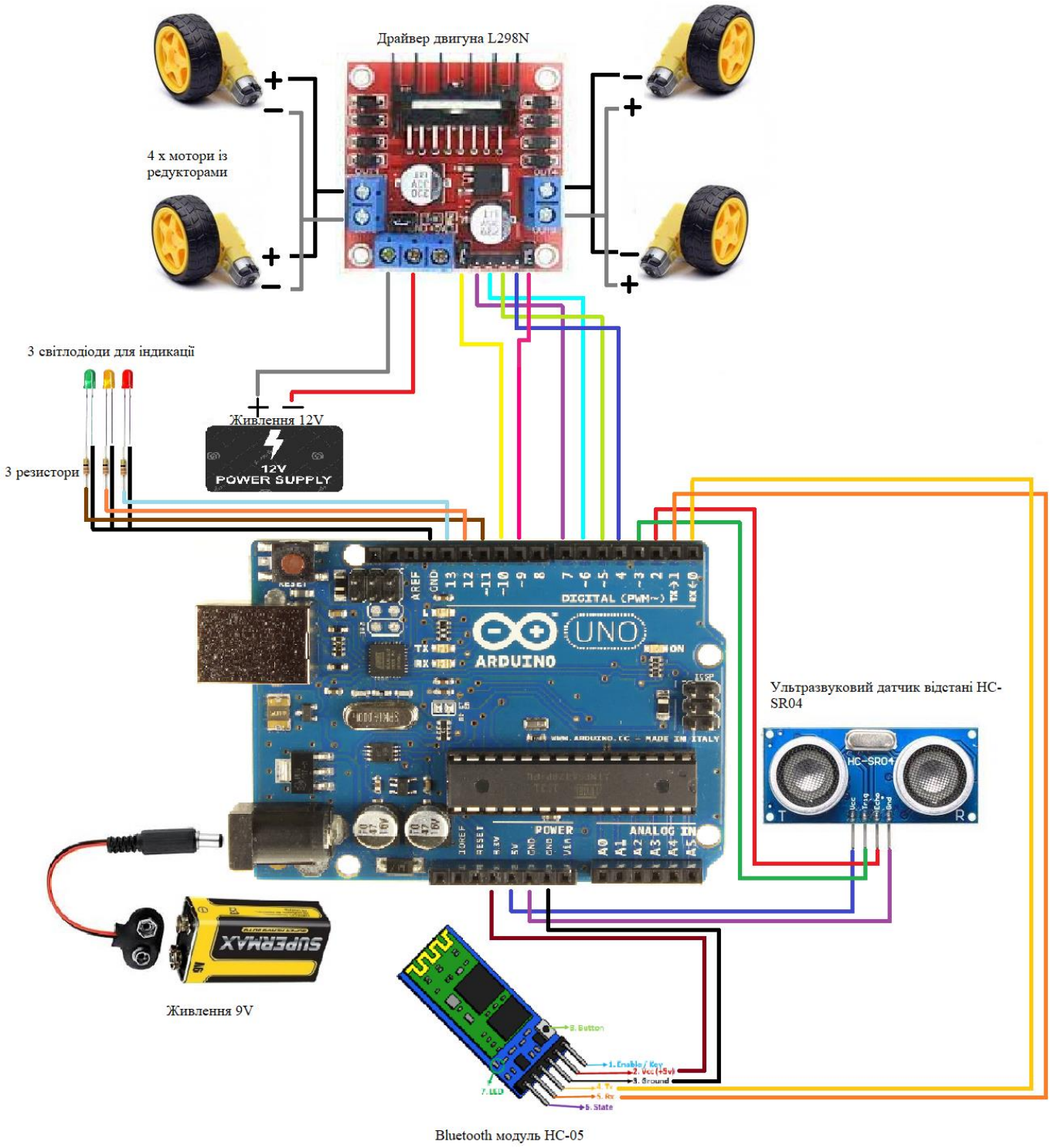

Рисунок 25. Проєктувальна схема макета

Модель макета рухомої платформи складається з таких елементів:

- Мікроконтролер Arduino Uno;
- Драйвер двигуна L298N;
- Модуль Bluetooth HC-05;
- Ультразвуковий датчик відстані HC-SR04;
- Портативний зарядний пристрій 5V або акумулятор від 7 до 12V;
- Джерело живлення 12V;
- Різноколірні світлодіоди (зелений, жовтий, червоний);
- Резистори (для захисту від виходу з ладу світлодіодів);
- Мотори з редукторами;
- Комутаційні проводи типу «мама-мама» та «тато-тато»;
- Корпус та кріплення.

Усі модулі, датчики та драйвери досить легко під’єднати треба лише рухатися по створеній схеми. У світлодіодах для запобігання швидкого виходу з ладу треба від'єднати до контакту анода (довший) резистор від 100 Ом до 200 Ом. Живлення до Arduino Uno можна подавати через USB, акумулятор чи батарею. Остаточним є під'єднання живлення до драйвера моторів та самих коліс з редукторами. Для компактності усі елементи розмістити максимально раціонально.

2.5 Розробка застосунку керування за допомогою сервісу RemoteXY

2.5.1 Сервіс дистанційного керування RemoteXY

Розвиток технологій керування стрімко набирає оберти, а з ними й усі тісно пов'язані галузі. Ще декілька десятиліть тому людство не уявляло життя в колі великої кількості гаджетів. Керувати пристроями почали спочатку завдяки механічній взаємодії. І наразі люди поступово розвивають ці технології. Широкого застосування набули способи керування пристроями, такі як: термінал, інфрачервоний порт, Bluetooth, Internet та подібні.

За основу, для взаємодії зі своїм проєктом, взято бездротову технологію зв'язку Bluetooth, що широко використовується з початку 21 століття, а зараз у більшості smart-пристроях. Для створення графічного інтерфейсу вибрано систему розробки RemoteXY. Особливості саме цієї системи є розробка та використання мобільних графічних інтерфейсів для керування мікроконтролерами, наприклад, віддалено зі смартфона чи планшета. До складу системи RemoteXY входить:

1) Редактор мобільних графічних інтерфейсів для контролерів, розміщений на вебсайті за посиланням: https://remotexy.com./

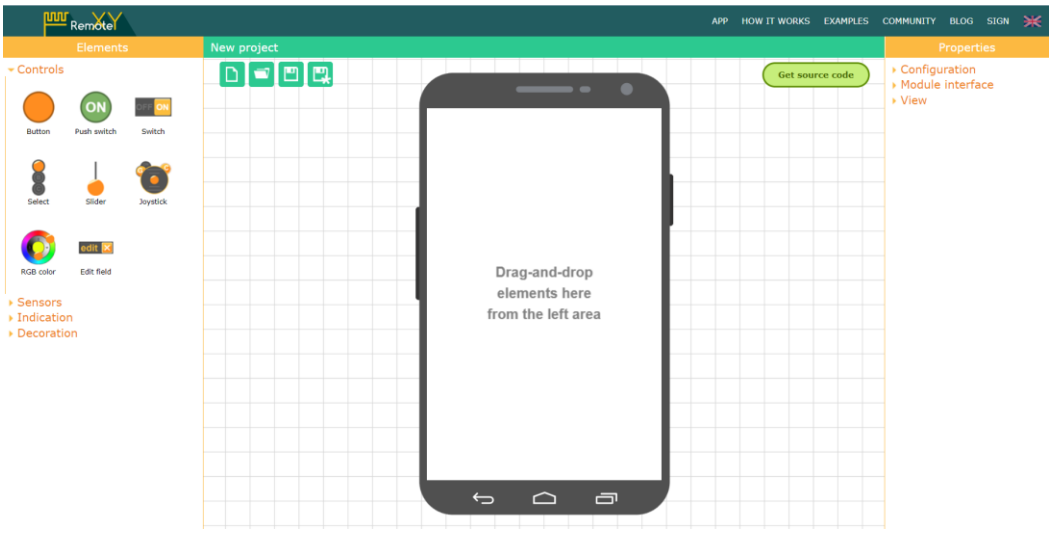

Рисунок 26. Функціонал онлайн редактора RemoteXY

2) Мобільний додаток RemoteXY, що дозволяє підключитися до МК та зображати графічні інтерфейси. Завантажити застосунок можна за посиланням: http://remotexy.com/ru/download/.

Відмінні особливості сервісу:

- Конфігурація графічного інтерфейсу зберігається у контролері. При підключенні немає жодної взаємодії зі сторонніми серверами для того, щоб завантажити графічно інтерфейс. Конфігурація графічного інтерфейсу завантажується на мобільний додаток з контролера.
- З одного мобільного додатка ви можете керувати всіма своїми пристроями. Кількість пристроїв не обмежена.

Підтримуються наступні способи зв'язку між мікроконтролером та мобільним пристроєм:

- Інтернет із будь-якого місця через хмарний сервер;
- Wi-Fi в режимі клієнта та точки доступу;
- Bluetooth;
- Ethernet за IP-адресою або URL;
- USB OTG (доступно лише для Android із підтримкою USB OTG).

Генератор вихідного коду інтерфейсу підтримує такі контролери:

- Arduino UNO, MEGA, Leonardo, Pro Mini, Nano, MICRO та сумісні AVR МК;
- контролери на ESP8266 та ESP32;
- ChipKIT UNO32, ChipKIT uC32, ChipKIT Max32.

Підтримка інтегрованих середовищ розробки: Arduino IDE, FLProg IDE та MPIDE.

Мультиплатформність мобільних операційних систем, а саме: Android та iOS.

2.5.2 Застосунок для керування платформою

Чудовою можливість розробити графічний інтерфейс управління є сервіс RemoteXY, що використовуючи елементи управління, індикації та оформлення у будь-якій комбінації сформує доцільний застосунок. Розміщуючи елементи на екрані за допомогою онлайн-редактора, можна розробити інтерфейс під будь-яке завдання. Отримавши вихідний код програми для мікроконтролера, надається структуру для взаємодії вашої програми з елементами керування та індикації. За допомогою однієї мобільної програми можна керувати великою гамою пристроїв із різними графічними інтерфейсами керування. Основною частиною з вихідного коду є конфігурація інтерфейсу за допомогою якої й відбувається формування візуального оформлення на контролері та відображення у застосунку.

Основні кроки для взаємодії мікроконтролера та застосунку:

1) Розроблення інтерфейсу застосунку на сайті RemoteXY та вивантаження вихідного коду за допомогою відповідної кнопки.
2) Завантаження вихідного коду із конфігурацією до МК.
3) Підключення модулів зв'язку.
4) Встановлення мобільного застосунку для керування.
5) Створення з'єднання увімкненої платформи та застосунку.

Збалансований функціонал застосунку — можливість ефективно взаємодіяти з пристроєм та доцільно витрачати час на досягнення цілей. У ході розробки нашого мобільного інтерфейсу використовувались такі елементи: онлайн графік для аналітики зміни відстані, перемикач для зміни режиму роботи, написи для більшого розуміння та багатофункціональний джойстик із можливістю керування за допомогою G-sensor вбудованого у мобільному гаджеті. Зручний та витриманий дизайн розроблений для обох орієнтацій, а саме: вертикальній та горизонтальній.

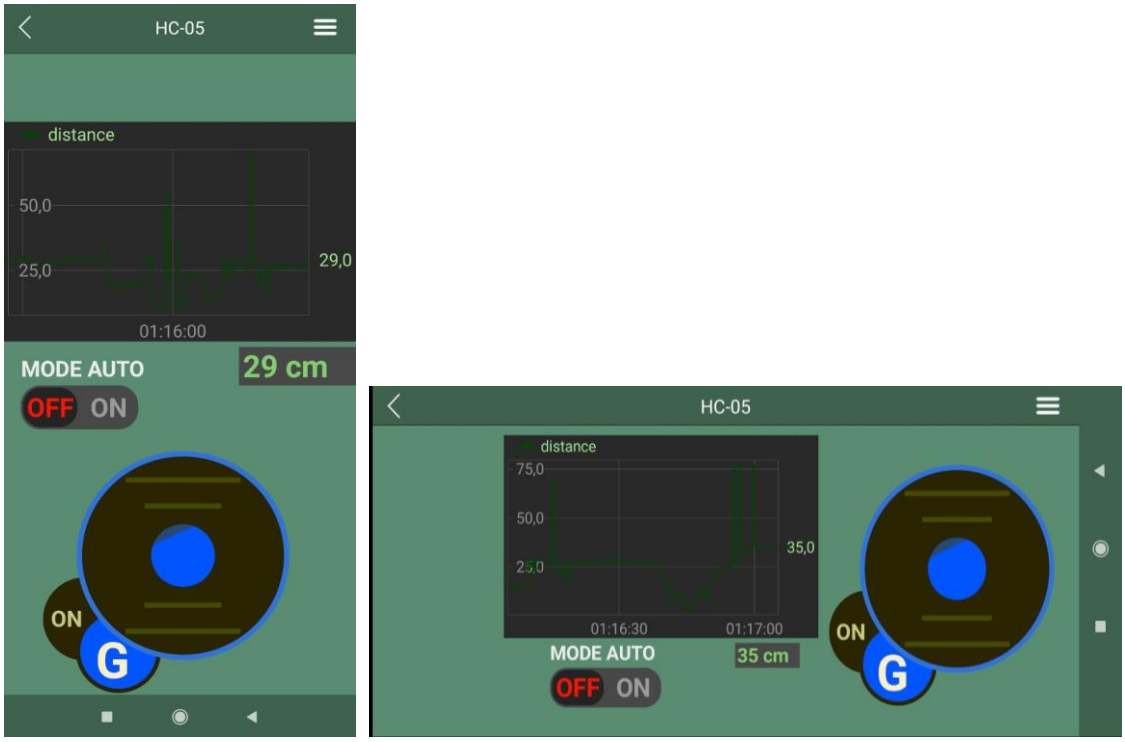

Рисунок 27. Приклад візуального оформлення застосунку у вертикальній та горизонтальній орієнтації

Використання розробленого застосунку передбачає у собі:

a) Аналіз відстані між предметами за допомогою графіка та зміни в ньому часових затримок, що змінюються у реальному часі. За допомогою приближення та віддалення відбувається масштабування змінних по абсцисі та ординаті графіку.
b) Поточне значення відстані між предметами зображується не лише на графіку, а також у написі під ним у сантиметрах.
c) Перемикач відповідає за зміну режиму роботи макета, при увімкненні макет пересувається самостійно, а при вимкненні діє звичайний режим керування макетом.
d) Джойстик має два варіанти управління: перший — звичайного джойстика, другий — керування за допомогою G-сенсору (акселерометра).

2.6 Результати роботи

У ході виконання даної роботи була розроблена програма, що виконувалась на мікроконтролері за написаною логікою роботи. Створений мобільний додаток для керування макетом розробленої платформи. Зібрано макет дипломної роботи спираючись на схемотехнічне рішення.

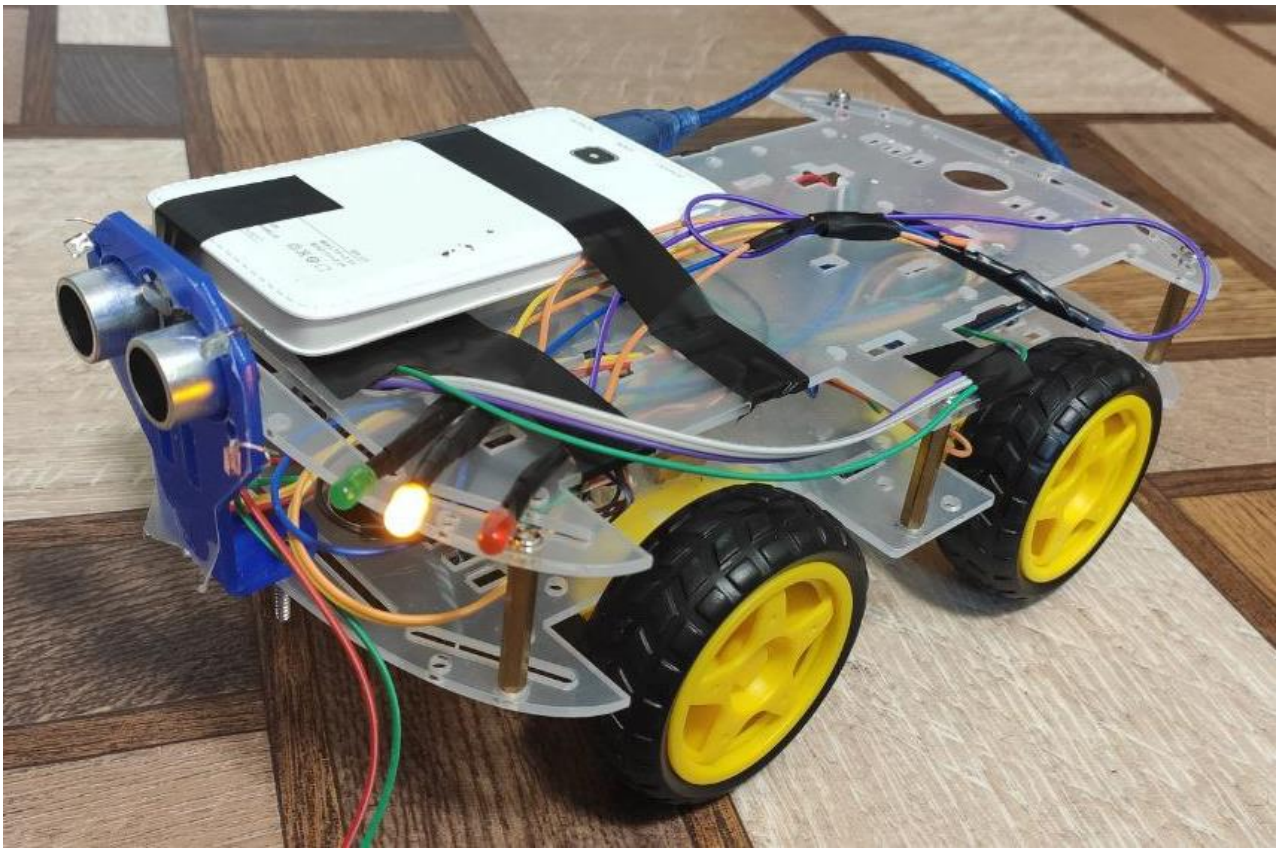

Рисунок 28. Макет рухомої платформи з дистанційним керуванням

У побудованому макеті є деякі недоліки, що можна легко виправити за бажанням:

- Для повної автономності платформи бажано змінити живлення моторів. Замінити живлення від мережі на живлення від акумулятора 12 V.
- Датчик відстані може інколи не помічати маленькі перешкоди на шляху. Додати ще один датчик та паралельно аналізувати дані з них.

Висновки

Платформа Arduino є гнучкою системою, що дозволяє розробляти складні системи, які пов'язують логічну взаємодію роботу елементів між собою. Завдяки мікроконтролеру плати Arduino Uno, який взято за основу проєкту, було розроблено демонстраційний макет рухомої платформи розробки з низкою можливостей. Однією з них є дистанційне керування за допомогою застосунку.

Інтегроване середовище розробки Arduino IDE продемонструвало повний спектр можливостей при розробці великого проєкту. Простий синтаксис та можливість імпортувати допоміжні бібліотеки допомогли з легкістю написати логіку роботи рухомої платформи з модифікаціями для полегшеного користування.

Під'єднані до плати Arduino Uno допоміжні датчики та інші складові запобігли можливим проблемам під час розробки та відіграли важливу роль у створенні макета. Для подальшої модифікації платформи існує можливість приєднувати додаткові датчики або драйвери, що розкриють ще більший потенціал платформи.

Крім цього, написано та реалізовано простий алгоритм інструкцій для рухомої платформи у випадку вибору самостійного режиму пересування, що не потребує додаткового керування. Режим можна обрати у розробленому застосунку, що також має додатковий функціонал, а саме: керування платформою у двох режимах, аналіз місцевості та індикацію.

Розроблений демонстраційний макет може бути використаний у якості демонстрації наочного прикладу дистанційного керування платформою розробки в курсах «Периферійні пристрої», «Мікропроцесорна техніка», «Схемотехніка» та «Програмування вбудованих систем».

Використана література

1. Схемотехніка електронних систем: У 3 кн. Кн. 3. Мікро-процесори та мікро-контролери: Підручник / В. І. Бойко, А. М. Гуржій, В.Я. Жуйков та ін. – 2-ге вид., допов. і переробл. – Київ: Вища шк., 2004. 399 с.: іл.
2. Чешко І. В. Вступ до спеціальності «Електроніка»: навчальний посібник — Суми: Сумський державний університет, 2017. 148 с.
3. «Проектування радіоелектронних схем» для студентів радіофізичного факультету / Пархоменко Д. А., Смирнов Є. М. – Київ: Радіофізичний факультет Київського національного університету імені Тараса Шевченка, 2013. 74 с.
4. «Автоматизація та комп'ютерно-інтегровані технології» / О.М. Павловський; КПІ ім. Ігоря Сікорського, 2021. 104 с.
5. Мікропроцесорна техніка: Електронний підручник / В.Я. Жуйков, Ж59 Т.О. Терещенко, Ю.С. Ямненко, А.В.Заграничний ; відп. ред. О.В. Борисов. 2016. 440 с.
6. Електронний магазин радіодеталей URL: https://radiostore.com.ua/.
7. Евстифеев А.В. Микроконтроллеры AVR семейств Tiny и Mega фирмы ATMEL, 5е изд., стер. — Москва: Издательский дом «ДодэкаXXI», 2008. 560 с.
8. Основы микропроцессорной техники: учебное пособие / С. Н. Ливенцов, А. Д. Вильнин, А. Г. Горюнов. – Томск: Изд-во Томского политехнического университета, 2007. 118 с.
9. AVR® MCU Peripherals Quick Reference Card URL: https://www.microchip.com/en-us/products/microcontrollers-and-microprocessors/8-bit-mcus/.
10. Remote control Arduino – RemoteXY URL: https://remotexy.com/en/.
11. Офіційний сайт Arduino URL: https://www.arduino.cc/.

Додаток А

Програмний код, що виконують на мікроконтролері Arduino Uno, для реалізації логіки роботи та створення інтерфейсу застосунку:

```
#define REMOTEXY_MODE__HARDSERIAL

#include <RemoteXY.h> // бібліотека для створення візуала застосунку та підключення допоміжних можливостей

#include <CyberLib.h> // бібліотека для пришвидшення дій

#define REMOTEXY_SERIAL Serial

#define REMOTEXY_SERIAL_SPEED 9600 // символьна швидкість для передачі інформації через послідовний порт

#pragma pack(push, 1)

#define size_buff 5 // розмір масиву для зберігання певної кількості замірів

#define range 1 // поріг допустимих значень

uint16_t sensor[size_buff]; // масив

uint8_t RemoteXY_CONF[] = // 99 bytes

  { 255,3,0,20,0,92,0,16,157,2,68,49,258,2,68,41,0,4,63,40,

  8,135,100,105,115,116,97,110,99,101,0,67,4,52,44,14,5,42,45,21,

  7,120,26,16,5,47,79,8,42,42,13,64,38,38,190,94,94,2,1,268,

  49,24,8,3,52,21,8,16,26,135,36,79,78,0,79,70,70,0,129,0,

  268,44,24,4,3,47,25,4,8,77,79,68,69,32,65,85,84,79,0 }; // конфігураційний файл у вигляді байтів інформації

#define PIN_MOTOR_RIGHT_UP 7 // один з двох виходів для надання напрямку правих моторів

#define PIN_MOTOR_RIGHT_DOWN 6 // один з двох виходів для надання напрямку правих моторів
```

```
#define PIN_MOTOR_RIGHT_SPEED 10 // вихід для регулювання ШІМ сигналу для швидкості правих моторів

#define PIN_MOTOR_LEFT_UP 5 // один з двох виходів для надання напрямку лівих моторів

#define PIN_MOTOR_LEFT_DOWN 4 // один з двох виходів для надання напрямку лівих моторів

#define PIN_MOTOR_LEFT_SPEED 9 // вихід для регулювання ШІМ сигналу для швидкості лівих моторів

long duration, cm; // змінні для визначення часу проходження echo та відстані до об'єкта

unsigned char RMotor[3] =

  {PIN_MOTOR_RIGHT_UP, PIN_MOTOR_RIGHT_DOWN, PIN_MOTOR_RIGHT_SPEED}; // праві мотори із зазначеними параметрами

  unsigned char LMotor[3] =

  {PIN_MOTOR_LEFT_UP, PIN_MOTOR_LEFT_DOWN, PIN_MOTOR_LEFT_SPEED}; // ліві мотори із зазначеними параметрами

struct {

  int8_t joystick_1_x; // =-100..100 x-coordinate joystick position

  int8_t joystick_1_y; // =-100..100 y-coordinate joystick position

  uint8_t switch_1; // перемикач зміни режиму керування

 float onlineGraph_1; // графік зображення зміни відстані до об'єкта

  char text_1[16]; // змінна для доцільного зображення відстані до об'єкта

  uint8_t connect_flag;

} RemoteXY;

#pragma pack(pop)


void setup()
```

```
{
RemoteXY_Init ();
D3_Out; D3_Low; // пін trig ультразвукового сонара
D2_In; // пін echo ультразвукового сонара
Serial.begin(REMOTEXY_SERIAL_SPEED); // ініціалізація послідовного з'єднання
pinMode (PIN_MOTOR_RIGHT_UP, OUTPUT); // задання виходів для керування моторами
pinMode (PIN_MOTOR_RIGHT_DOWN, OUTPUT);
pinMode (PIN_MOTOR_LEFT_UP, OUTPUT);
pinMode (PIN_MOTOR_LEFT_DOWN, OUTPUT);
pinMode(12, OUTPUT); // 12 вихід – жовтий світлодіод
pinMode(13, OUTPUT); // 13 вихід – червоний світлодіод
}
void loop() // основний цикл виконання МК
{
  RemoteXY_Handler ();
uint16_t cm = Display(); // виклик функції індикації з поверненням відстані до об'єкта
  if(RemoteXY.switch_1) // при увімкненні перемикача на ON
  {
    SmartMode(cm); // режим самостійного пересування
  }
  else // при значенні перемикача на OFF
  {
```

```
  SpeedMore (RMotor, RemoteXY.joystick_1_y + RemoteXY.joystick_1_x); // задання напряму та швидкості правих моторів
  SpeedMore (LMotor, RemoteXY.joystick_1_y - RemoteXY.joystick_1_x); // задання напряму та швидкості лівих моторів
  }
}
void SpeedMore (unsigned char * motomode, int vect) // метод налаштування швидкості та напряму мотора
{
  if (vect>100) vect=100;
  if (vect<-100) vect=-100;
  if (vect>0)
  {
    digitalWrite(motomode[0], HIGH);
    digitalWrite(motomode[1], LOW);
    analogWrite(motomode[2], vect*2.55);
  }
  else if (vect<0)
  {
    digitalWrite(motomode[0], LOW);
    digitalWrite(motomode[1], HIGH);
    analogWrite(motomode[2], (-vect)*2.55);
  }
```

```
  else
  {
    digitalWrite(motomode[0], LOW);
    digitalWrite(motomode[1], LOW);
    analogWrite(motomode[2], 0);
  }
}
 uint16_t GetDistance() // метод для розрахунку ультразвукового далекоміра
{
  D3_High; delay_us(10); D3_Low; // початок замірювання
   uint16_t duration = pulseIn(2,HIGH); // тривалості часу проходження echo
   delay_ms((50000-duration)/1000); // затримка між відправленнями
  return duration/58; // переведення в см
}
uint16_t Display() // метод індикації відстані до об'єкта
{
  for (uint8_t i = 0; i < size_buff; ++i) { sensor[i]=GetDistance();} // проведення size_buff кількості вимірів
    uint16_t cm = find_similar(sensor, size_buff, range); // фільтрація вимірів
     // інфографіка дальності до об'єкта у застосунку
  Serial.print("Відстань до об'єкту: "); // інфографіка дальності до об'єкта
  Serial.print(cm);
  Serial.println(" см.");
```

```
  RemoteXY.onlineGraph_1 = cm;
  sprintf (RemoteXY.text_1, "%d cm", cm);
  if(RemoteXY.switch_1) // при положенні перемикача на ON
  {
    analogWrite(11,cm); // ШІМ регулювання яскравість зеленого світлодіода, що залежить від дальності до об'єкта
    digitalWrite(12, LOW); // вимкнення жовтого світлодіода
    digitalWrite(13, LOW); // вимкнення червоного світлодіода
  }
  else // при положенні перемикача на OFF
  {
  if (cm > 100 && cm<1000) // при відстані до об'єкта від 101 до 999
  {
    analogWrite(11, 255); // максимальна яскравість зеленого світлодіода за допомогою ШІМ сигналу
    Serial.println(" green on ");
  }
  else
  {
    analogWrite(11, 0); // вимкнення зеленого світлодіода
    }
  if (cm <= 100 && cm > 25) // при відстані до об'єкта від 26 до 100
  {
    digitalWrite(12, HIGH); // увімкнення жовтого світлодіода
```

```
    Serial.println(" yellow on ");
    }
  else
  {
    digitalWrite(12, LOW); //вимкнення жовтого світлодіода
    }
  if (cm <= 25 || cm >1000) // при відстані менше ніж 26 та більше як 1000
  {
    digitalWrite(13, HIGH); // увімкнення червоного світлодіода
    Serial.println(" red on ");
    }
  else
  {
    digitalWrite(13, LOW); // вимкнення червоного світлодіода
    }
  }
 return cm; // повернення відстані для подальшого використання
}
void SmartMode(int LengthToCrimp) // розумний режим пересування
{
if(LengthToCrimp>200) // при відстані до об'єкта більшою за 200 см
{
```

```
 SpeedMore (RMotor, 100); // рух вперед зі 100% швидкістю
 SpeedMore (LMotor, 100);
}
else if(LengthToCrimp>50) // відстань більша як 50 см та менше ніж 200 см
  {
    SpeedMore (RMotor, 75); // рух вперед зі 75% швидкістю
    SpeedMore (LMotor, 75);
  }
else if(LengthToCrimp>25) // при відстані до об'єкта більшою за 25 см та менше ніж 50 см
   {
    SpeedMore (RMotor, 50); // рух вперед зі 50% швидкості
    SpeedMore (LMotor, 50);
    }
else
    {
    SpeedMore (RMotor, 0); // зупинка
    SpeedMore (LMotor, 0);
        delay(1000);
     if(LengthToCrimp<=10) // при відстані до об'єкта меншою чи рівною 10 см
        {
          SpeedMore (RMotor, -100); // рух назад
          SpeedMore (LMotor, -100);
```

```
        delay(500);
        SpeedMore (RMotor, 0); // зупинка
        SpeedMore (LMotor, 0);
        delay(1000); // затримка 1 сек
      }
   SpeedMore (RMotor, -100); // поворот праворуч
   SpeedMore (LMotor, 100);
   delay(500); // затримка 0,5 сек
   }
}
```