

Міністерство освіти і науки України  
Державний заклад  
«Луганський національний університет імені Тараса Шевченка»

Навчально-науковий інститут математики та інформаційних технологій

Кафедра інформаційних технологій та систем

**Тютюнник Андрій Володимирович**

**ПРОЄКТУВАННЯ ТА РОЗРОБКА СИСТЕМИ ДЛЯ  
КОНТРОЛЬОВАНОГО НАГРІВАННЯ ВОДИ НА БАЗІ ARDUINO ЯК  
ЕЛЕМЕНТ ІНТЕРНЕТУ РЕЧЕЙ**

**кваліфікаційна робота**

**здобувача вищої освіти другого (магістерського) рівня**

**освітньої програми «Комп'ютерні мережі»**

**за спеціальністю 123 Комп'ютерна інженерія**

Особистий підпис \_\_\_\_\_ Андрій Тютюнник

Науковий керівник \_\_\_\_\_ Володимир ДОНЧЕНКО,  
старший викладач кафедри  
інформаційних технологій та систем

Завідувач кафедри \_\_\_\_\_ Микола СЕМЕНОВ,  
кандидат педагогічних наук, доцент  
кафедри інформаційних технологій  
та систем

Полтава – 2025

## **АНОТАЦІЯ**

**Тютюнник А. В.**

**Тема:** Проєктування та розробка системи для контрольованого нагрівання води на базі Arduino як елемент інтернету речей.

**Спеціальність:** 123 «Комп'ютерна інженерія».

**Установа:** ЛНУ імені Тараса Шевченка, 2025 р.

**Магістерська робота містить:** 107 с., 81 рис., 1 табл., 1 додат., 51 джерело.

**Об'єкт дослідження** – використання засобів мікропроцесорної техніки для створення пристрою контрольованого нагрівання води.

**Предмет дослідження** – проєктування та розробка системи для контрольованого нагрівання води на базі платформи Arduino.

**Мета дослідження** - проєктування та розробка системи для контрольованого нагрівання води на базі платформи Arduino як елемента IoT.

**Результати роботи.** У результаті проведених досліджень були опрацьовані існуючі способи дослідження та моделювання систем для контрольованого нагрівання води. Створено систему для контрольованого нагрівання води. Розроблено програмне забезпечення для системи нагрівання води та керування його налаштувань. Побудовано макет системи та налаштовані усі електричні компоненти. Розроблено програмне забезпечення для керування макетом зі смартфона.

**Ключові слова.** ВИМІРЮВАННЯ РІВНЯ ВОДИ, ВИМІРЮВАННЯ ТЕМПЕРАТУРИ, МІКРОКОНТРОЛЕР, МІКРОПРОЦЕСОРНА ТЕХНІКА, APP INVENTOR 2, ARDUINO, BLUETOOTH.

## ABSTRACT

**Tiutiunnyk Andrii**

**Theme:** Design and development of a system for controlled water heating based on Arduino as an element of the Internet of Things.

**Speciality:** 123 "Computer Engineering"

**Institution:** Luhansk Taras Shevchenko National University (LTSNU), 2025.

**Master's work of:** 107 pages, 81 Fig., 1 Table, 1 adj., 51 source.

**A research object** is using microprocessor technology to create a controlled water heating device.

**The article of research** is design and development of a system for controlled water heating based on the Arduino platform.

**An aim of work** is design and development of a system for controlled water heating based on the Arduino platform as an IoT element.

**Job performances.** As a result of the research, the existing methods of research and modeling of systems for controlled water heating were developed. A system for controlled water heating has been created. Software for water heating system and control of its settings has been developed. The system layout is built and all electrical components are configured. Software for layout management from a smartphone has been developed.

**Keywords.** WATER LEVEL MEASUREMENT, TEMPERATURE MEASUREMENT, MICROCONTROLLER, MICROPROCESSOR EQUIPMENT, APP INVENTOR 2, ARDUINO, BLUETOOTH.

## ЗМІСТ

<b>ВСТУП.....</b>	<b>6</b>
<b>РОЗДІЛ 1. АНАЛІТИЧНА ЧАСТИНА .....</b>	<b>8</b>
1.1. Розвиток мікропроцесорної техніки.....	8
1.2. Огляд мікроконтролерів .....	10
1.2.1. Arduino Uno .....	11
1.2.2. Arduino Mega 2560.....	13
1.3. Огляд периферійних пристроїв.....	15
1.3.1. Пристрої вводу інформації.....	16
1.3.1.1. Потенціометр.....	16
1.3.1.2. Фоторезистор.....	18
1.3.1.3. Датчик температури DS18B20.....	22
1.3.1.4. Датчик тиску MPXV5050 .....	26
1.3.1.5. Інфрачервоний модуль .....	28
1.3.2. Пристрої виводу інформації.....	32
1.3.2.1. Реле для комутації .....	32
1.3.2.2. Драйвер двигуна.....	38
1.3.2.3. Сервопривід.....	44
1.3.2.4. Водяна помпа .....	46
1.3.2.5. RGB – світлодіод.....	46
1.3.2.6. Модуль звуку.....	49
1.3.2.7. Дисплей LCD 1602.....	53
1.3.3. Пристрої вводу та виводу інформації - модуль Bluetooth.....	57
Висновки до розділу.....	59
<b>РОЗДІЛ 2. ТЕОРЕТИЧНІ ОСНОВИ ТА ПРАКТИЧНІ АСПЕКТИ РОЗРОБКИ ПРИСТРОЮ.....</b>	<b>60</b>
2.1. Використання бібліотек для розширення функціональності програмного забезпечення.....	60
2.1.1. LiquidCrystal_I2C.h.....	61
2.1.2. OneWire.h .....	62
2.1.3. ServoSmooth.h .....	64
2.1.4. GyverButton.h .....	65

2.2. Види інтерфейсів для взаємодії периферійних пристроїв .....	66
2.2.1. SPI (Serial Peripheral Interface) .....	67
2.2.2. I2C (Inter-Integrated Circuit) або двопровідний інтерфейс .....	69
2.2.3. UART .....	71
2.3. Вдосконалення мікроконтролера Arduino Uno .....	72
2.3.1. Брязкіт контактів - програмне і апаратне усунення .....	73
2.3.2. Підключення декілька кнопок до одного аналогового входу .....	76
2.3.3. Збільшення аналогових входів завдяки мультиплексору CD4051 .....	85
2.3.4. Розширення кількості цифрових виходів за допомогою зсувного реєстра 74HC595 .....	88
Висновки до розділу 2 .....	93
<b>РОЗДІЛ 3. РЕАЛІЗАЦІЯ ПРИСТРОЮ ДЛЯ КОНТРОЛЬОВАНОГО НАГРІВАННЯ ВОДИ, ЯК ЕЛЕМЕНТ ІНТЕРНЕТУ РЕЧЕЙ.....</b>	<b>94</b>
3.1. Конструювання пристрою контрольованого нагрівання води .....	94
3.1.1. Схема електричних з'єднань пристрою .....	95
3.1.2. Прототипування на макетній платі .....	95
3.1.3. Макет для корпусу пристрою .....	97
3.1.4. Приклад програмної реалізації .....	97
3.1.5. Принцип функціонування пристрою для контрольованого нагрівання води .....	97
3.2. Розробка мобільного додатку для управління системою контрольованого нагрівання води .....	99
3.2.1. Середовище розробки MIT App Inventor 2 .....	99
3.2.2. Реалізація програмного забезпечення для мобільного додатку .....	99
3.2.3. Принцип дії програми .....	101
Висновки до розділу .....	102
<b>ВИСНОВКИ .....</b>	<b>103</b>
<b>СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ .....</b>	<b>104</b>
<b>ДОДАТОК А .....</b>	<b>108</b>

## ВСТУП

Розвиток технологій автоматизації та інтернету речей (IoT) сприяє створенню нових рішень для оптимізації та підвищення ефективності процесів у повсякденному житті, промисловості та науці. Однією з найперспективніших сфер застосування таких технологій є розробка систем, що забезпечують точний контроль та управління фізичними параметрами середовища. Серед таких параметрів особливу увагу займає температура, контроль якої є критично важливим для багатьох галузей: від побутових пристроїв до складних промислових установок.

Системи для контрольованого нагрівання води мають широкий спектр застосувань: у побуті для підтримки комфортної температури в «розумних» домах, у медичних лабораторіях для проведення досліджень, у харчовій промисловості для приготування продуктів тощо. Одним із сучасних підходів до створення таких систем є використання IoT-технологій, що дозволяють забезпечити гнучке дистанційне керування, автоматизацію процесів та інтеграцію з іншими компонентами розумного середовища.

Основою розробки таких рішень є мікроконтролери, зокрема платформа Arduino, яка зарекомендувала себе як зручний і доступний інструмент для створення прототипів та готових систем. Arduino надає розробникам можливість швидкого підключення сенсорів, виконання програмного забезпечення та інтеграції з іншими пристроями через різноманітні протоколи зв'язку.

**Об'єкт дослідження** – використання засобів мікропроцесорної техніки для створення пристрою контрольованого нагрівання води.

**Предмет дослідження** – проектування та розробка системи для контрольованого нагрівання води на базі платформи Arduino.

**Мета дослідження** - проектування та розробка системи для контрольованого нагрівання води на базі платформи Arduino як елемента IoT.

**Для досягнення поставленої мети необхідно виконати такі завдання:**

- Опрацювати існуючі способи дослідження та моделювання систем для контрольованого нагрівання води.

- Створити систему для контрольованого нагрівання води.
- Розробити програмне забезпечення для системи нагрівання води та керування його налаштувань.
- Побудувати макет системи та налаштувати усі електричні компоненти.
- Розробити програмне забезпечення для керування макетом зі смартфона.

**Методи дослідження** – основи мікроелектроніки, цифрова схемотехніка, проведення експерименту.

**Наукова новизна отриманих результатів:** запропоновано удосконалення технології, що забезпечують вимірювання рівня води та контроль температури у побутових пристроях нагрівання води, які побудовані на використанні мікропроцесорної техніки.

**Практична цінність отриманих результатів:** реалізовано систему, яка працює на базі мікроконтролера Arduino Uno. Мікроконтролер вдосконалено для підключення більшої кількості периферійних приладів та елементів комутацій сигналу, які потребує система для задуманої роботи пристрою. Система повністю автоматизована і може бути керованою, як органами управління, так і смартфоном на відстані.

## **РОЗДІЛ 1. АНАЛІТИЧНА ЧАСТИНА**

Мікропроцесор – це високоінтегрована цифрова схема, призначена для виконання обчислювальних операцій над двійковими даними [1]. Схема отримує на вхід послідовність двійкових кодів, інтерпретує їх як команди та виконує відповідні дії, результати яких подаються на вихід [2]. Архітектура мікропроцесора включає як комбінаційні, так і послідовні логічні елементи, забезпечуючи виконання як простих арифметичних операцій, так і складних алгоритмів.

Інтеграція всього процесора на одному мікрочіпі суттєво знизила вартість виробництва та підвищила надійність завдяки мінімізації кількості з'єднань. Високий рівень автоматизації виробничих процесів дозволяє виготовляти великі партії мікропроцесорів за відносно низькою ціною. Зменшення розмірів транзисторів на мікрочіпі при збереженні площі кристала призвело до постійного зростання обчислювальної потужності мікропроцесорів при незмінній вартості виробництва.

Історично, до появи мікропроцесорів використовувалися комп'ютери, побудовані на дискретних логічних елементах. Згодом їх замінили мікроконтролери, а потім і більш потужні мікропроцесори. Сьогодні мікропроцесори знайшли широке застосування в різних галузях, від вбудованих систем і портативних пристроїв до суперкомп'ютерів. Їх універсальність та висока продуктивність зумовили практичну відсутність альтернатив у сфері обчислювальної техніки.

### **1.1. Розвиток мікропроцесорної техніки**

До появи мікропроцесорів обчислювальні машини використовували громіздкі та енергоємні вакуумні трубки та транзистори. Компанія IBM, як один з лідерів у цій галузі, створила потужні обчислювальні системи, проте їх висока вартість та складність обслуговування обмежували застосування в бізнесі та побуті. З появою інтегральних схем, зокрема в калькуляторах 1960-х років, закладено фундамент для подальшого розвитку мікроелектроніки.

Незважаючи на досягнення IBM, саме компанія Intel, заснована в 1968 році,



стала піонером у розробці мікропроцесорів першого покоління. Перші **4-бітні** мікропроцесори, хоча й мали обмежені можливості, заклали основи для подальшої еволюції. Потужність таких процесорів вимірювалася кількістю бітів, що представляли найменшу одиницю інформації, яку процесор міг обробити за один такт.

Наступним етапом розвитку стала поява **8-бітних** мікропроцесорів. Компанія Intel, зберігаючи лідерські позиції, випустила в 1972 році мікропроцесор 8008, який став основою для подальших розробок. 8-бітна архітектура значно розширила можливості мікропроцесорів, дозволивши створювати більш складні та функціональні пристрої. Мікропроцесор 8080, який став наступником 8008, визначив основні напрямки розвитку мікропроцесорної техніки на багато років вперед.

Розвиток мікропроцесорної техніки у другій половині XX століття характеризувався стрімким зростанням розрядності. Після появи 8-бітних мікропроцесорів, компанії активно працювали над створенням більш потужних рішень.

### ***16-бітні мікропроцесори***

У середині 1970-х років компанії, такі як National Semiconductor [4], приєдналися до гонки за створенням 16-бітних мікропроцесорів. Ці процесори, хоча й мали обмежену продуктивність порівняно з подальшими поколіннями, знайшли своє застосування в ранніх персональних комп'ютерах, таких як Macintosh. Однак, швидкий розвиток технологій призвів до того, що 16-бітна архітектура швидко застаріла.

### ***32-бітні мікропроцесори***

До кінця 1970-х років на ринку з'явилися перші 32-бітні мікропроцесори, розроблені компаніями National Semiconductor, Hewlett-Packard [5] та іншими. Збільшення розрядності дозволило значно підвищити продуктивність та обсяг оброблюваних даних, що стало поштовхом для розвитку персональних комп'ютерів та робочих станцій. 32-бітна архітектура стала основою для подальшої еволюції мікропроцесорів.

## ***64-бітні мікропроцесори***

З початку 1990-х років на ринку з'явилися перші 64-бітні мікропроцесори. Компанії Intel та AMD стали основними конкурентами на цьому ринку. Підписання угоди між IBM та Intel дало можливість AMD виробляти сумісні з процесорами Intel мікросхеми, що призвело до посилення конкуренції та стимулювало подальший розвиток технологій.

### ***Чіну RISC***

Архітектура з обмеженим набором команд (RISC) виникла в результаті досліджень IBM, проведених у другій половині 1970-х років [6]. Головна ідея RISC полягає в оптимізації виконання простих, часто використовуваних інструкцій за рахунок відмови від складних і рідко використовуваних. Такий підхід дозволив створити процесори з високою продуктивністю та ефективністю. На сьогоднішній день архітектура RISC широко застосовується в мобільних пристроях та інших вбудованих системах, успішно конкуруючи з архітектурами CISC, що використовують комплексні набори команд.

## **1.2. Огляд мікроконтролерів**

Сучасний ринок пропонує широкий асортимент мікроконтролерів, що відрізняються за характеристиками та призначенням. Вибір оптимального мікроконтролера для конкретного проєкту вимагає ретельного аналізу вимог системи.

Основні критерії вибору мікроконтролера:

- Придатність для застосування: мікроконтролер повинен відповідати вимогам за продуктивністю, надійністю та функціональністю.
- Апаратні ресурси: кількість портів введення-виведення, наявність периферійних модулів (АЦП, ЦАП, таймери тощо) повинні відповідати потребам системи.
- Продуктивність: обчислювальна потужність ядра мікроконтролера повинна забезпечити виконання необхідних обчислень у реальному часі.

- Додаткові фактори: доступність, вартість, наявність розробницьких інструментів, підтримка виробника.

Для дослідження, проведеного в рамках даної роботи, були обрані мікроконтролери Arduino Uno та Arduino Mega. Вибір зумовлений такими факторами:

- Доступність: платформи Arduino широко доступні на ринку, мають низьку вартість і велику спільноту розробників.
- Програмна підтримка: для Arduino існує велика кількість бібліотек і прикладів коду, що значно спрощує розробку.
- Гнучкість: модульна конструкція платформ Arduino дозволяє легко розширювати функціональність за рахунок підключення додаткових модулів.
- Співвідношення ціна-якість: мікроконтролери Arduino пропонують оптимальне співвідношення ціни та функціональності.

### 1.2.1. Arduino Uno

Arduino Uno - це одна з найпопулярніших платформ для швидкого прототипування електронних пристроїв, що базується на мікроконтролері ATmega328P [7]. Створена компанією Arduino.cc, плата Uno відрізняється простотою використання, доступністю та великою спільнотою розробників.



Рис. 1.1. Мікроконтролер Arduino Uno

Плата Uno оснащена набором цифрових та аналогових входів/виходів, що дозволяє підключати різноманітні датчики, актуатори та інші електронні

компоненти. Програмування плати здійснюється в середовищі розробки Arduino IDE за допомогою мови програмування, що базується на C++. Для завантаження програмного коду використовується USB-з'єднання.

*Ключові особливості Arduino Uno:*

Мікроконтролер ATmega328P: забезпечує достатню обчислювальну потужність для більшості проєктів.

Просте програмування: середовище Arduino IDE інтуїтивно зрозуміле навіть для початківців.

Відкритість: апаратна схема та програмне забезпечення Arduino розповсюджуються за відкритими ліцензіями, що дозволяє вносити зміни та розширювати функціональність платформи.

Сумісність: існує велика кількість додаткових модулів (щитів) та бібліотек, що розширюють можливості платформи.

Спільнота: активна спільнота розробників Arduino забезпечує підтримку, обмін досвідом та розробку нових проєктів.

Назва "Uno" (з італійської - "один") була обрана на честь випуску першої версії середовища розробки Arduino IDE [8]. Плата Uno стала еталонною моделлю для платформи Arduino і широко використовується як для навчання, так і для розробки професійних проєктів.

Однією з відмінних рис Arduino Uno є використання мікросхеми ATmega16U2 в якості USB-to-serial перетворювача замість традиційних чіпів FTDI. Це дозволило спростити процес виробництва та знизити вартість плати.

Платформа Arduino Uno виникла в Інституті дизайну взаємодії Ivrea (IDII) в Італії як результат магістерського проєкту Ернандо Баррагана, виконаного під керівництвом Массімо Банзі та Кейсі Реас у 2003 році [7, 8]. Головною метою проєкту було створення доступної та простої у використанні платформи для навчання програмуванню мікроконтролерів, яка б стала альтернативою дорогим комерційним рішенням того часу.

*Технічні характеристики Arduino Uno:*

Плата Arduino Uno базується на мікроконтролері ATmega328P і має наступні основні характеристики:

Мікроконтролер: ATmega328P [7]

Напруга живлення: 5 В (номінальна), діапазон робочих напруг 7-20 В

Цифрові входи/виходи: 14 (з них 6 з функцією ШІМ)

Аналогові входи: 6

Сила струму на цифровому виході: 20 мА

Сила струму на виводі 3.3 В: 50 мА

Пам'ять: флеш-пам'ять 32 КБ (з них 0,5 КБ займає завантажувач), SRAM 2 КБ, EEPROM 1 КБ

Тактова частота: 16 МГц

Габарити: 68,6 мм x 53,4 мм

Маса: 25 г

Платформа Arduino Uno відіграла значну роль у популяризації електроніки та програмування серед широкого кола користувачів. Її відкритість, простота використання та доступність зробили Arduino одним з найпопулярніших інструментів для швидкого прототипування та навчання.

### **1.2.2. Arduino Mega 2560**

Arduino Mega 2560 є розширеною версією популярної платформи Arduino, побудованою на базі потужного мікроконтролера ATmega2560 [10]. Ця плата пропонує значно більше ресурсів порівняно зі своїми попередниками, включаючи більшу кількість цифрових та аналогових входів/виходів, а також кілька послідовних портів UART. Така архітектура дозволяє реалізувати складніші проєкти, що вимагають великої кількості периферійних пристроїв.

Плата оснащена всім необхідним для роботи: кварцовим резонатором, роз'ємами живлення та програмування, а також кнопкою скидання. Для живлення можна використовувати як зовнішній блок живлення, так і підключення до комп'ютера через USB-кабель. Arduino Mega 2560 сумісна з більшістю щитів, розроблених для попередніх версій платформи, що дозволяє розширювати її функціональність за допомогою додаткових модулів.



Рис. 1.2. Мікроконтролер Arduino Mega 2560

Плата Arduino Mega 2560 є однією з найпотужніших платформ в сімействі Arduino, що відрізняється значно розширеною функціональністю порівняно зі своїми попередниками. Однією з ключових особливостей Mega 2560 є використання мікроконтролера ATmega16U2 (або ATmega8U2 в ранніх версіях) для реалізації інтерфейсу USB-UART замість традиційних мікросхем FTDI [10]. Таке рішення спрощує процес оновлення прошивки та забезпечує більшу інтеграцію з платформою Arduino.

Починаючи з версії R2, на платі Mega 2560 був доданий резистор підтяжки до землі лінії HWB мікроконтролера 8U2, що додатково спростило процедуру переходу пристрою в режим DFU. Версія R3 принесла ще більше удосконалень: з'явилися виводи SDA і SCL для підключення зовнішніх пристроїв, а також новий вивід IOREF, який дозволяє платам розширення адаптуватися до різної напруги живлення (5 В або 3.3 В).

Arduino Mega 2560 сумісна з більшістю щитів, розроблених для попередніх версій платформи Arduino (Uno, Diecimila, Duemilanove) завдяки однаковому розташуванню основних компонентів, таких як цифрові та аналогові входи/виходи, роз'єми живлення та ICSP. Це дозволяє легко розширювати функціональність плати за допомогою додаткових модулів.

Технічні характеристики Arduino Mega 2560:

- Мікроконтролер: ATmega2560
- Напруга живлення: 5 В (номінальна), діапазон робочих напруг 7-20 В
- Цифрові входи/виходи: 54 (з них 15 з функцією ШІМ)

- Аналогові входи: 16
- Пам'ять: флеш-пам'ять 256 КБ, SRAM 8 КБ, EEPROM 4 КБ
- Тактова частота: 16 МГц

Arduino Mega 2560 є потужною і універсальною платформою для розробки електронних пристроїв. Вона поєднує в собі високу продуктивність, розширені можливості та сумісність з великою кількістю додаткових модулів. Завдяки своїм характеристикам, Arduino Mega 2560 є відмінним вибором для реалізації складних проєктів у сферах робототехніки, автоматизації та Інтернету речей.

### **1.3. Огляд периферійних пристроїв**

Периферійні пристрої відіграють ключову роль у взаємодії мікроконтролера з зовнішнім середовищем, забезпечуючи введення та виведення даних [1]. Залежно від напрямку передачі інформації, периферійні пристрої поділяються на три основні категорії:

*Пристрої вводу:* призначені для збору даних з навколишнього середовища та передачі їх до мікроконтролера. До цієї категорії належать датчики різного типу (температури, вологості, світла, тиску, відстані тощо), кнопки, джойстики, мікрофони, камери та інші пристрої, які перетворюють фізичні величини в електричні сигнали, зрозумілі мікроконтролеру.

*Пристрої виводу:* використовуються для відображення або впливу на зовнішнє середовище відповідно до отриманих від мікроконтролера команд. До цієї категорії відносяться світлодіоди, мотори, сервоприводи, гучномовці, дисплеї та інші виконавчі пристрої.

*Пристрої вводу/виводу:* здатні виконувати як функції вводу, так і функції виводу даних. Типовими прикладами таких пристроїв є пристрої зберігання даних (флеш-пам'ять, жорсткі диски) та сенсорні екрани, які можуть як зчитувати інформацію, так і відображати її.

Для створення системи контрольованого нагрівання води може знадобитися наступний набір периферійних пристроїв:

- Датчики: датчик температури DS18B20, датчик тиску MPXV5050, фоторезистор для вимірювання освітленості.

- Виконавчі пристрої: твердотільне реле для керування нагрівальним елементом, драйвер мотора для управління водяною помпою, сервопривід для регулювання клапанів, RGB-світлодіод для індикації стану системи, модуль звуку для звукових сигналів.
- Пристрої вводу/виводу: дисплей LCD 2004 для відображення інформації, потенціометр для ручної настройки параметрів, модуль Bluetooth для бездротового керування.

Використання такого набору периферійних пристроїв дозволить створити систему, яка зможе автоматично регулювати температуру води, тиск у системі, а також реагувати на зміни зовнішніх умов (наприклад, освітленості).

### **1.3.1. Пристрої вводу інформації**

#### **1.3.1.1. Потенціометр**

Потенціометр – це електронний компонент, який представляє собою змінний резистор з трьома виводами. Його основна функція полягає у плавній зміні електричного опору між двома зовнішніми виводами за допомогою рухомого контакту (повзунка) [11]. Завдяки цій властивості, потенціометр широко використовується в електронних схемах як елемент для ручного регулювання різних параметрів.



Рис. 1.3. Потенціометр

Потенціометри, як елементи змінного опору, поділяються на два основних типи: аналогові та цифрові [11].

Аналогові потенціометри функціонують на принципі безперервної зміни опору шляхом переміщення контактного повзунка по резистивному елементу. Їхньою перевагою є плавність регулювання, однак вони мають обмеження за діапазоном регулювання та габаритами.



Цифрові потенціометри використовують мережу резисторів, де кожен резистор відповідає певному дискретному значенню опору. За допомогою електронних ключів вибирається потрібний резистор, що дозволяє змінювати загальний опір з кроком. Цифрові потенціометри, як правило, реалізовані у вигляді інтегральних мікросхем і забезпечують високу точність та повторюваність налаштувань.

В проєктах на основі платформи Arduino найчастіше використовуються цифрові потенціометри, оскільки вони забезпечують більшу точність налаштувань та зручність управління за допомогою цифрових сигналів. Типове підключення цифрового потенціометра до Arduino показано на рис. 1.4.

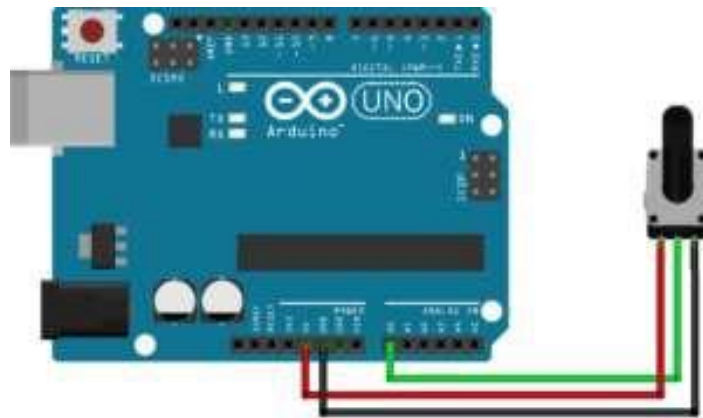


Рис. 1.4. Підключення потенціометра

Для інтеграції потенціометра в схему на базі Arduino необхідно з'єднати його виводи з відповідними виводами плати:

- Чорний провід: підключається до загального проводу (GND) для забезпечення заземлення.
- Червоний провід: підключається до джерела живлення 5 В для подачі напруги на потенціометр.
- Зелений провід (центральний вивід): підключається до аналогового входу A0 плати Arduino. Саме на цьому виводі зчитується змінна напруга, пропорційна положенню повзунка потенціометра.

Принцип роботи.

При зміні положення повзунка потенціометра змінюється опір між центральним виводом і одним з крайніх. Це призводить до відповідної зміни

напруги на центральному виводі, яка подається на аналоговий вхід Arduino. Внутрішній аналого-цифровий перетворювач (АЦП) плати Arduino перетворює отриману аналогову напругу в цифрове значення в діапазоні від 0 до 1023. Таким чином, зміна положення повзунка потенціометра відображається у зміні цифрового значення, яке може бути зчитане за допомогою функції `analogRead()`.

Екстремальні значення:

Мінімальне значення: коли повзунок потенціометра встановлений в одне крайнє положення, напруга на центральному виводі дорівнює нулю, а відповідне цифрове значення, зчитане АЦП, також дорівнює нулю.

Максимальне значення: при встановленні повзунка в протилежне крайнє положення, напруга на центральному виводі дорівнює напрузі живлення (5 В), а цифрове значення досягає максимального значення 1023.

Підключення потенціометра до Arduino дозволяє створити інтерактивні системи, в яких користувач може вручну змінювати параметри роботи пристрою. За допомогою потенціометра можна регулювати яскравість підсвічування, гучність звуку, швидкість двигунів та інші параметри. Отримане аналогове значення з потенціометра може бути використане для керування різними виконавчими пристроями або для подальшої обробки в мікроконтролері.

#### **1.3.1.2. Фоторезистор**

Фоторезистор – це пасивний електронний компонент, опір якого залежить від інтенсивності світлового потоку, що падає на його чутливу поверхню [12]. Цей ефект, відомий як фоторезистивний ефект, обумовлений зміною провідності напівпровідникового матеріалу під впливом світла.

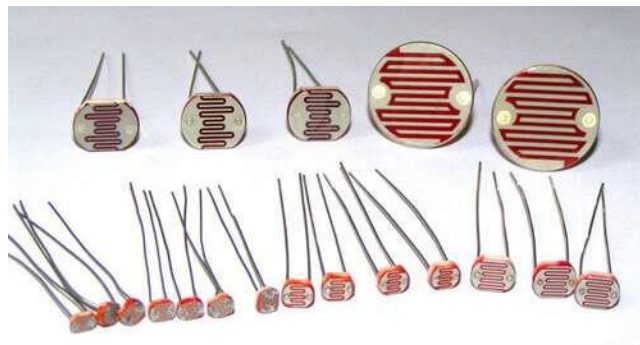


Рис.1.5. Фоторезистори

Принцип роботи

Фоторезистор складається з двох електродів, між якими розташований напівпровідниковий шар. В темряві опір цього шару високий, оскільки кількість вільних носіїв заряду в напівпровіднику невелика. При освітленні світлові фотони передають свою енергію електронам напівпровідника, збуджуючи їх і збільшуючи таким чином кількість вільних носіїв заряду. Це призводить до зменшення опору фоторезистора.

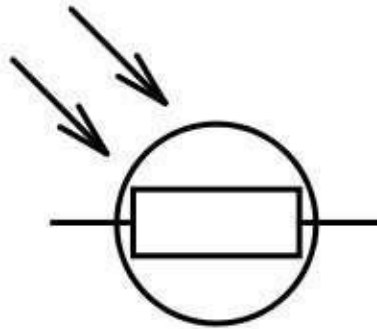


Рис.1.6. Графічне позначення фоторезистору

Для виготовлення фоторезисторів використовують різні напівпровідникові матеріали, такі як сульфід кадмію, сульфід свинцю та інші сполуки. Вибір матеріалу визначає спектральну чутливість фоторезистора, тобто діапазон довжин хвиль світла, на який він реагує. Наприклад, для детекування ультрафіолетового випромінювання використовують фоторезистори з відповідними спектральними характеристиками. Фоторезистори можуть мати різноманітну конструкцію. Найчастіше використовують фоторезистори в корпусі з прозорим віконцем, через яке світло потрапляє на чутливу поверхню. Така конструкція забезпечує захист фоточутливого елемента від зовнішніх впливів і дозволяє використовувати фоторезистор в різних умовах експлуатації.

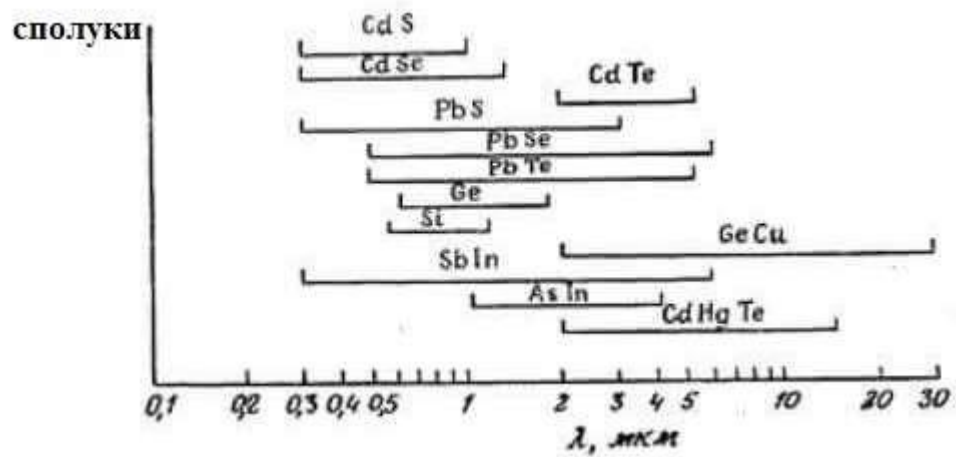


Рис. 1.7. Спектральні характеристики матеріалів

Фоторезистори відрізняються від багатьох інших напівпровідникових приладів відсутністю р-n переходу. Це означає, що для них не характерна одностороння провідність, тобто струм може протікати через фоторезистор у будь-якому напрямку без значних втрат.

Для перевірки працездатності фоторезистора можна використовувати мультиметр у режимі вимірювання опору. При цьому необхідно по чергові вимірювати опір фоторезистора в умовах освітлення та в темряві. Значне зменшення опору при освітленні свідчить про справність елемента.

Типова залежність опору фоторезистора від інтенсивності освітлення наведена на рисунку 1.8. З цього графіку видно, що зі збільшенням освітленості опір фоторезистора зменшується. Ця залежність є нелінійною, що обумовлено фізичними процесами, які відбуваються в напівпровіднику під впливом світла.

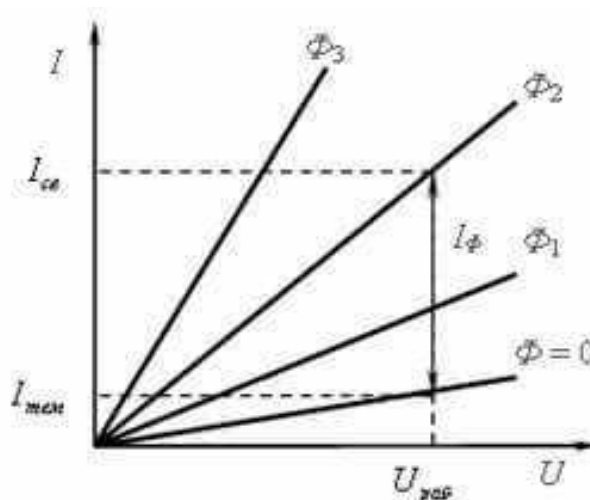


Рис. 1.8. Залежність опору від освітленості

Сила струму в ланцюзі з фоторезистором при постійній напрузі змінюється пропорційно інтенсивності світлового потоку. При відсутності освітлення ( $\Phi = 0$ ) струм мінімальний, а при яскравому освітленні ( $\Phi = \Phi_3$ ) він досягає максимального значення. Графічна залежність цієї зміни наведена на рисунку 1.9.

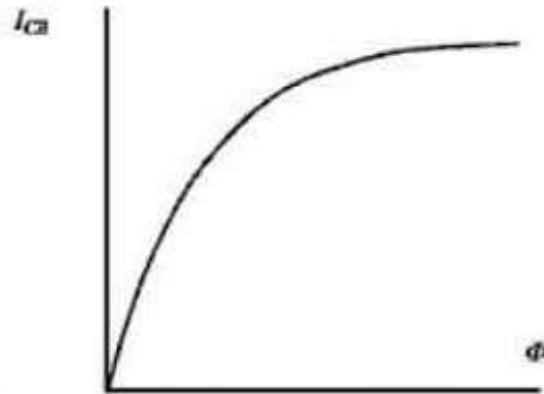


Рис. 1.9. Зміна струму за постійній напрузі

Залежність опору фоторезистора від освітленості наведено на рисунку 1.10.

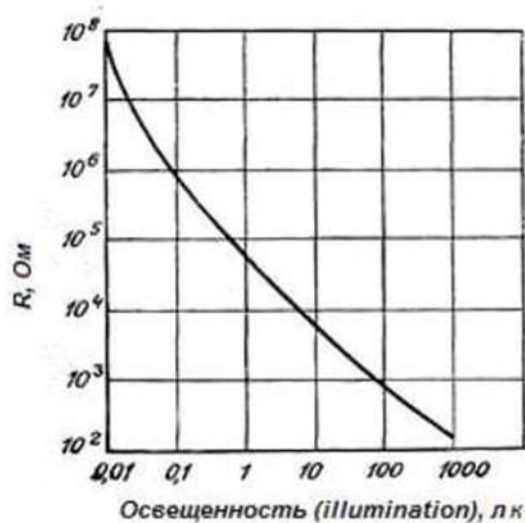


Рис. 1.10. Залежність опору від освітленості

Ключовими характеристиками фоторезистора є темновий опір (опір при відсутності освітлення) та інтегральна фоточутливість, яка характеризує чутливість елемента до зміни світлового потоку. Швидкодія фоторезистора обмежена його інерційністю, що визначається граничною частотою. Це означає, що при змінному освітленні з частотою, що перевищує граничну, фоторезистор не встигає змінювати свій опір, що призводить до зниження його ефективності. Схема підключення до мікроконтролера Arduino наведена на рис. 1.11.

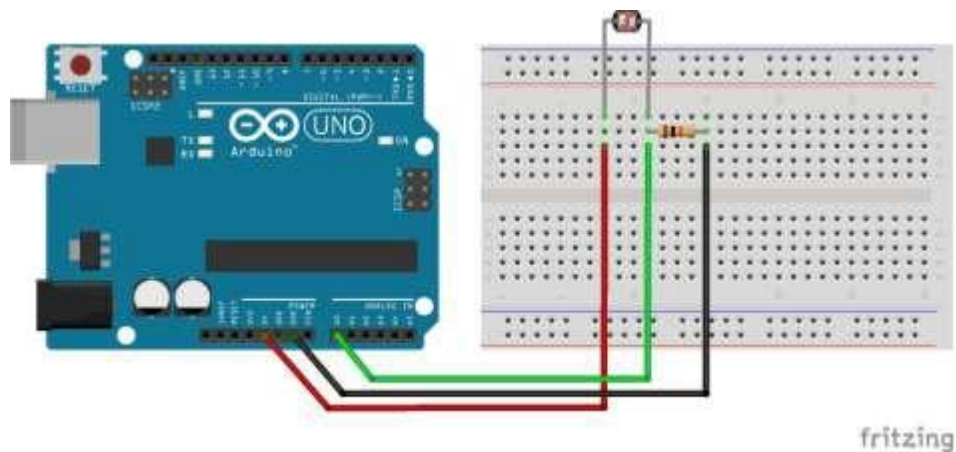


Рис.1.11. Схема підключення фоторезистора

### 1.3.1.3. Датчик температури DS18B20

DS18B20 - це інтелектуальний цифровий датчик температури, який володіє рядом корисних функцій: зберігання даних про вимірювання, сигналізація про перевищення встановлених температурних порогів, можливість зміни точності вимірювань та налаштування режиму роботи. При цьому датчик має компактні розміри і доступний у водонепроникному виконанні, що розширює сферу його застосування [13].

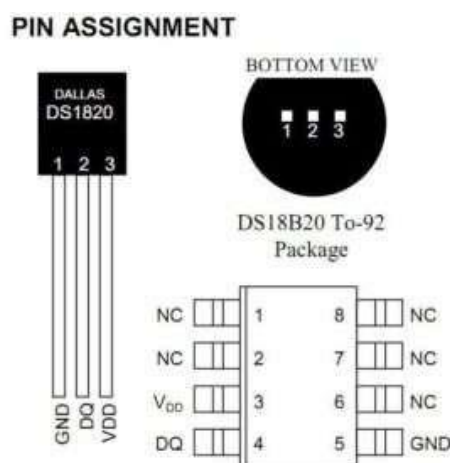


Рис. 1.12. Датчик температури DS18B20 [13]

Цифровий датчик температури DS18B20 є універсальним пристроєм, що відзначається високою точністю вимірювань та широким діапазоном робочих температур ( $-55^{\circ}\text{C} \dots +125^{\circ}\text{C}$ ) [13]. Датчик оснащений вбудованою пам'яттю (SRAM та EEPROM) для зберігання конфігураційних даних та результатів вимірювань.

Датчик доступний у різних типах корпусів, включаючи TO-92 (найбільш поширений), 8-Pin SO та 8-Pin  $\mu\text{SOP}$ . Корпус TO-92 часто виготовляється у

водонепроникному виконанні, що розширює сферу його застосування.

Для підключення датчика до мікроконтролера, як правило, використовується трипровідний інтерфейс 1-Wire. Однак, завдяки режиму "паразитного живлення" (рис. 1.13), кількість необхідних проводів можна звести до двох. Цей режим особливо зручний для віддалених датчиків.

#### *Функціональні можливості*

- Висока точність: похибка вимірювань не перевищує  $0,5^{\circ}\text{C}$  в діапазоні від  $-10^{\circ}\text{C}$  до  $+85^{\circ}\text{C}$ .
- Програмована роздільна здатність: можливість встановлення роздільної здатності вимірювань від  $0,0625^{\circ}\text{C}$  до 12 біт.
- Функція тривоги: датчик може генерувати сигнал тривоги при досягненні заданих температурних порогів.
- Унікальний ідентифікатор: кожен датчик має власний унікальний серійний номер.
- Проста інтеграція: для підключення датчика до мікроконтролера не потрібні додаткові зовнішні елементи.
- Підтримка великої кількості датчиків: на одній лінії 1-Wire можна підключити до 127 датчиків.

#### Переваги використання DS18B20:

- Висока точність і надійність.
- Широкий діапазон робочих температур.
- Простота підключення та використання.
- Компактні розміри.
- Можливість роботи в різних умовах (в тому числі у вологому середовищі).
- Гнучкі налаштування та широкий функціонал.

Датчики DS18B20 широко використовуються в різних областях, таких як:

- Автоматизація виробництва.
- Метеорологічні станції.
- Системи контролю мікроклімату.

- Медична техніка.
- Побутова електроніка.

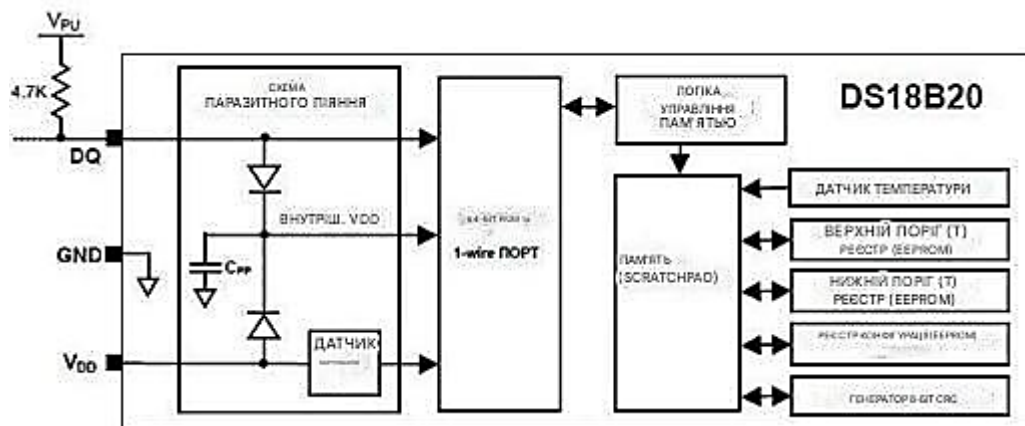


Рис. 1.13. Схема підключення «паразитного живлення»

Основне завдання датчика DS18B20 полягає у вимірюванні температури та перетворенні отриманих аналогових даних у цифровий формат, придатний для подальшої обробки мікроконтролером. Датчик дозволяє задати необхідну точність вимірювань шляхом вибору кількості бітів: 9, 10, 11 або 12 бітів, що відповідає роздільній здатності 0,5°C, 0,25°C, 0,125°C та 0,0625°C відповідно.

При підключенні живлення датчик переходить у режим очікування. Для ініціалізації процесу вимірювання мікроконтролер надсилає відповідну команду. Після завершення вимірювання результат зберігається у внутрішній пам'яті датчика (SRAM) у вигляді двох байтів. Далі датчик повертається в режим очікування.

Датчик може працювати в двох режимах: з зовнішнім живленням та в режимі паразитного живлення. В режимі зовнішнього живлення мікроконтролер керує процесом вимірювання, змінюючи стан лінії даних. В режимі паразитного живлення живлення датчика здійснюється за рахунок енергії, що знімається з лінії даних, тому на шині завжди має бути високий рівень сигналу.

Внутрішня пам'ять датчика DS18B20 розподілена таким чином:

- Байти 1-2: Зберігають виміряне значення температури.
- Байти 3-4: Використовуються для зберігання меж вимірювань.
- Байти 5-6: Зарезервовані для майбутнього використання.
- Байти 7-8: Використовуються для підвищення точності вимірювань.



- Байт 9: Зберігає CRC (циклічну перевірку надмірності) для забезпечення цілісності даних.

Датчик температури DS18B20 є цифровим пристроєм, який перетворює значення температури в двійковий код, зрозумілий для мікроконтролера Arduino. Обмін даними між датчиком і Arduino здійснюється за протоколом 1-Wire. Процес комунікації включає такі етапи:

Ініціалізація: на початку взаємодії Arduino надсилає спеціальний сигнал скидання, на який датчик відповідає сигналом готовності.

Запис даних: Arduino передає датчику команди для виконання певних операцій, наприклад, команди запуску вимірювання температури або зміни налаштувань.

Читання даних: після виконання команди датчик повертає Arduino результати вимірювань або інші необхідні дані.

Схема підключення цифрового датчика DS18B20 наведена на рисунку 1.14.

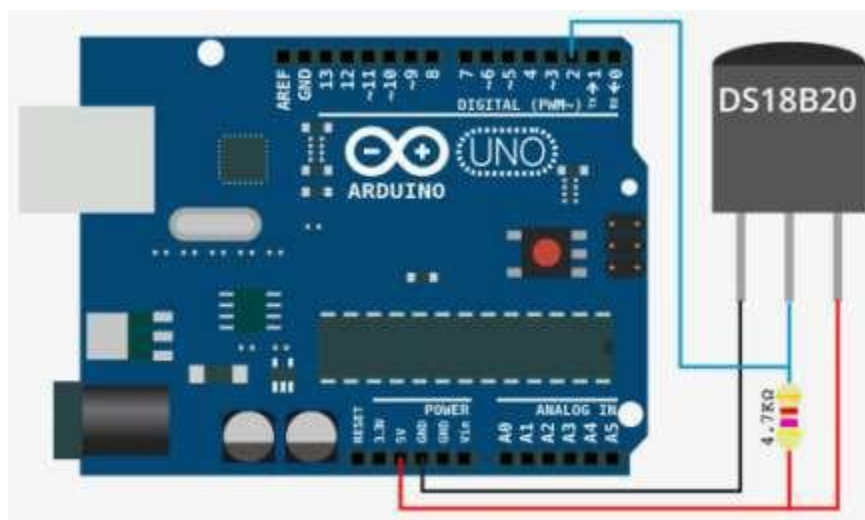


Рис. 1.14. Схема підключення цифрового датчика DS18B20

- ✓ GND з термодатчика приєднується до GND Ардуіно;
- ✓ vdd підключається до 5V;
- ✓ data - до будь-якого цифрового піну.

У режимі паразитного живлення, для живлення датчика DS18B20 використовується енергія, що знімається безпосередньо з лінії даних. Тобто, контакт Vdd датчика з'єднується з GND на Arduino. Однак, слід зазначити, що цей режим слід використовувати з обережністю, оскільки він може негативно

вплинути на швидкість роботи датчика та стабільність вимірювань.

Алгоритм зчитування температури з датчика DS18B20 передбачає виконання наступних кроків:

1. Ідентифікація датчика: спочатку необхідно визначити адресу підключеного датчика та перевірити його працездатність.
2. Ініціалізація вимірювання: датчику надсилається команда на початок процесу вимірювання температури. Варто зазначити, що цей процес може тривати до 750 мс.
3. Зчитування результатів: після завершення вимірювання з датчика зчитується отримане значення температури і записується у відповідний регістр.
4. Виведення даних: зчитане значення температури виводиться на монітор послідовного порту у зручному для користувача форматі. При необхідності виконується конвертація одиниць вимірювання (з градусів Цельсія в градуси Фаренгейта).

#### 1.3.1.4. Датчик тиску MPXV5050

Датчик тиску MPXV5050 спроектований для вимірювання тиску до 50 кПа, що відповідає приблизно тиску водяного стовпа висотою 5 метрів.

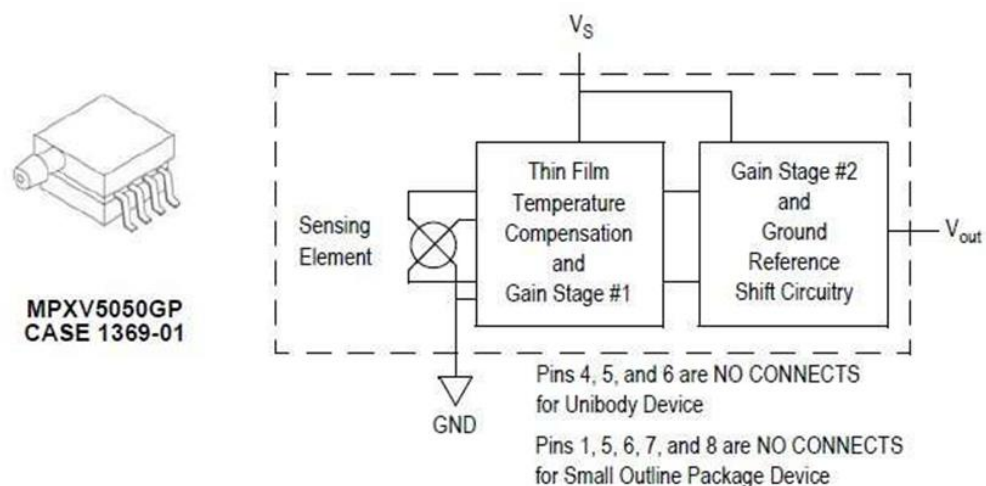


Рис. 1.15. Датчик тиску MPXV5050 [14]

Технічні характеристики наведено у таблиці 1.1.

**Технічні характеристики датчика тиску MPXV5050**

Тип датчику	Інтегрований
Тип тиску	Відносний
POР,кПа	від 0 до 50
PМАХ,кПа	200
VFSS (тип.),мВ	4.5
Нелінійність, %	2.5
Чутливість,мВ/кПа	90
VCC,В	від 4.75 до 5.25
ICC,мА	7
ТА,°С	від -40 до 125

**Конструкція для вимірювання тиску рідини [14]**

Для вимірювання гідростатичного тиску у водному середовищі пропонується використовувати конструкцію, що складається з мідної трубки, зануреної у воду, та датчика тиску, підключеного до верхнього кінця трубки [14]. Вибір міді як матеріалу для трубки обумовлений її високими фізико-хімічними властивостями: високою тепло- та електропровідністю, стійкістю до корозії, а також хорошими технологічними характеристиками [15].

Занурена вертикально мідна трубка з відкритим нижнім кінцем заповнюється водою до рівня, що відповідає глибині вимірювання. Датчик тиску, підключений до верхнього кінця трубки, реєструє гідростатичний тиск водяного стовпа, який прямо пропорційний глибині занурення.

Слід зазначити, що через деяку стисливість води та повітря, що міститься в трубці, рівень води всередині трубки дещо відрізняється від рівня води у водоймі. Тому датчик фактично реєструє тиск на рівні Н1, який дещо вищий за глибину занурення нижнього кінця трубки. Однак, якщо відома величина цього перепаду тиску, то можна з високою точністю визначити глибину занурення, за умови, що внутрішній діаметр трубки є постійним по всій її довжині.

Ключові переваги запропонованої конструкції:

- Простота: конструкція відносно проста у виготовленні та експлуатації.
- Надійність: мідь є стійким до корозії матеріалом, що забезпечує

довготривалу роботу датчика.

- Точність: при правильному калібруванні та врахуванні поправки на стисливість води, можна досягти високої точності вимірювань.

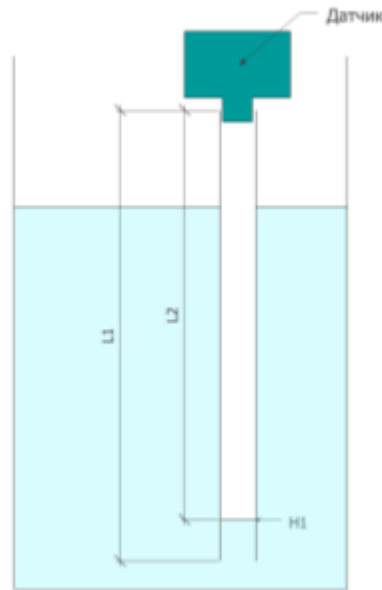


Рис. 1.16. Конструкція вимірювання тиску

$P$  - виміряний тиск всередині трубки,  $P_a$  - початковий атмосферний тиск,  $V_1$  - початковий об'єм повітря,  $V_2$  - об'єм після занурення,  $L_1$  - довжина всієї трубки,  $L_2$  - довжина трубки від датчика до  $H_1$ .

Оскільки кількість повітря в трубці не змінюється, то

$$P_a * V_1 = P * V_2 \quad [16],$$

Оскільки перетин (площа) труби постійна, то  $P_a * L_1 = P * L_2$ ;

$$L_2 = (P_a * L) / P.$$

Тиск водяного стовпа висотою 1 метр дорівнює 10 кПа, а атмосферний тиск орієнтовно 100 кПа [17], тому можна порахувати на скільки стиснено повітря, якщо опустити трубку завдовжки 1 метр, на глибину 1 метр.

$$L_2 = (100 * 1\text{м}) / (100 + 10) = 0.9 \text{ м}$$

Тобто вода буде на рівні 10 см від нижнього зрізу труби.

#### 1.3.1.5. Інфрачервоний модуль

Інфрачервоні модулі, що складаються з приймача та передавача (пульта дистанційного керування), є одним з найпоширеніших і найпростіших способів бездротового управління електронними пристроями. Інфрачервоне

випромінювання, невидиме для людського ока, легко виявляється спеціалізованими приймачами, інтегрованими в сучасну електроніку. Бібліотека Arduino IRremote забезпечує зручний інструментарій для створення власних систем дистанційного керування різноманітними пристроями за умови прямої видимості між пультом та приймачем [18].



Рис. 1.17. ІЧ приймач та інфрачервоний пульт

Інфрачервоне (ІЧ) випромінювання, невидиме для людського ока, широко використовується в сучасних електронних пристроях завдяки своїм унікальним властивостям та доступності. Воно займає спектральний діапазон від 750 до 1000 нанометрів, що робить його найближчим до видимого світла. Однією з ключових переваг ІЧ-випромінювання є його взаємодія з різними матеріалами: деякі матеріали, такі як скло, стають непрозорими для ІЧ-променів, тоді як інші, наприклад, парафін, залишаються прозорими.

Системи дистанційного керування, що використовують ІЧ-випромінювання, складаються з двох основних компонентів: ІЧ-передавача (пульту) та ІЧ-приймача. Принцип їх роботи полягає в наступному:

*Передача сигналу:* при натисканні кнопки на пульті дистанційного керування, інформація про натиснуту кнопку кодується у вигляді послідовності імпульсів інфрачервоного світла. Ці імпульси мають певну частоту і тривалість.

*Прийом сигналу:* ІЧ-приймач, встановлений в керованому пристрої, захоплює інфрачервоне випромінювання і декодує отриманий сигнал. Після декодування пристрій виконує відповідну команду.

Переваги використання ІЧ-випромінювання в системах дистанційного керування:

- Низька вартість: ІЧ-діоди та ІЧ-приймачі є недорогими компонентами.
- Компактність: розміри ІЧ-компонентів дозволяють легко інтегрувати їх в різноманітні пристрої.
- Безпека для здоров'я: ІЧ-випромінювання нешкідливе для людського організму.
- Стійкість до перешкод: завдяки вузькоспрямованому характеру ІЧ-променів, рівень перешкод від інших джерел випромінювання є незначним.
- Проста реалізація: створення систем дистанційного керування на основі ІЧ є відносно простим завданням завдяки доступності готових рішень та бібліотек для мікроконтролерів.

Обмеження:

- Пряма видимість: для коректної роботи системи необхідна пряма видимість між пультом і приймачем.
- Коротка дальність дії: дальність дії ІЧ-систем обмежена кількома метрами.

Інфрачервоне дистанційне керування широко використовується в побутовій електроніці (телевізори, кондиціонери, аудіосистеми), промисловості (автоматизація виробництва) та інших сферах.

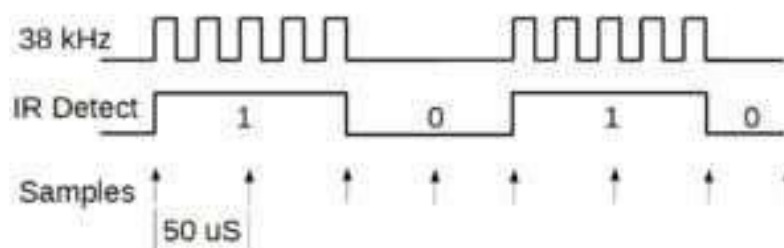


Рис. 1.18. Послідовність пакетів імпульсів

Приймач інфрачервоного сигналу – це електронний компонент, який перетворює невидиме для людського ока інфрачервоне випромінювання, що

надходить від пульта дистанційного керування, на електричний сигнал, зрозумілий для мікроконтролера. Внутрішньо приймач містить фотодіод, який перетворює світловий сигнал на слабкий електричний струм. Цей сигнал потім підсилюється, фільтрується для усунення шумів та демодулюється, тобто витягується корисна інформація. Для цих цілей використовуються спеціальні мікросхеми, такі як TSOP312, які вже містять усі необхідні компоненти: фотодіод, підсилювачі, фільтри та демодулятор. Зовнішні елементи, такі як резистори та конденсатори, можуть використовуватися для додаткової настройки.

Дані передаються за протоколами RC5 або NEC, які визначають структуру і формат інформації, що передається. Типовий приймач має три виводи: живлення, земля та вихідний сигнал, який підключається до мікроконтролера. Дальність дії таких систем зазвичай становить до 40 метрів, але може змінюватися залежно від потужності передавача, чутливості приймача та наявності перешкод.

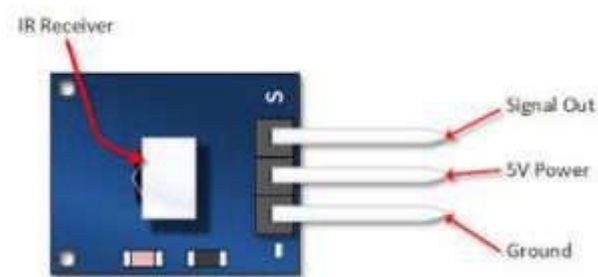


Рис. 1.19. Виводи ІЧ приймача

Для зручності можна використовувати готові модулі ІЧ приймача.



Рис. 1.20. Модуль ІЧ приймача

Схема підключення датчика до Ардуіно зображена на рисунку 1.21:

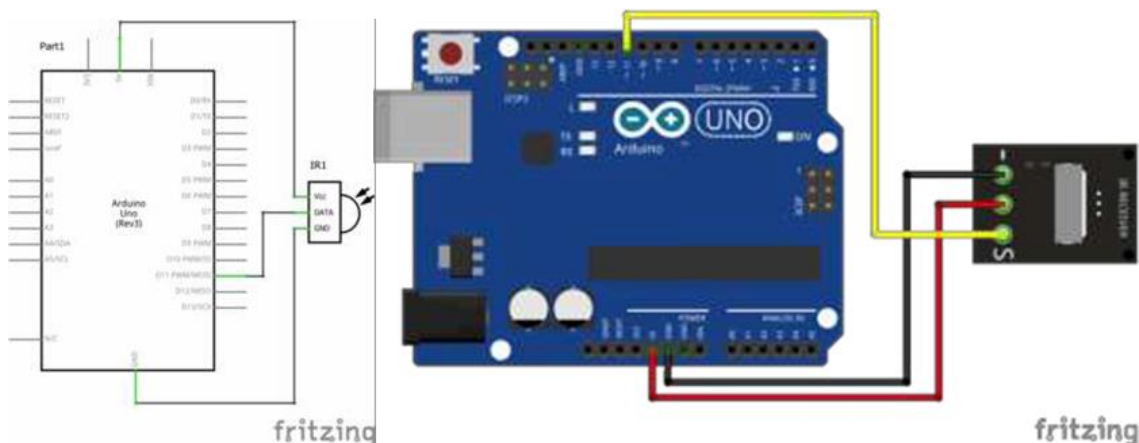


Рис. 1.21. Схема підключення до Ардуіно

## 1.3.2. Пристрої виводу інформації

### 1.3.2.1. Реле для комутації

Реле: це електромеханічний або електронний пристрій, який дозволяє керувати потужними електричними ланцюгами за допомогою низьковольтних сигналів. Простіше кажучи, реле – це своєрідний перемикач, який може вмикати і вимикати сильні струми, що протікають через навантаження, наприклад, двигуни, лампи розжарювання чи насоси.

Мікроконтролери, такі як Arduino, здатні обробляти інформацію та приймати рішення, але вони не спроектовані для безпосереднього керування потужними навантаженнями. Підключення таких навантажень безпосередньо до виходів мікроконтролера може призвести до його пошкодження. Реле ж дозволяє ізолювати мікроконтролер від потужного навантаження, забезпечуючи надійну та безпечну роботу системи.

Основний принцип роботи електромагнітного реле полягає в наступному: коли через котушку реле протікає електричний струм, створюється магнітне поле, яке притягує рухомий контакт (якорь). В результаті цього контакту замикається або розмикається електричний ланцюг навантаження.

Основні характеристики реле:

- Напруга і струм спрацювання: мінімальні значення напруги або струму, необхідні для спрацювання реле.
- Напруга і струм відпускання: значення напруги або струму, при яких реле повертається в початковий стан.



- Час спрацювання і відпускання: швидкість, з якою реле реагує на зміну вхідного сигналу.
- Робочі струм і напруга: максимально допустимі значення струму і напруги в контактній групі реле.
- Внутрішній опір: опір котушки реле.

Існують два основних типи реле:

Електромагнітні реле: класичні реле, в яких рухомий контакт приводиться в дію електромагнітом.

Твердотільні реле: сучасні реле, в яких замість механічних контактів використовуються напівпровідникові елементи. Вони мають більш тривалий термін служби та менші розміри.

### **Електромагнітне реле**

Електромагнітне реле: це електромеханічний пристрій перемикавання, дія якого ґрунтується на принципі електромагнетизму. Принцип роботи реле полягає в тому, що при проходженні електричного струму через котушку, створюється магнітне поле, яке притягує до себе феромагнітний якор. Рух якоря призводить до механічного перемикавання контактів, що дозволяє керувати електричним ланцюгом навантаження.



Рис. 1.22. Електромагнітне реле

Електромагнітне реле: це електромеханічний пристрій, що використовується для дистанційного керування електричними ланцюгами. Принцип його роботи базується на взаємодії електромагнітного поля та механічних сил.

При подачі керуючого сигналу на котушку реле, створюється магнітне поле, яке притягує до себе феромагнітний якор, переміщуючи його. Рух якоря

призводить до механічного замикання або розмикання електричних контактів, таким чином змінюючи стан керованого ланцюга. У відсутності керуючого сигналу, якір повертається у вихідне положення під дією пружини або сили тяжіння.

Електромагнітні реле широко застосовуються в автоматичних системах керування, електроприводах, релейному захисті та інших галузях електротехніки. Вони дозволяють:

- Регулювати напругу та струм в електричних ланцюгах.
- Виконувати функції накопичувачів та перетворювачів електричної енергії.
- Фіксувати відхилення параметрів від заданих значень.

Електромагнітні реле класифікують за різними ознаками:

- За родом струму: на реле постійного та змінного струму.
- За конструкцією: на якірні та герконові реле.
- За швидкістю дії: розрізняють реле з різною швидкістю спрацювання.
- За ступенем захисту: реле можуть бути герметичними, закритими та відкритими.

### **Твердотільні реле**

Твердотільне реле (ТТР) – це електронний комутаційний пристрій, створений на основі напівпровідникових компонентів, таких як транзистори, симістори або тиристори. На відміну від електромагнітних реле, ТТР не має рухомих механічних частин, що забезпечує високу надійність та довговічність. Конструктивно ТТР часто виконуються за гібридною технологією [20], що передбачає комбінування різних електронних компонентів на єдиній підкладці. Така конструкція дозволяє створювати компактні та функціонально насичені пристрої.



Рис. 1.23. Твердотільне реле

Твердотільні реле: це електронні пристрої комутації, що базуються на напівпровідникових компонентах, зокрема транзисторах, симісторах та тиристорах. На відміну від електро cơханічних реле, твердотільні реле не мають рухомих частин, що забезпечує високу надійність, довговічність та безшумність роботи.

Переваги твердотілих реле:

1. Довгий термін служби: відсутність механічних контактів мінімізує знос і підвищує надійність.
2. Висока швидкість спрацювання: швидке перемикання без затримок.
3. Компактні розміри: можливість розміщення на друкованих платах.
4. Відсутність акустичних перешкод: тиха робота без виникнення іскор та дуг.
5. Низьке енергоспоживання: ефективне використання енергії.
6. Висока ізоляція: забезпечує безпеку роботи.
7. Стійкість до вібрацій та ударів: збереження працездатності в складних умовах.
8. Можливість використання у вибухонебезпечних середовищах: відсутність іскроутворення.

Твердотільне реле працює на основі оптичної ізоляції та керування силовими ключами. Керуючий сигнал подається на світлодіод, світловий потік якого через оптичну систему потрапляє на фотодіод. Сигнал з фотодіода керує силовим ключем (транзистором, симістором або тиристором), який і комутує навантаження.

Недоліки:

- Нагрівання: при великих струмах і напругах можливе значне нагрівання пристрою, що обмежує його робочі характеристики.
- Вартість: зазвичай твердотільні реле дорожчі за електромагнітні.

Твердотільні реле класифікують за такими ознаками:

- Тип навантаження: однофазні та трифазні.
- Спосіб управління: постійним або змінним струмом, ручне або автоматичне.
- Метод комутації: контроль переходу через нуль, довільне включення, фазове управління.

Твердотільні реле широко застосовуються в автоматичній, електроніці, побутовій техніці та промисловості. Одним з популярних прикладів є модулі типу SONGLE SRD-05VDC, які використовуються для керування навантаженнями в схемах на основі мікроконтролерів, таких як Arduino.

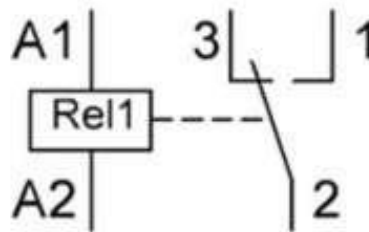


Рис. 1.24. Схема реалізації реле

Між A1 і A2 знаходиться металевий сердечник. При ввімкненні електричного струму, до нього притягнеться якор (2). 1, 3 - нерухомі контакти. За відсутності струму якор є близько контакту 3.

Схема підключення реле до Ардуїно зображена на рисунку 1.25

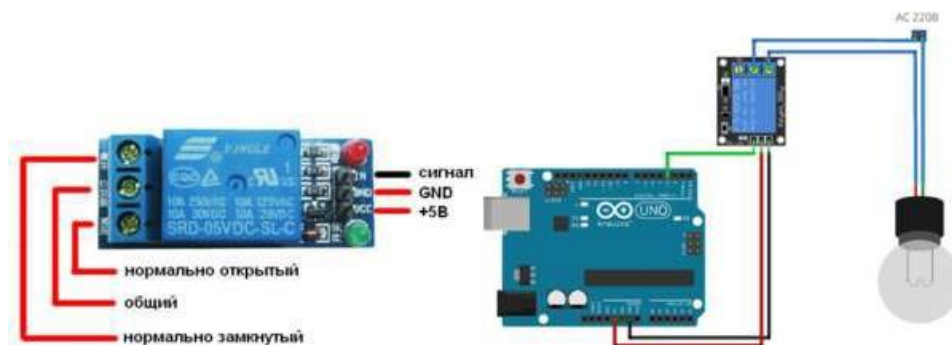


Рис. 1.25. Схема підключення реле до Ардуїно

Реле - це електромеханічний пристрій, який дозволяє керувати електричними ланцюгами за допомогою невеликого керуючого сигналу. Для підключення реле до мікроконтролера Arduino, як правило, використовують три основні контакти: загальний (COM), нормально відкритий (NO) та нормально замкнутий (NC).

Процес підключення:

1. Заземлення: контакт GND реле з'єднується з аналогічним контактом на платі Arduino.
2. Живлення: контакт VCC реле підключається до джерела живлення 5В на Arduino.
3. Керування: контакт In реле під'єднується до цифрового виходу Arduino, який буде подавати керуючий сигнал.

Більшість реле працюють за принципом інверсії: коли на вхід In подається високий логічний рівень (логічна одиниця), котушка реле знеструмлюється, а коли подається низький рівень (логічний нуль), котушка вмикається. Це означає, що для керування реле ми фактично змінюємо логічний стан на його вході.

Для більш плавного та надійного керування котушкою реле часто використовують транзистор. Коли на базу транзистора подається низький логічний рівень, він відкривається, дозволяючи протікати струму через котушку реле. Таким чином, реле включається. Для вимкнення реле на базу транзистора подається високий логічний рівень, що призводить до закриття транзистора і знеструмлення котушки.

Для візуального контролю за роботою реле часто використовують світлодіоди. Червоний світлодіод може сигналізувати про подачу живлення на реле, а зелений - про замикання контактів.

Для керування станом реле в програмі для Arduino використовується функція `digitalWrite()`. Встановивши логічний нуль на відповідному піні, ми вмикаємо реле, а встановивши логічну одиницю - вимикаємо.

Спрощена схема роботи:

При подачі живлення на Arduino транзистор, який керує реле, знаходиться у закритому стані.

Коли ми хочемо включити реле, ми виконуємо команду `digitalWrite()`, яка встановлює низький логічний рівень на піні, підключеному до бази транзистора.

Транзистор відкривається, струм протікає через котушку реле, і контакти реле змінюють свій стан відповідно до їх призначення (нормально відкриті замикаються, нормально замкнуті розмикаються).

Для вимкнення реле ми виконуємо команду `digitalWrite()`, яка встановлює високий логічний рівень на піні, закриваючи транзистор і знеструмлюючи котушку реле.

Підключення і керування реле за допомогою Arduino є відносно простим процесом, який вимагає розуміння основних принципів роботи цифрових виходів мікроконтролера та роботи самого реле. Використання транзистора для керування котушкою реле забезпечує надійну ізоляцію між мікроконтролером і потужним навантаженням, яке може підключатися до контактів реле.

#### **1.3.2.2. Драйвер двигуна**

Мікроконтролери Arduino, такі як Uno, Mega чи Nano, хоч і є потужними інструментами для створення різних електронних пристроїв, мають обмеження за силою струму, який вони можуть забезпечити. Загальна сила струму, яку може споживати весь мікроконтролер, зазвичай обмежена 800 міліампер, а для кожного окремого виходу цей ліміт ще менший - близько 40 міліампер.

Це означає, що безпосереднє підключення до Arduino таких пристроїв, як електродвигуни постійного струму, сервоприводи або водяні насоси, є неможливим. Під час запуску або зупинки цих пристроїв виникають сильні пікові струми, які можуть значно перевищити допустимі значення для виходів Arduino і призвести до пошкодження мікроконтролера.

Для вирішення цієї проблеми використовують спеціальні мікросхеми або модулі, які називають драйверами двигунів. Основним елементом таких драйверів є так званий Н-міст. Н-міст - це електронна схема, яка дозволяє

змінювати напрямок струму в електричному ланцюзі, тобто, керувати обертанням двигуна.

Драйвери двигунів дозволяють підключити до Arduino потужні навантаження, такі як електродвигуни, сервоприводи та інші, забезпечуючи при цьому необхідну ізоляцію та захист мікроконтролера від перевантажень. Існують різноманітні драйвери двигунів, які відрізняються за кількістю каналів (тобто, кількістю двигунів, які можна підключити), силою струму, що витримується, та набором додаткових функцій.

### **Мікросхема або плата розширення Motor Shield**

Motor Shield представляє собою спеціалізовану плату розширення для мікроконтролерів Arduino, призначену для ефективного керування електродвигунами. Вона забезпечує необхідну електроніку та інтерфейси для підключення та управління як двигунами постійного струму, так і кроковими двигунами. Найпоширенішими мікросхемами, що використовуються в таких платах, є L298N та L293D, які дозволяють одночасно керувати кількома двигунами.

Ключовою особливістю Motor Shield є наявність стандартного набору контактів Arduino, що спрощує її інтеграцію в існуючі проекти. Це дозволяє розширювати функціональність системи шляхом підключення інших плат розширення. Крім того, більшість моделей Motor Shield мають можливість вибору джерела живлення, що забезпечує гнучкість у конструюванні електронних пристроїв. Для індикації робочого стану на платі зазвичай передбачений світлодіод.

### **Принцип дії Н-моста в драйверах двигунів**

Основним елементом, що забезпечує функціональність Motor Shield, є електронна схема, відома як Н-міст. Свою назву вона отримала завдяки характерній конфігурації, яка нагадує букву "Н". Н-міст складається з чотирьох ключових елементів (транзисторів або інших напівпровідників), з'єднаних певним чином. Завдяки цій конфігурації, Н-міст дозволяє змінювати напрямок електричного струму, який протікає через обмотки двигуна, що, в свою чергу,

забезпечує зміну напрямку обертання валу. Таким чином, за допомогою Н-моста можна керувати не тільки швидкістю обертання двигуна, але й його напрямком. Схема моста зображена на рисунку 1.26.

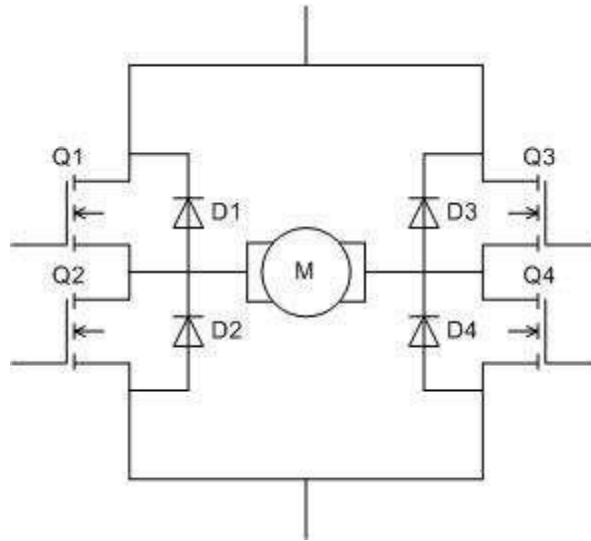


Рис. 1.26. Електронна схема Н-мосту

Н-міст є фундаментальною схемою електроніки потужності, що широко застосовується для керування електродвигунами постійного струму. Його основна функція полягає у зміні напрямку струму в навантаженні, що дозволяє регулювати швидкість обертання та напрямок руху двигуна.

Типовий Н-міст складається з чотирьох силових ключів (Q1, Q2, Q3, Q4), які зазвичай реалізовані на основі потужних транзисторів. Вибір типу транзистора залежить від конкретних вимог до схеми:

IGBT-транзистори є оптимальним вибором для високовольтних додатків завдяки їхній високій швидкодії та здатності витримувати великі струми.

Польові транзистори з ізольованим затвором (MOSFET) широко використовуються для керування двигунами завдяки їхній високій швидкодії та простоті керування.

Біполярні транзистори рідше застосовуються в потужних схемах через меншу швидкодію та складність керування. Вони можуть використовуватися лише в малопотужних схемах.

Для запобігання короткого замикання джерела живлення між виводами Н-моста паралельно ключам підключаються діоди (D1, D2, D3, D4). Зазвичай для



цих цілей використовують швидкісні діоди Шотткі, які мають низьке падіння напруги і високу ефективність.

Змінюючи комбінацію відкритих і закритих ключів в Н-мості, можна досягти різних режимів роботи двигуна:

*Пряме обертання:* відкриваються певна пара ключів, створюючи шлях для протікання струму в одному напрямку через обмотки двигуна.

*Зворотне обертання:* відкривається інша пара ключів, змінюючи напрямок струму і, відповідно, напрямок обертання двигуна.

*Гальмування:* одночасне закриття всіх ключів призводить до гальмування двигуна за рахунок перетворення кінетичної енергії обертання в теплову.

Таблиця 2.2

#### Основні стани Н-моста

Q1	Q2	Q3	Q4	Стан
1	0	0	1	Поворот мотора праворуч
0	1	1	0	Поворот мотора ліворуч
0	0	0	0	Вільне обертання
0	1	0	1	Гальмування
1	0	1	0	Гальмування
1	1	0	0	Коротке замикання
0	0	1	1	Коротке замикання

#### Драйвер двигуна L298N

Розглядуваний модуль призначений для керування кроковими двигунами з широким діапазоном робочої напруги – від 5 до 35 вольт. Його функціональність забезпечується за допомогою інтегральної мікросхеми L298N [22], яка є спеціалізованим драйвером для керування електродвигунами. Однією з ключових переваг цієї мікросхеми є можливість одночасного керування двома кроковими двигунами.

Кожен з виходів мікросхеми L298N здатний забезпечити максимальний струм до 2 ампер. Це означає, що кожен підключений до модуля двигун може споживати до 2 ампер. У випадку, якщо необхідно збільшити загальний струм, що подається на навантаження, існує можливість паралельного підключення кількох двигунів. Однак слід зазначити, що при такому підключенні необхідно ретельно розрахувати загальний струм, щоб уникнути перевантаження

мікросхеми та інших елементів модуля.



Рис. 1.27. Драйвер двигуна L298N

Виводи мікросхеми L298N зображено на рисунку 1.28:



Рис. 1. 28. Виводи мікросхеми L298N

Розглянутий модуль призначений для керування двома електродвигунами постійного струму і забезпечує необхідну електроніку для їхнього підключення та управління. Функціональні блоки модуля мають такі призначення:

- Vcc: до цього контакту підключається зовнішнє джерело живлення, яке забезпечує енергією всю схему.
- GND: контакт для підключення загального проводу (землі).
- IN1, IN2, IN3, IN4: ці входи використовуються для керування роботою мікросхеми L298N та, відповідно, для регулювання напрямку обертання і швидкості двигунів. Шляхом зміни логічних рівнів на цих входах можна досягти плавного регулювання швидкості обертання.
- OUT1, OUT2, OUT3, OUT4: це виходи, до яких підключаються обмотки двигунів.

- S1: цей перемикач дозволяє вибрати джерело живлення для схеми: зовнішнє або внутрішній стабілізатор.
- ENABLE A, ENABLE B: ці входи використовуються для увімкнення/вимкнення відповідних каналів і дозволяють керувати кожним двигуном окремо. Існують два режими роботи: активний, коли швидкістю двигуна можна керувати за допомогою мікроконтролера, та пасивний, коли двигун працює на максимальних обертах.

При підключенні двох двигунів до модуля необхідно звернути увагу на полярність їх підключення. Неправильне підключення може призвести до обертання двигунів у протилежні сторони від очікуваного. Тому перед включенням схеми слід ретельно перевірити правильність підключення всіх проводів.

Детальна схема підключення модуля L298N до мікроконтролера Arduino представлена на рисунку 1.28. Ця схема ілюструє, як правильно підключити всі необхідні сигнали та живлення для забезпечення коректного функціонування системи.

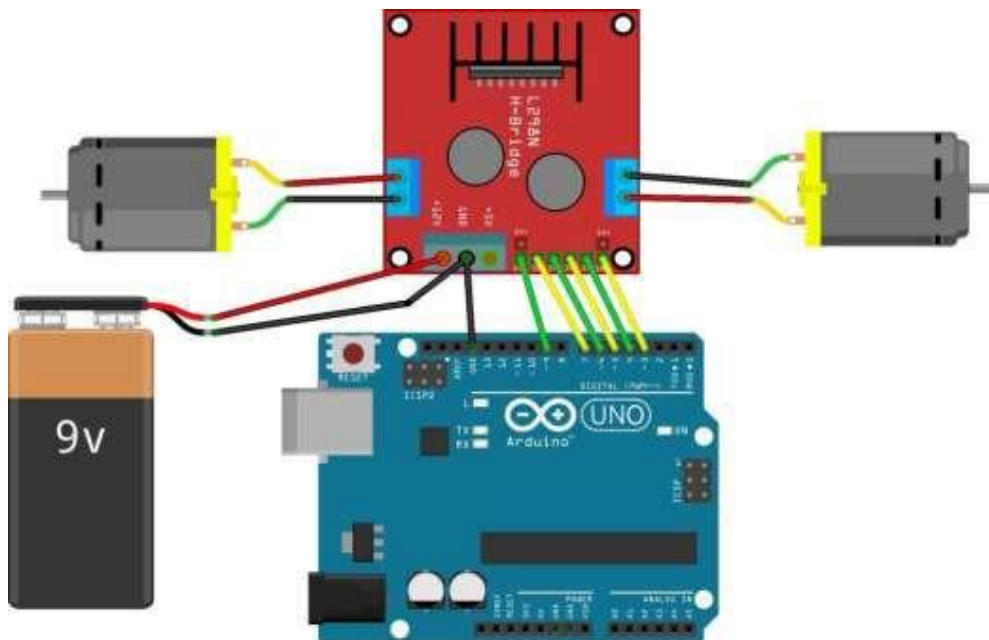


Рис. 1.28. Підключення L298N до Arduino

Напрямок обертання електродвигуна визначається логічними рівнями сигналів (HIGH або LOW), що подаються на відповідні входи драйвера. Для

плавного регулювання швидкості обертання застосовується модуляція ширини імпульсів (ШІМ). Шляхом зміни співвідношення часу знаходження сигналу в високому і низькому станах, можна плавно змінювати середнє значення напруги на обмотках двигуна, тим самим регулюючи його швидкість.

### 1.3.2.3. Сервопривід

Сервоприводи (також відомі як серводвигуни) є незамінними компонентами в системах, де вимагається висока точність руху або підтримання певного кута повороту валу. Завдяки вбудованим датчикам положення, сервоприводи забезпечують точне позиціонування виконавчих механізмів навіть при відносно невеликих швидкостях і середньому крутному моменті. Типовими прикладами застосування сервоприводів є робототехніка, системи управління польотом моделей літаків та дронів, а також різноманітні промислові автоматизовані системи. Для ілюстрації можна навести приклад популярної моделі сервопривода SG90 [23].



Рис. 1.29. Сервопривід SG90

Серводвигуни є електромеханічними пристроями, що забезпечують точне позиціонування виконавчих механізмів. Вони складаються з електродвигуна постійного струму, редуктора, датчика положення та електронної схеми управління. Для підключення серводвигуна необхідні три провідники: живлення, заземлення та сигнал. Сигнал управління представляє собою ШІМ-сигнал змінної ширини імпульсів, що визначає кут повороту валу. Електронна схема сервопривода порівнює вхідний ШІМ-сигнал з відгуком датчика положення і генерує керуючий сигнал для двигуна таким чином, щоб зменшити похибку.

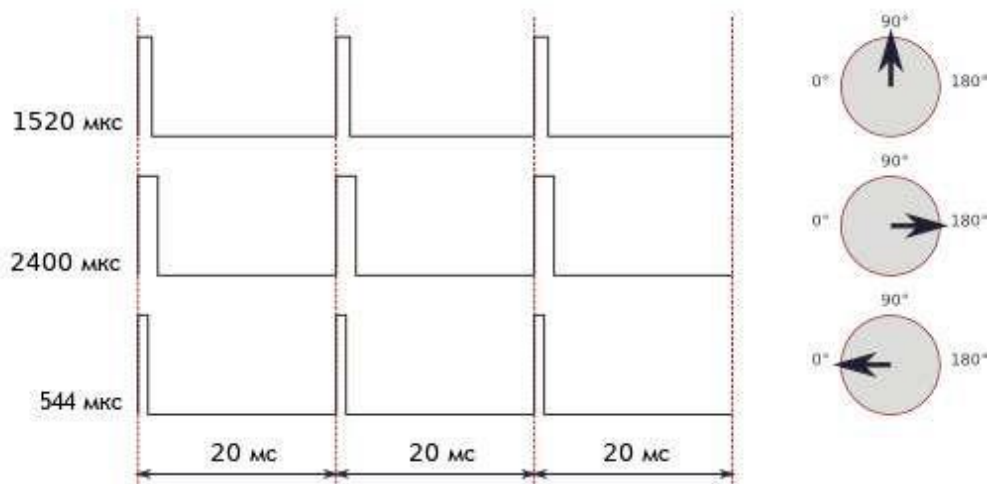


Рис. 1.30. Відношення довжини імпульсу до кута повороту

Керування положенням сервопривода здійснюється за допомогою імпульсів заданої частоти, але змінної тривалості. Ці імпульси надходять на вхід сервопривода і порівнюються з внутрішнім еталонним імпульсом. Різниця між тривалістю зовнішнього та еталонного імпульсів визначає напрямок і величину повороту валу сервопривода. Якщо зовнішній імпульс коротший за еталонний, вал обертається в одному напрямку, а якщо довший – в протилежному. Коли тривалості імпульсів рівні, вал зупиняється.

Частота і тривалість імпульсів мають стандартизовані значення. Найчастіше використовується частота 50 Гц, що відповідає періоду імпульсу 20 мс. Діапазон тривалості імпульсів, як правило, становить від кількох мілісекунд до декількох десятків мілісекунд. Кожній тривалості імпульсу відповідає певний кут повороту валу сервопривода. Наприклад, для сервопривода SG-90 [23] мінімальна тривалість імпульсу (близько 544 мкс) відповідає нульовому куту повороту, а максимальна (близько 2400 мкс) – 180°.

Сервопривод SG-90 [23] є типовим представником мініатюрних сервоприводів і має такі характеристики:

Маса: 9 г.

Розміри: 21,5 x 11,8 x 22,7 мм.

Напруга живлення: 4,8-6 В.

Крутний момент: 1,2 кг\*см при напрузі 4,7 В.

Час повороту на 60°: 0,12 с при напрузі 4,7 В.

Робоча температура: від  $-30^{\circ}\text{C}$  до  $+60^{\circ}\text{C}$ .

Довжина кабелю: 23 см.

Матеріал шестерень: нейлон.

Тип: аналоговий.

Сервоприводи є універсальними пристроями для автоматизації рухів у різних системах. Їхнє застосування обмежується лише фантазією розробника. Завдяки своїй компактності і простоті керування, сервоприводи SG-90 широко використовуються в робототехніці, моделюванні та інших галузях.

#### **1.3.2.4. Водяна помпа**

Мініатюрна водяна помпа призначена для перекачування невеликих об'ємів рідини зі швидкістю до 120 літрів на годину. Завдяки своїм компактным розмірам і низькій напрузі живлення (2.5-6 В), вона широко застосовується в побутових системах автоматичного поливу рослин, акваріумістиці та гідропоніці. Можливе використання помпи для підтримки рівня води в системах нагрівання, зокрема, у поєднанні з сонячними батареями.



Рис. 1.31. Водяна помпа

Мініатюрна помпа працює від джерела постійного струму напругою від 2,5 до 6 вольт та споживає від 0,4 до 1,5 ватт. Вона здатна перекачувати до 2 літрів рідини за хвилину (або 120 літрів за годину) на висоту до 1,1 метра. Корпус помпи виготовлений з герметичного пластику, що забезпечує довговічність та стійкість до корозії. Помпа може працювати як з водою, так і з маслом.

#### **1.3.2.5. RGB – світлодіод**

Конструкція RGB-світлодіода передбачає чотири виводи. Три виводи підключені до позитивних електродів світлодіодів червоного, зеленого та

синього кольорів. Четвертий вивід є загальним катодом для всіх трьох світлодіодів.



Рис. 1.32. RGB світлодіод

Щоб уникнути пошкодження світлодіода через надмірний струм, до кожного кольорового сегмента (червоного, зеленого та синього) RGB-світлодіода підключається резистор опором 270 Ом. Цей резистор встановлюється між катодом світлодіода і виводом мікроконтролера Arduino, який керує яскравістю відповідного кольору.

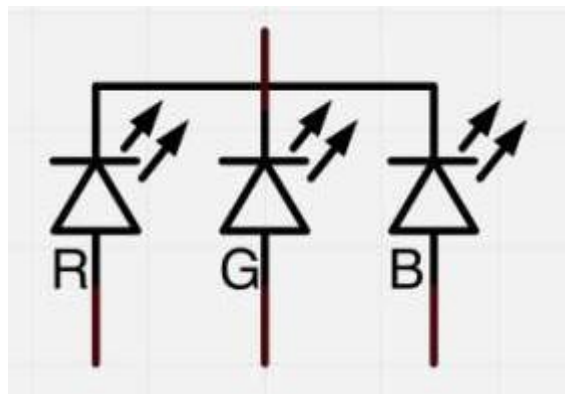


Рис.1.33. Умовне позначення світлодіодів на схемі

Існують два основних типи RGB-світлодіодів: із загальним катодом та із загальним анодом. У світлодіодах із загальним катодом (найпоширеніший тип), негативні виводи всіх кольорових елементів з'єднані разом. Для керування яскравістю кожного кольору необхідно подавати сигнал на відповідний позитивний вивід. У світлодіодах із загальним анодом, навпаки, позитивні виводи всіх кольорових елементів з'єднані разом, а для керування яскравістю використовують негативні виводи.

Принцип роботи RGB-світлодіодів базується на здатності людського ока сприймати різноманітність кольорів шляхом комбінування трьох основних: червоного, зеленого та синього [24]. У сітківці ока є три типи конусів, чутливих до кожного з цих кольорів. Змінюючи інтенсивність світіння червоного, зеленого та синього світлодіодів, можна отримати практично будь-який колір видимого спектру. Це явище називається адитивним синтезом кольору.

Аналогічний принцип використовується в сучасних дисплеях, зокрема, в рідкокристалічних. Кожен піксель такого дисплея складається з трьох субпікселів червоного, зеленого та синього кольору, розміщених дуже близько один до одного. Завдяки цьому, око сприймає їх як єдиний піксель, колір якого залежить від інтенсивності світіння кожного субпікселя [25].

Наприклад, якщо всі три кольорових елементи RGB-світлодіода світять з однаковою інтенсивністю, то результуючий колір буде білим. Якщо ж вимкнути синій елемент, а червоний і зелений світитимуть з однаковою інтенсивністю, то ми отримаємо жовте світло. Таким чином, змінюючи співвідношення інтенсивностей червоного, зеленого та синього компонентів, можна отримати широкий спектр кольорів.

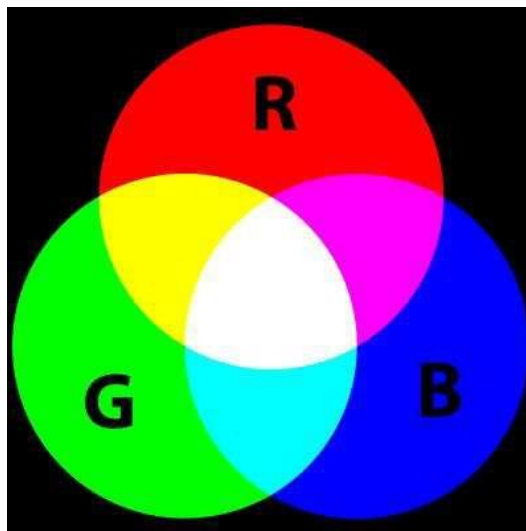


Рис. 1.34. Принцип змішування кольорів

Змінюючи інтенсивність світіння кожного з трьох кольорових елементів RGB-світлодіода (червоного, зеленого та синього), можна отримати безліч різних відтінків. Однак, повністю чорний колір, що відповідає повній відсутності світла, таким способом отримати неможливо. Найближчим до чорного буде стан,



коли всі світлодіоди вимкнені. Схема підключення RGB-світлодіода до мікроконтролера Arduino детально проілюстрована на рисунку 1.35.

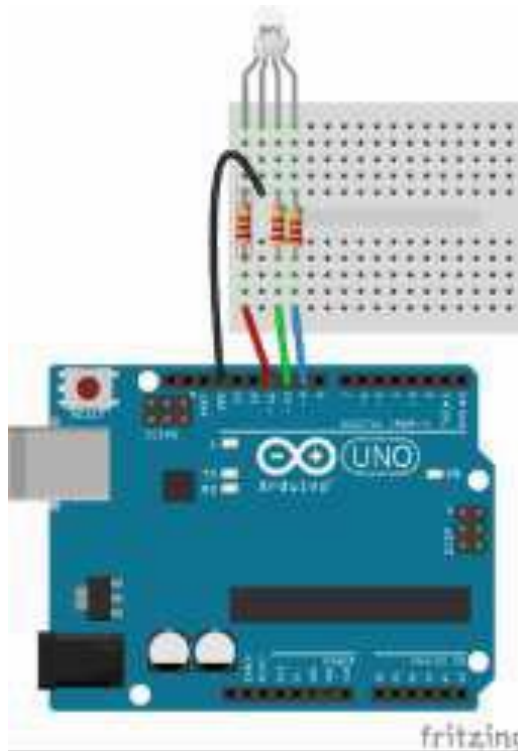


Рис. 1.35. Схема підключення RGB світлодіода до Ардуіно

#### 1.3.2.6. Модуль звуку

Для створення звукових сигналів в електронних пристроях на базі Arduino часто використовують модулі, відомі як зумери або п'єзодинаміки. Ці компоненти призначені для перетворення електричних імпульсів, поданих з мікроконтролера, на звукові хвилі. Завдяки своїй простоті та доступності, зумери широко застосовуються в побутовій техніці, іграшках та інших електронних пристроях, де необхідне звукове сповіщення.



Рис. 1.36. П'єзоелемент або зуммер

Конструктивно зумер являє собою металеву пластинку, на яку нанесено шар п'єзокераміки. Ці два елементи слугують електродами. Принцип роботи зумера заснований на п'єзоелектричному ефекті: під дією електричного поля п'єзокераміка деформується, викликаючи вібрацію металевої пластини. Результатом цих вібрацій є звук [26].

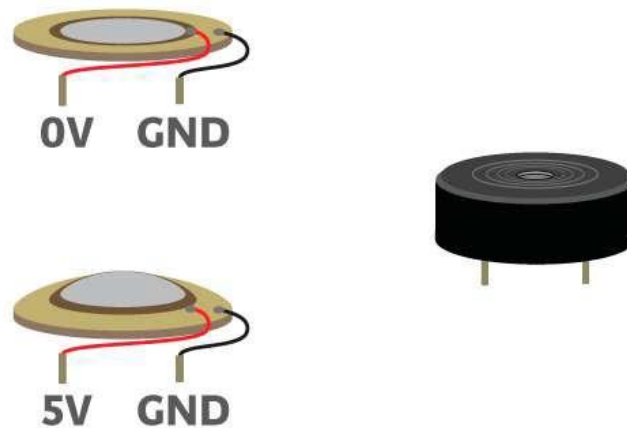


Рис. 1.37. Принцип дії зуммера

Зумери поділяються на два основних типи: активні та пасивні. Обидва типи працюють на принципі п'єзоелектричного ефекту, однак відрізняються за своєю схемотехнікою. Активні зумери мають вбудований генератор, що забезпечує фіксовану частоту звучання і, як правило, більшу гучність. Пасивні зумери вимагають зовнішнього джерела сигналу для генерації звуку, що дозволяє змінювати його частоту та інші параметри. Для підключення до мікроконтролерів типу Arduino найбільш зручними є готові модулі активних зумерів, які мають спрощену схему підключення. Типовий приклад такого модуля зображено на рисунку 1.38.



Рис. 1.38. Модуль звука для Ардуіно

Зумери, як електроакустичні перетворювачі, відзначаються значно простішою конструкцією порівняно зі звичними електромагнітними динаміками. Це спрощення робить зумери економічно вигіднішими у виробництві та експлуатації. Крім того, частота звукового сигналу, що генерується зумером, може бути легко змінена шляхом програмної модифікації управляючого пристрою.

Активні та пасивні зумери відрізняються за принципом роботи. Активні зумери мають вбудований генератор звукової частоти, що дозволяє їм самостійно генерувати звуковий сигнал при подачі напруги живлення. Пасивні зумери потребують зовнішнього джерела змінної напруги, яке задає параметри звукового сигналу. Як правило, таким джерелом слугує мікроконтролер, наприклад, Arduino. Активні зумери, завдяки своїй конструкції, забезпечують більшу гучність звучання. Типові параметри активних зумерів: робоча частота близько 2,5 кГц з допустимим відхиленням  $\pm 300$  Гц та напруга живлення в діапазоні 3,5-5 В.

Однією з ключових переваг активних зумерів є відсутність необхідності в складних програмних алгоритмах для генерації звукового сигналу. Це суттєво спрощує розробку програмного забезпечення для пристроїв, що використовують такі зумери. Крім того, ідентифікувати тип зумера (активний або пасивний) можна за допомогою вимірювання електричного опору між його виводами. Активні зумери, як правило, мають більший опір. Схема підключення активного зумера до мікроконтролера Arduino зображено на рисунку 1.39.

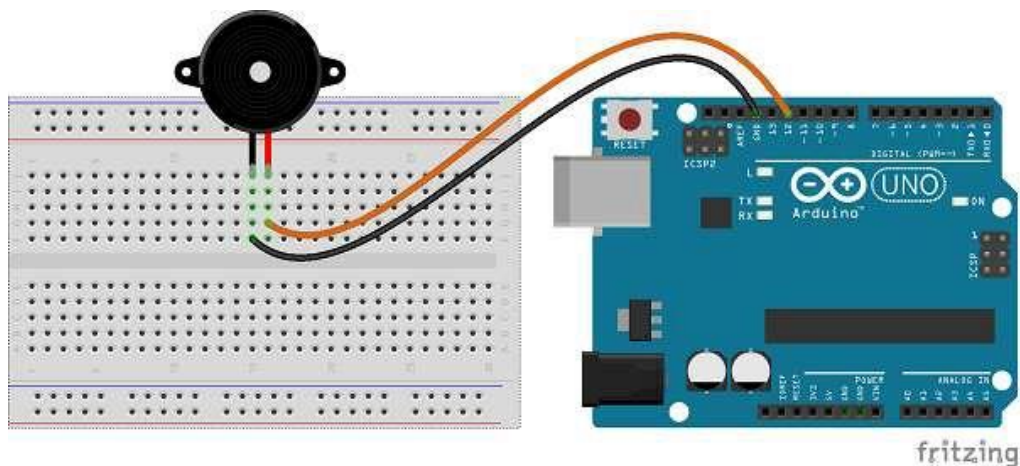


Рис. 1.39. Підключення зумера до Ардуіно

Електрична схема підключення п'єзoeлементa без супроводжуючих модулів зображена на рисунку 1.40.

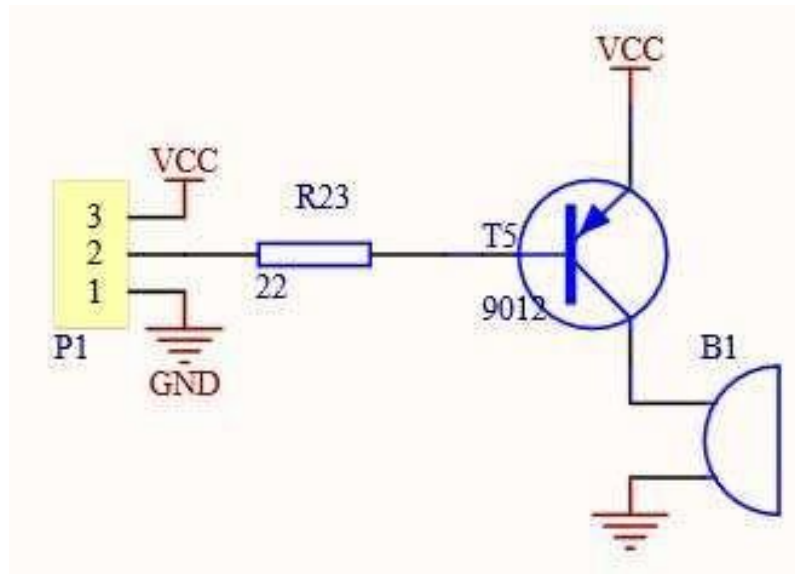


Рис. 1.40. Схема підключення п'єзoeлементa без супроводжуючих модулів

Зумер, як правило, має два виводи, які слід підключати з дотриманням полярності. Темний провід (часто чорний або коричневий) зазвичай відповідає загальному проводу (землі), а світлий (червоний) – до цифрового виходу мікроконтролера, що підтримує широтно-імпульсну модуляцію (ШІМ). Цей вибір не є випадковим: ШІМ дозволяє регулювати гучність звуку, змінюючи ширину імпульсів.

Цифровий вхід мікроконтролера Arduino використовується для зчитування стану логічних рівнів (0 або 1). До цього входу можна підключати різноманітні датчики, кнопки або резистори. При зміні напруги на вході, Arduino фіксує цю зміну і відповідним чином реагує на неї, наприклад, змінюючи стан виходів або виконуючи певну послідовність команд.

Правильне підключення зумера до мікроконтролера Arduino та розуміння принципу роботи цифрових входів є ключовими моментами при створенні електронних пристроїв, що видають звукові сигнали. Застосування ШІМ дозволяє досягти гнучкої регулювання гучності звуку, а використання цифрових входів – створювати інтерактивні системи, що реагують на зовнішні впливи.

### 1.3.2.7. Дисплей LCD 1602

Рідкокристалічні дисплеї LCD 1602 є поширеним вибором для виведення текстової інформації у багатьох електронних проєктах завдяки своїй доступності та широкому спектру модифікацій [27]. Однак, їхнє застосування з мікроконтролерами типу Arduino Uno або Nano має певні обмеження. Класична схема підключення LCD 1602 вимагає використання мінімум 6 цифрових виводів мікроконтролера, що може бути значним обмеженням для проєктів з обмеженою кількістю доступних портів. Це пов'язано з тим, що для керування кожним сегментом символів на дисплеї потрібен окремий сигнал керування.

Для подолання цього обмеження часто використовують LCD модулі з інтерфейсом I2C. Цей інтерфейс дозволяє підключити дисплей до мікроконтролера лише за допомогою двох проводів (SCL і SDA), суттєво зменшуючи кількість зайнятих портів. Це робить LCD дисплеї з інтерфейсом I2C більш універсальними та зручними у використанні з платформами Arduino.

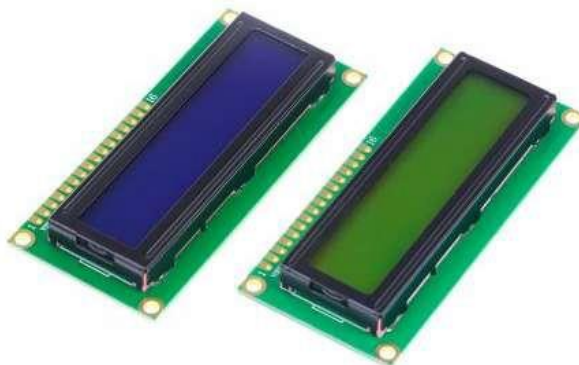


Рис. 1.41. Дисплей LCD 1602

Кожен з виводів рідкокристалічного дисплея відіграє критичну роль у його функціонуванні:

- **GND (земля):** слугує точкою відліку для електричних потенціалів у схемі.
- **5 В (живлення):** забезпечує необхідну напругу для роботи електроніки дисплея.
- **Контраст:** дозволяє регулювати яскравість зображення шляхом зміни напруги на цьому виводі.
- **RS (Register Select):** визначає, чи буде наступна передана інформація інтерпретована як команда (наприклад, встановити курсор) чи як дані для

відображення на екрані.

- **RW (Read/Write):** використовувався в старих моделях дисплеїв для читання даних з дисплея. В сучасних моделях зазвичай заземлений.
- **Enable (E):** Сигнал, який інформує дисплей про те, що дані на лініях даних готові для прийому.
- **DB0-DB7 (Data Bits):** група з 8 ліній, за якими передаються біти даних, що визначають, які сегменти рідких кристалів мають бути увімкнені для формування зображення.
- **A (анод) та K (катод):** виводи для підключення зовнішнього джерела живлення для підсвічування дисплея.



Рис. 1.42. Виводи LCD 1602

Рідкокристалічний дисплей (LCD) має наступні технічні характеристики:

- Тип відображення: символів, з можливістю завантаження користувацьких символів.
- Підсвічування: світлодіодне, що забезпечує яскраве та рівномірне підсвічування символів.
- Контролер: HD44780, що відповідає за керування роботою дисплея.
- Живлення: 5 вольт постійного струму.
- Розмір дисплея: 16 символів по ширині та 2 рядки по висоті.
- Робоча температура: від -20°C до +70°C.



- Температура зберігання: від  $-30^{\circ}\text{C}$  до  $+80^{\circ}\text{C}$ .
- Кут огляду: 180 градусів, що забезпечує гарну видимість з різних кутів.

Стандартна схема приєднання монітора безпосередньо до мікроконтролера Ардуіно без I2C зображена на рисунку 1.43.

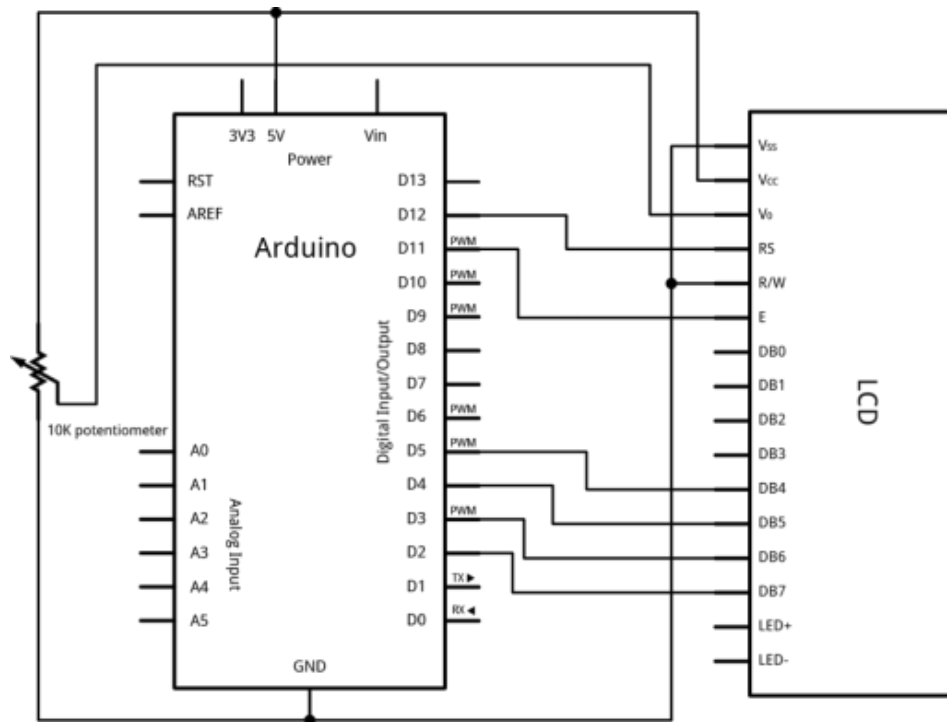


Рис. 1.43. Схема підключення дисплея до Ардуіно без I2C

I2C (Inter-Integrated Circuit) – це популярний послідовний інтерфейс, який використовується для зв'язку між різними електронними компонентами в системах. Однією з головних переваг I2C є його простота: для підключення багатьох пристроїв достатньо лише двох ліній – лінії даних (SDA) та лінії тактування (SCL). Це дозволяє суттєво економити порти мікроконтролера, що особливо важливо для платформ з обмеженими ресурсами, таких як Arduino. Крім того, I2C підтримує підключення великої кількості пристроїв до однієї шини, що робить його зручним для створення складних електронних систем.

Ще одним важливим плюсом є широка підтримка I2C сучасними мікроконтролерами. Більшість платформ, включаючи Arduino, мають вбудовані модулі для роботи з I2C, що значно спрощує процес розробки.

Однак, як і будь-яка технологія, I2C має свої обмеження. Одним з них є ємнісне обмеження шини. Наявність занадто великої кількості пристроїв або

довгих проводів може призвести до спотворення сигналів і, як наслідок, до нестабільної роботи системи. Крім того, програмування мікроконтролера для роботи з великою кількістю різних пристроїв, підключених до I2C, може бути досить складним завданням. Особливо це стосується ситуацій, коли виникають збої в роботі шини, оскільки їх локалізація може зайняти багато часу.

Підсумовуючи, I2C є потужним інструментом для розробки електронних пристроїв, але його використання вимагає розуміння його особливостей та обмежень. Перед вибором I2C для конкретного проєкту необхідно ретельно оцінити такі фактори як кількість підключених пристроїв, довжина шини та складність програмного забезпечення.



Рис. 1.44. Модуль i2c для LCD 1602

Для швидкого та зручного підключення рідкокристалічного дисплея до плати Arduino рекомендується використовувати готові модулі, які мають вбудовану підтримку протоколу I2C. Це значно спрощує процес підключення, оскільки вимагає лише під'єднання двох проводів (SDA та SCL) до відповідних pinів на Arduino. Для забезпечення підсвічування дисплея може знадобитися додаткове джерело живлення. Контрастність зображення можна легко налаштувати за допомогою вбудованого потенціометра. Схема підключення такого модуля до Arduino детально проілюстрована на рисунку 1.45.



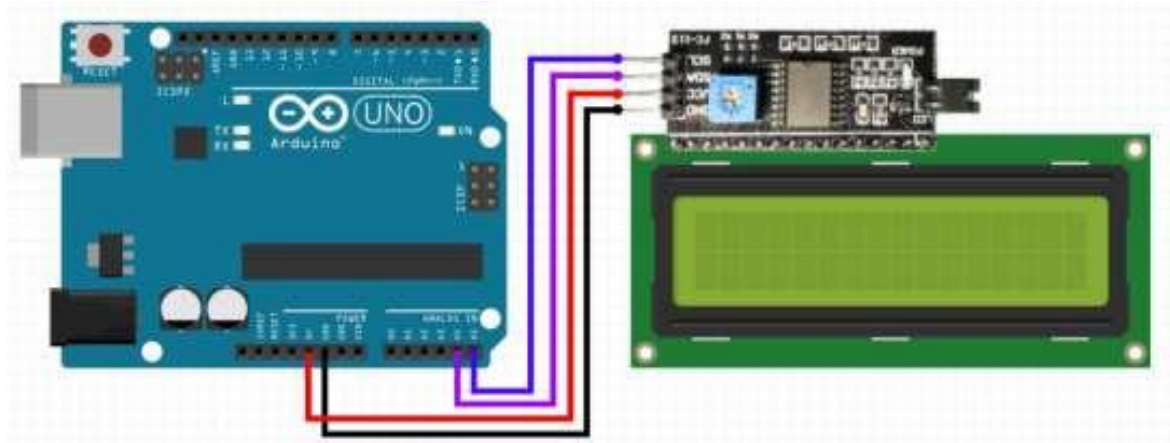


Рис. 1.45. Підключення LCD екрану до Ардуіно по I2C

Рідкокристалічний дисплей з інтерфейсом I2C підключається до плати за допомогою чотирьох проводів: два для передачі даних (SDA та SCL) і два для живлення (GND та VCC). Конкретніше, вивід GND дисплея з'єднується з загальним проводом (GND) на платі, вивід VCC підключається до 5-вольтового живлення, а лінії даних SDA та SCL підключаються до відповідних аналогових портів A4 та A5 на платі.

### 1.3.3. Пристрої вводу та виводу інформації - модуль Bluetooth

Для забезпечення бездротового зв'язку між мікроконтролером Arduino та іншими пристроями часто використовується модуль Bluetooth [28]. Взаємодія між цими компонентами здійснюється за допомогою послідовного інтерфейсу. Модуль Bluetooth, як правило, має чотири основні виводи, функції яких детально описані нижче та проілюстровані на рисунку 1.46.



Рис. 1.46. Модуль Bluetooth

Вивід VCC призначений для підключення до джерела живлення з напругою 5 В, яке, як правило, забезпечується мікроконтролером Arduino. Вивід GND використовується для заземлення модуля. Для передачі даних від модуля

Bluetooth до мікроконтролера використовується вивід TXD, який слід підключити до виводу RX мікроконтролера. У випадку Arduino Uno це вивід з номером 0. Навпаки, для прийому даних з мікроконтролера використовується вивід RXD модуля Bluetooth, який підключається до виводу TX мікроконтролера (вивід 1 у Arduino Uno).

Слід звернути увагу на те, що вхід TX модуля Bluetooth, як правило, розрахований на напругу 3,3 В. Оскільки вихід TX мікроконтролера Arduino видає напругу 5 В, необхідно вжити заходів для зниження рівня сигналу до безпечного для модуля значення. Для цього часто використовується подільник напруги, який складається з двох резисторів. Такий подільник дозволяє зменшити напругу сигналу, що подається на вхід TX модуля Bluetooth, до необхідних 3,3 В. Схема такого підключення детально проілюстрована на рисунку 1.47.

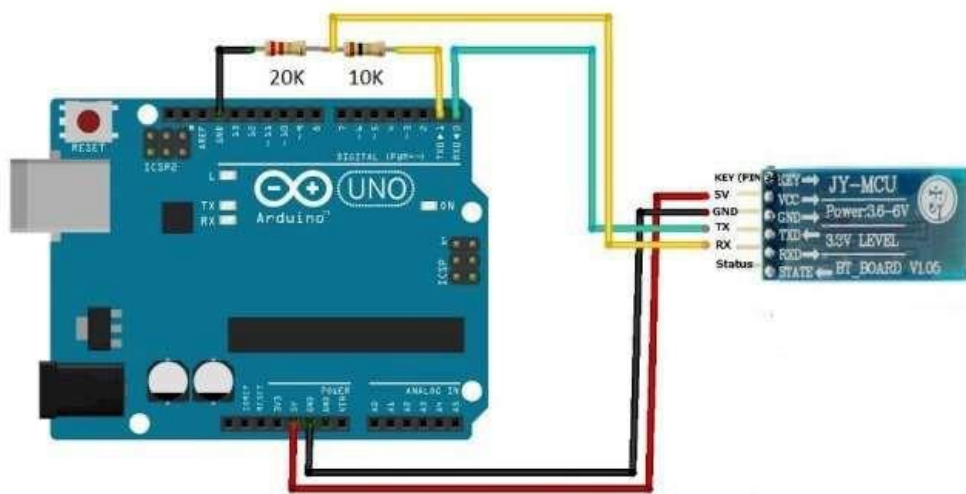
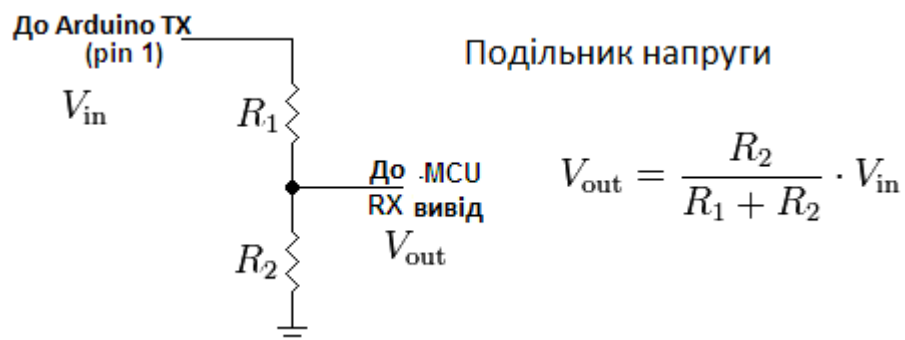


Рис. 1.47. Схема підключення модуля Bluetooth до Ардуіно

На схемі використані резистори 20кОм і 10 кОм, на основі рівняння для подільника напруги:



$R_1 = 10\text{кОм}$ ,  $R_2 = 20\text{кОм}$ ,  $V_{in} = 5\text{ В}$ . Для  $V_{out}$  отримуємо:  $V_{out} = 5\text{ В} \times$

$$20\text{кОм} / (20\text{кОм} + 10\text{кОм}) = 5\text{В} \times 20\text{кОм}/30\text{кОм} = 5\text{В} \times 2/3 = 10\text{В}/3 = 3.33 \text{ В}.$$

Можна використовувати будь-яку іншу комбінацію резисторів, але R2 повинно мати в два рази менший опір, ніж R1.

### **Висновки до розділу**

Проведений в розділі аналіз технологій та компонентів, необхідних для створення системи контролюваного нагрівання води на базі платформи Arduino, дозволяє зробити висновок про високий потенціал цієї платформи для розробки інтелектуальних систем управління мікрокліматом. Порівняльний аналіз мікроконтролерів Arduino Uno та Mega, а також детальний опис периферійних пристроїв, надають розробникам необхідну інформацію для створення ефективних та надійних систем. Отримані результати можуть бути використані як основа для подальших досліджень у галузі автоматизації систем опалення та гарячого водопостачання, а також для розробки прототипів інноваційних рішень.

## **РОЗДІЛ 2. ТЕОРЕТИЧНІ ОСНОВИ ТА ПРАКТИЧНІ АСПЕКТИ РОЗРОБКИ ПРИСТРОЮ**

Для розробки алгоритмів управління функціонуванням пристрою було обрано інтегроване середовище розробки Arduino IDE [29]. Arduino IDE є потужним інструментом, що дозволяє інженерам і дослідникам створювати програмне забезпечення для мікроконтролерів Arduino. Це середовище забезпечує зручний інтерфейс для написання, компіляції та завантаження програмного коду.

Скетчі, написані в Arduino IDE, використовують спрощений синтаксис мови C++, що робить їх доступними для широкого кола користувачів. Після написання скетчу, середовище розробки виконує його перевірку на синтаксичні помилки, перетворює код на мову C++ та компілює його в машинний код, готовий до завантаження в мікроконтролер. Цей процес автоматизації значно спрощує розробку програмного забезпечення для Arduino.

### **2.1. Використання бібліотек для розширення функціональності програмного забезпечення**

Для ефективної розробки програмного забезпечення для системи контрольованого нагрівання води було використано механізм бібліотек. Бібліотека Arduino представляє собою набір функцій та структур даних, що зберігається окремо від основного коду програми [30]. Цей підхід дозволяє модуляризувати програмний код, спрощуючи його структуру та підтримку.

Використання бібліотек значно спрощує роботу з різноманітними периферійними пристроями, такими як датчики, актуатори, дисплеї тощо. Бібліотеки надають готові функції для взаємодії з цими пристроями, що дозволяє розробнику зосередитися на розв'язанні конкретного завдання, не заглиблюючись у деталі низькорівневого програмування.

Для реалізації функціональності системи контрольованого нагрівання води були використані такі бібліотеки: LiquidCrystal\_I2C.h, OneWire.h, ServoSmooth.h, GyverButton.h, що дозволило значно скоротити час розробки та підвищити надійність програмного забезпечення.

### 2.1.1. LiquidCrystal\_I2C.h

Для візуалізації інформації в розробленій системі було використано рідкокристалічний дисплей, керований за допомогою бібліотеки LiquidCrystal\_I2C [31]. Ця бібліотека є частиною стандартного набору Arduino IDE і забезпечує зручний інтерфейс для взаємодії з дисплеями, що підключені за протоколом I2C.

Бібліотека LiquidCrystal\_I2C надає широкий спектр функцій для управління дисплеєм, включаючи ініціалізацію, очищення, встановлення курсора, виведення тексту, створення користувацьких символів та керування підсвічуванням. Детальний опис основних функцій наведено нижче:

- Ініціалізація дисплея: функція `begin(cols, rows, [char_size])` використовується для початкової настройки дисплея, включаючи встановлення кількості стовпців і рядків, а також розміру символів.
- Очищення та позиціонування курсора: функції `clear()`, `home()`, `setCursor(col, row)` дозволяють очистити дисплей, встановити курсор у початкову позицію або перемістити його в довільну точку дисплея.
- Керування відображенням: функції `display()`, `noDisplay()`, `blink()`, `noBlink()`, `cursor()`, `noCursor()` дозволяють керувати видимістю дисплея, миганням курсора та підкресленням символів.
- Прокручування дисплея: функції `scrollDisplayLeft()` та `scrollDisplayRight()` дозволяють прокручувати вміст дисплея вліво або вправо.
- Керування напрямком прокручування тексту: функції `leftToRight()` та `rightToLeft()` дозволяють встановити напрямок прокручування тексту.
- Створення користувацьких символів: функція `createChar(num, array)` дозволяє створити власні символи та зберігати їх у пам'яті дисплея.
- Виведення тексту: функція `print(text)` використовується для виведення текстової інформації на дисплей.
- Керування підсвічуванням: функції `backlight()`, `noBacklight()` та

setBacklight(flag) дозволяють керувати підсвічуванням дисплея.

Для підключення дисплея до мікроконтролера Arduino необхідно вказати його адресу на шині I2C, кількість стовпців і рядків. Приклад коду для ініціалізації дисплея:

```
#include <Wire.h>

#include <LiquidCrystal_I2C.h> LiquidCrystal_I2C lcd (address, col, row); void
setup () {
    lcd.init ();
}
```

Де параметр:

- address: Адреса дисплея на шині I2C - 0x27 або 0x3F;
- col: кількість стовпців, які реалізовані у дисплея;
- row: кількість рядків, які реалізовані у дисплея.

Використання бібліотеки LiquidCrystal\_I2C значно спрощує процес розробки програмного забезпечення для керування рідкокристалічними дисплеями, дозволяючи розробникам зосередитися на вирішенні більш складних завдань.

### **2.1.2. OneWire.h**

Інтерфейс 1-Wire [32] відрізняється простотою реалізації, оскільки для обміну даними вимагає лише одного сигнального проводу та заземлення. Однією з унікальних особливостей цього інтерфейсу є можливість паразитного живлення підключених пристроїв без використання додаткового джерела живлення. Сигнальний провід одночасно слугує для передачі даних та живлення пристроїв, що мають достатню ємність для накопичення енергії. Така конструкція дозволяє спростити схему підключення та зменшити кількість необхідних компонентів.

Для початку сеансу обміну даними на шині 1-Wire необхідно виконати процедуру скидання (reset). Майстер (контролер) ініціює скидання шляхом встановлення логічного нуля на сигнальному проводі протягом мінімум 480 мкс. Після цього майстер відпускає лінію, дозволяючи їй перейти в невизначений

Якщо на шині присутні пристрої, готові до обміну даними, вони реагують на сигнал скидання. Для підтвердження готовності пристрої активізують свій вихід, тимчасово опускаючи рівень сигналу на лінії до логічного нуля на проміжок часу від 60 до 240 мкс. Таким чином, майстер, прочитавши стан лінії після скидання, може визначити, чи присутні на шині пристрої, готові до взаємодії.

- Скидання (reset): активний низький сигнал тривалістю не менше 480 мкс, що ініціює сеанс обміну.
- Підтвердження скидання: активний низький сигнал від пристроїв тривалістю 60-240 мкс, що свідчить про їхню готовність до обміну.
- Резистор підтяжки: забезпечує встановлення лінії даних у логічну одиницю в неактивному стані.



63

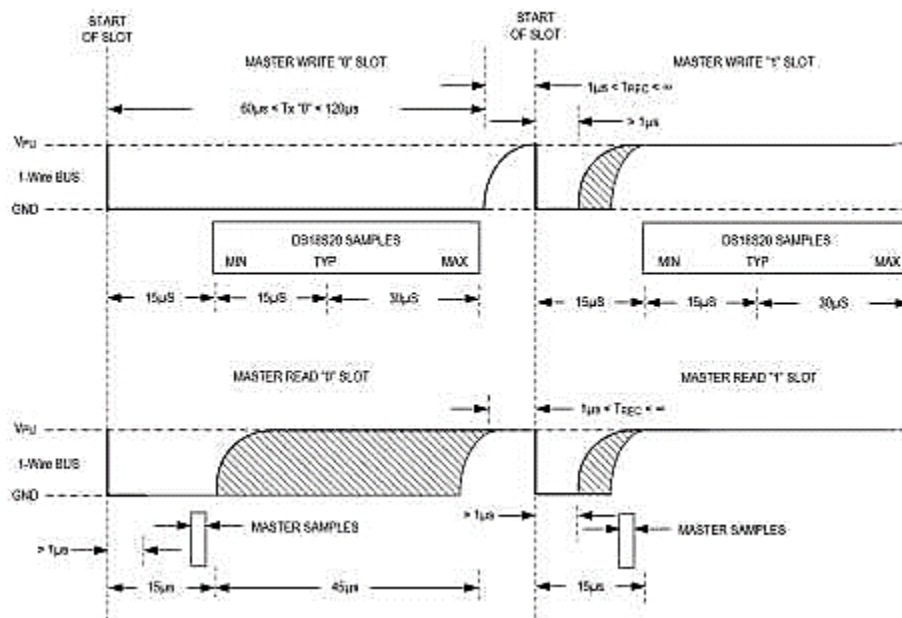


Рис. 2.2. Передача даних 1-Wire [32]

### 2.1.3. ServoSmooth.h

Для реалізації плавного руху сервоприводів у проєкті було використано бібліотеку ServoSmooth [33], яка є розширенням стандартної бібліотеки Servo.h. Головною відмінністю ServoSmooth є вбудований алгоритм, що забезпечує плавне змінення положення валу сервопривода за заданим часом, шляхом поступового збільшення або зменшення ширини імпульсу ШІМ.

Бібліотека ServoSmooth використовує метод tick(), який необхідно викликати періодично в циклі програми. Внутрішній таймер цього методу розраховує необхідні зміни ширини імпульсу ШІМ для плавного переміщення сервопривода до заданої позиції.

Бібліотека ServoSmooth надає широкий спектр функцій для управління сервоприводами, включаючи:

- ініціалізація: функція attach() дозволяє підключити сервопривод до певного цифрового виводу мікроконтролера та встановити початкові параметри, такі як мінімальна та максимальна ширина імпульсу ШІМ, а також цільовий кут повороту.
- управління рухом: функції setTarget() та setTargetDeg() дозволяють встановити нову цільову позицію для сервопривода відповідно в мікросекундах або градусах. Функції setSpeed() та setAccel()



дозволяють налаштувати швидкість та прискорення руху сервопривода.

- отримання інформації про поточний стан: функції `getCurrent()` та `getCurrentDeg()` дозволяють отримати поточну позицію сервопривода відповідно в мікросекундах або градусах.
- керування роботою: функції `start()` та `stop()` дозволяють запустити або зупинити рух сервопривода.

Для використання бібліотеки `ServoSmooth` необхідно включити її у свій проєкт та створити об'єкт класу `ServoSmooth`. Далі, за допомогою функції `attach()` необхідно підключити сервопривод до відповідного цифрового виводу мікроконтролера та встановити необхідні параметри. Для запуску руху сервопривода слід викликати функцію `start()`, а для зупинки - функцію `stop()`. В циклі програми необхідно періодично викликати функцію `tick()` для виконання алгоритму керування.

#### **2.1.4. GyverButton.h**

Бібліотека `GyverButton.h` [34] призначена для спрощення роботи з тактовими кнопками в середовищі Arduino. Вона дозволяє ефективно обробляти різноманітні події, пов'язані з натисканням кнопки, такі як одиночне натискання, утримання, подвійне та потрійне натискання.

Функціональні можливості бібліотеки:

- універсальність: бібліотека підтримує роботу як з нормально замкнутими, так і з нормально розімкнутими кнопками, а також дозволяє використовувати різні схеми підключення з підтягувальними резисторами.
- фільтрація дрижання контактів: вбудований механізм дебаунсингу дозволяє усунути помилкові спрацювання, спричинені дрижанням контактів кнопки.
- налаштовані параметри: можливість налаштування часу дебаунсингу, таймауту утримання, інтервалу між повторними натисканнями та інших параметрів.

- різноманітні події: бібліотека дозволяє відстежувати такі події, як натискання, відпускання, утримання, одиночне, подвійне та потрійне натискання, а також зміну стану кнопки з заданим кроком за часом.
- просте використання: інтуїтивно зрозумілий інтерфейс та набір функцій, що дозволяють легко інтегрувати бібліотеку в будь-який Arduino проєкт.

### **Основні класи та функції**

**Клас *GButton*:** основний клас для роботи з кнопкою. При створенні об'єкта цього класу необхідно вказати номер цифрового виводу, до якого підключена кнопка, а також додаткові параметри, такі як тип кнопки (нормально замкнута або нормально розімкнута) та схема підключення (з підтягувальним резистором або без).

Функції для налаштування:

`setDebounce()`: встановлює час дебаунсингу.

`setTimeout()`: встановлює таймаут утримання.

`setStepTimeout()`: встановлює інтервал між зміною стану при утриманні кнопки.

`setType()`: встановлює тип кнопки.

`setTickMode()`: вибирає режим опитування кнопки (ручний або автоматичний).

Функції для отримання інформації:

`isPress()`, `isRelease()`, `isClick()`, `isHelded()`, `isHold()`, `isSingle()`, `isDouble()`, `isTriple()`, `hasClicks()`, `getClicks()`, `isStep()`: Повертають інформацію про стан кнопки та події, що відбулися.

Функція `tick()`: виконує опитування стану кнопки та оновлює внутрішні змінні бібліотеки.

## **2.2. Види інтерфейсів для взаємодії периферійних пристроїв**

Послідовний зв'язок є однією з найпоширеніших методів взаємодії між мікроконтролером та периферійними пристроями. На відміну від паралельного зв'язку, при якому дані передаються одночасно по декількох лініях, послідовний

зв'язок передбачає послідовну передачу бітів даних по єдиній лінії. Такий підхід дозволяє значно зменшити кількість необхідних проводів, спростити схему підключення та знизити вартість пристроїв.

Для успішної передачі даних у послідовних протоколах необхідно забезпечити синхронізацію між передавачем і приймачем. Синхронізація може здійснюватися за допомогою спеціальних сигналів синхронізації або за рахунок використання стартового і стопового бітів, які додаються до кожного байту даних.

Незважаючи на послідовну передачу бітів, обробка даних всередині пристроїв здійснюється паралельно за допомогою регістрів. Це дозволяє досягти високої швидкості обробки інформації.

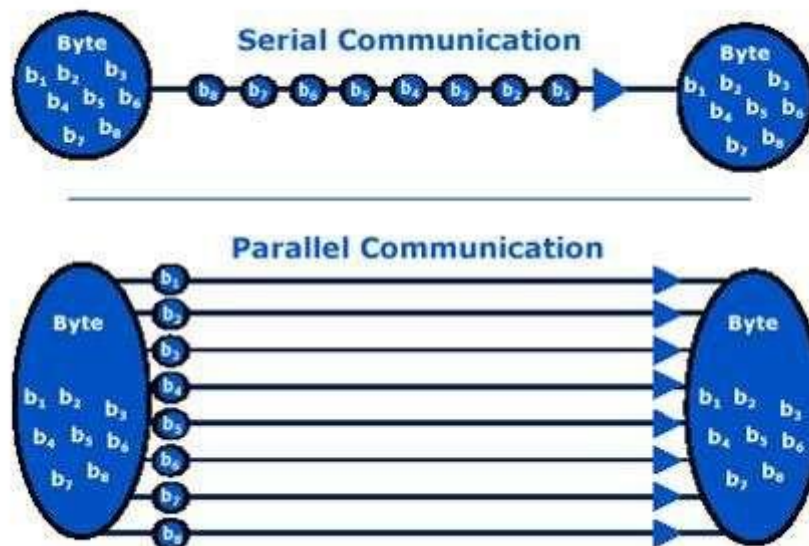


Рис. 2.3. Послідовний та паралельний зв'язок [35]

Послідовний зв'язок поділяється на синхронний та асинхронний в залежності від способу синхронізації передавача і приймача [35]. У синхронних системах передавач і приймач використовують спільне джерело тактових імпульсів, що забезпечує точну синхронізацію процесу передачі даних. На відміну від синхронних систем, асинхронні системи не потребують загального тактового генератора. Кожна зі сторін використовує власний незалежний тактовий генератор, що дозволяє досягти більшої гнучкості в конструюванні систем.

### 2.2.1. SPI (Serial Peripheral Interface)

Інтерфейс SPI (Serial Peripheral Interface) є синхронним послідовним протоколом, який широко використовується для зв'язку між мікроконтролером та

периферійними пристроями [36]. Основними лініями SPI є MOSI (Master Out Slave In), MISO (Master In Slave Out), SCK (Serial Clock) та SS (Slave Select).

**MOSI:** лінія, по якій головний пристрій (майстер) передає дані підлеглому пристрою (слейву).

**MISO:** лінія, по якій підлеглий пристрій передає дані головному.

**SCK:** тактова лінія, яка синхронізує передачу даних. Генератором тактових імпульсів зазвичай виступає головний пристрій.

**SS:** лінія вибору підлеглого пристрою. За допомогою цієї лінії головний пристрій вибирає конкретний підлеглий пристрій для взаємодії.

В SPI-інтерфейсі завжди присутній один головний пристрій і один або декілька підлеглих пристроїв. Головний пристрій ініціює всі передачі даних, встановлюючи необхідний стан на лінії SS та передаючи дані по лінії MOSI. Підлегли пристрої можуть лише відповідати на запити головного пристрою, передаючи дані по лінії MISO. Синхронізація передачі даних здійснюється за допомогою тактових імпульсів, які генеруються головним пристроєм і передаються по лінії SCK.

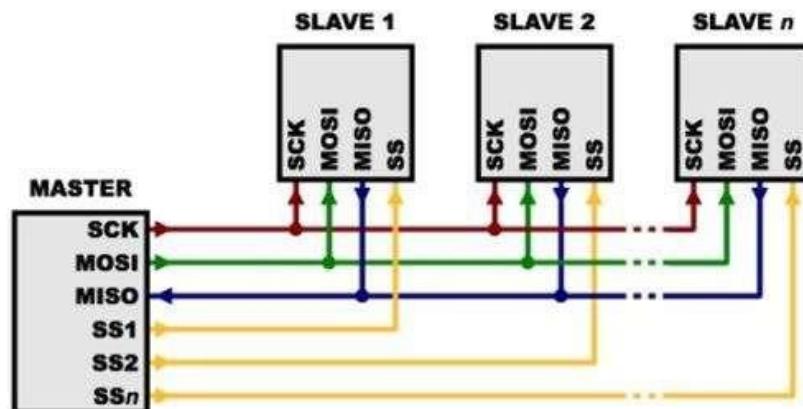


Рис. 2.4. Принцип підключення інтерфейсу SPI [36]

Для ефективної роботи інтерфейсу SPI мікроконтролер використовує спеціальні регістри, які керують процесом передачі та прийому даних. До основних регістрів SPI належать:

**SPDR (SPI Data Register):** 8-бітний регістр слугує для тимчасового зберігання байта даних, який необхідно передати або прийняти. При передачі дані завантажуються в цей регістр, а при прийомі – зчитуються з нього.

**SPSR (SPI Status Register):** реєстр містить біти стану, які відображають поточний стан модуля SPI. За допомогою цих бітів можна визначити, чи завершена передача або прийом даних, чи виникли якісь помилки тощо.

**SPCR (SPI Control Register):** у цьому реєстрі зберігаються біти, які дозволяють налаштувати роботу модуля SPI. За допомогою цих бітів можна вибрати режим роботи (майстер або раб), встановити швидкість передачі даних, ввімкнути або вимкнути різні функції.

#### *Переваги інтерфейсу SPI:*

1. Синхронність: завдяки використанню тактового сигналу, SPI забезпечує надійнішу передачу даних порівняно з асинхронними протоколами.

2. Гнучкість: дозволяє підключати до одного головного пристрою (майстера) декілька підлеглих пристроїв (слейвів).

3. Висока швидкість: забезпечує високу швидкість передачі даних, що робить його придатним для багатьох застосувань.

#### *Недоліки інтерфейсу SPI:*

1. Складність підключення багатьох пристроїв. Для підключення кожного додаткового підлеглого пристрою потрібна окрема лінія вибору (SS).

2. Централізоване управління. Весь процес комунікації контролюється головним пристроєм. Підлеглі пристрої не можуть ініціювати передачу даних самостійно.

### **2.2.2. I2C (Inter-Integrated Circuit) або двопровідний інтерфейс**

Інтерфейс I2C (Inter-Integrated Circuit) є універсальним послідовним протоколом, який використовується для з'єднання різних інтегральних схем в електронних пристроях [37]. На відміну від SPI, I2C використовує лише два провідники: лінію даних (SDA) та лінію тактування (SCL), що робить його більш економічним з точки зору кількості використовуваних виводів.

В основі роботи протоколу I2C лежить концепція майстер-слейв. Головний пристрій (майстер) ініціює всі операції обміну даними, генеруючи сигнали на лініях SDA та SCL. Підлеглі пристрої (слейви) відповідають на запити майстра.

Кожен пристрій на шині I2C має унікальну адресу, за якою майстер може звернутися до нього.

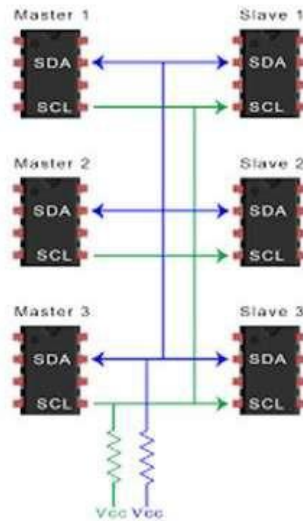


Рис. 2.5. Принцип взаємодії I2C інтерфейсу

Процес обміну даними починається з генерації стартового сигналу майстром. Далі майстер передає адресу підлеглого пристрою, з яким бажає взаємодіяти. Після підтвердження адреси підлеглим, починається передача або прийом даних. Кожен байт даних супроводжується бітом підтвердження (ACK), який підлеглий пристрій встановлює для підтвердження правильного прийому байта. Після завершення передачі даних майстер генерує сигнал зупинки.

*Переваги інтерфейсу I2C:*

1. проста реалізація: використання лише двох проводів спрощує підключення пристроїв;
2. гнучкість: можливість підключення декількох пристроїв до однієї шини;
3. мультимайстерна архітектура: дозволяє підключити декілька майстер-пристроїв до однієї шини.

*Недоліки інтерфейсу I2C:*

1. Швидкість передачі даних по I2C нижча, ніж по SPI.
2. Протокол I2C має більш складну структуру пакетів даних порівняно з SPI.

Інтерфейс I2C широко використовується в різних електронних пристроях, таких як: мікроконтролери, сенсори, пам'ять EEPROM, РК-дисплеї, годинники реального часу.

Інтерфейс I2C є універсальним і простим у використанні протоколом для з'єднання різних інтегральних схем. Його гнучкість та низька вартість роблять його популярним вибором для багатьох застосувань. Однак, при виборі між I2C та SPI необхідно враховувати такі фактори, як необхідна швидкість передачі даних, кількість підключених пристроїв та складність системи в цілому.

### **2.2.3. UART**

Протоколи UART (Universal Asynchronous Receiver-Transmitter) та USART (Universal Synchronous Asynchronous Receiver-Transmitter) є фундаментальними для послідовного зв'язку в електроніці [38]. Хоча їх назви схожі, функціональні можливості цих протоколів відрізняються.

Ключова відмінність полягає в підтримуваних режимах передачі даних. UART спроектований виключно для асинхронного послідовного зв'язку, тоді як USART підтримує як асинхронний, так і синхронний режими. Це означає, що USART надає більшу гнучкість у застосуванні.

В асинхронному режимі, характерному для обох протоколів, передавач і приймач використовують незалежні тактові генератори. Синхронізація досягається за рахунок спеціальних стартових і стопових бітів, які додаються до кожного байту даних. Швидкість передачі даних, вимірювана в бодах, визначає кількість бітів, переданих за одну секунду, і повинна бути однаковою для обох пристроїв для забезпечення успішного обміну даними.

Обидва протоколи, як правило, використовують точку-точка топологію, тобто для прямого обміну даними потрібна пара пристроїв, з'єднаних лініями передачі (Tx) та прийому (Rx). Це обмежує кількість пристроїв, які можуть одночасно взаємодіяти без додаткових мережевих елементів.

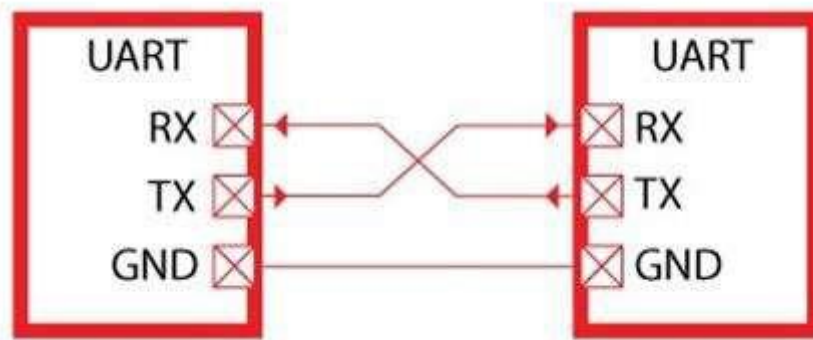


Рис. 2.6. Принцип взаємодії UART інтерфейсу

Для синхронного режиму використовується додатковий вивід ХСК. Імпульси тактування генеруються пристроєм, які відправляють дані в цей час.

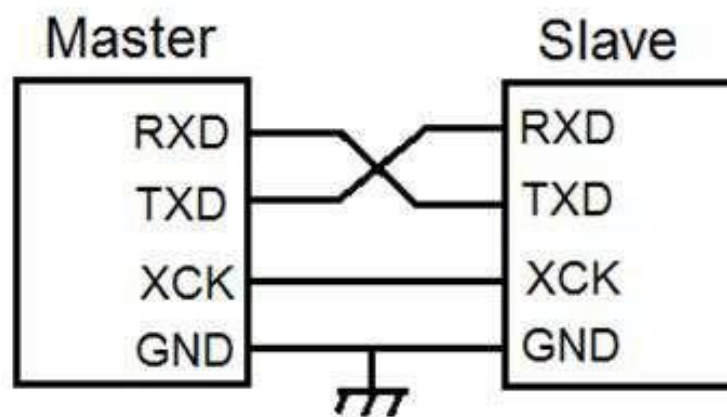


Рис. 2.7. UART інтерфейсу в синхронному режимі

UART/USART забезпечують як синхронний, так і асинхронний послідовний зв'язок, що розширює сферу їх застосування. Варіабельність швидкості передачі даних робить їх придатними для різних пристроїв та умов. Проте, обмеження топології з'єднання "точка-точка" обмежує масштабованість систем, побудованих на основі цих протоколів.

### 2.3. Вдосконалення мікроконтролера Arduino Uno

#### Модернізація мікроконтролера Arduino Uno

Існуюча конфігурація мікроконтролера Arduino Uno має обмежену кількість портів введення/виведення. Для розширення функціональності пристрою пропонується збільшити кількість доступних входів та виходів. Зокрема, планується додати додаткові порти для підключення кнопок та інших периферійних пристроїв. Крім того, буде вжито заходів для усунення паразитних



шумів (брязкоту контактів), що можуть негативно впливати на стабільність роботи системи."

Обраний мікроконтролер Arduino Uno має обмежену кількість входів для зчитування сигналу й виводів для відправки. Тому доречно вдосконалити мікроконтролер так, щоб збільшити кількість ввідів та виводів, підключити більшу кількість кнопок для комутації сигналу, а також усунути брязкіт контактів.

### 2.3.1. Брязкіт контактів - програмне і апаратне усунення

Брязкіт контактів – це поширена проблема в електромеханічних системах, яка проявляється у вигляді багаторазових коротких замикань та розривів електричного кола після замикання контактів. Причиною цього явища є пружні властивості матеріалів контактів, які призводять до їхнього «підстрибування». Брязкіт контактів може призвести до помилкових спрацювань електронних схем та зниження їх надійності. Для усунення цієї проблеми використовуються спеціальні програмні та апаратні методи.

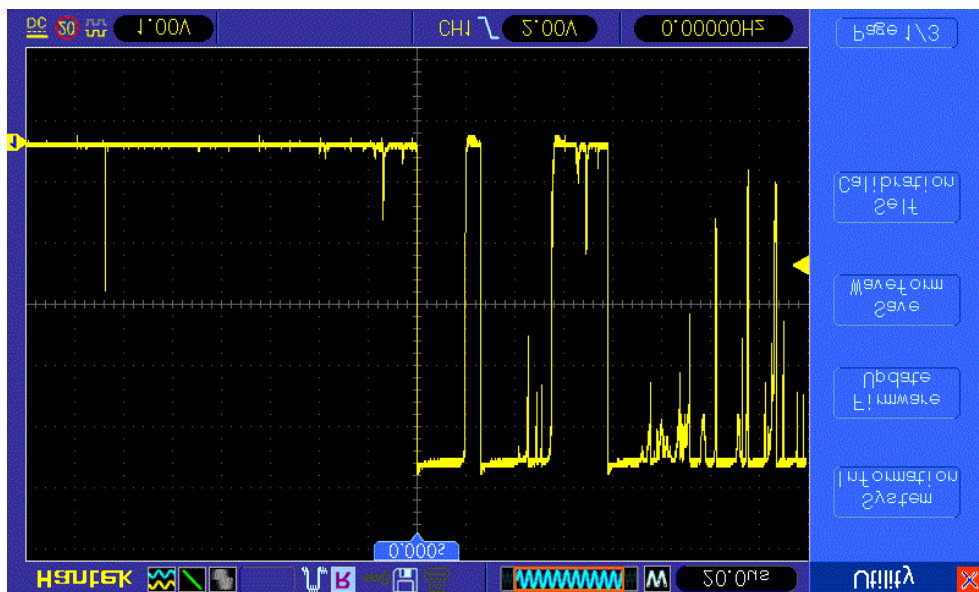


Рис. 2.8. Брязкіт під час натискання кнопки, зафіксований осцилографом

#### Програмне усунення брязкоту

Одним із поширених методів усунення брязкоту контактів є програмне дебаунсинг. Сутність методу полягає у введенні затримки після зміни стану контакту, що дозволяє контактним групам стабілізуватися перед подальшим опрацюванням сигналу.

Типовий алгоритм програмного дебаунсінгу передбачає порівняння попереднього стану контакту з поточним. У випадку розбіжності вводиться затримка, після якої повторно зчитується стан контакту. Якщо отримані значення збігаються, вважається, що брязкіт усунено.

Наведений у завданні лістинг функції `debounce()` є класичним прикладом реалізації програмного дебаунсінгу в середовищі Arduino IDE. Ця функція приймає на вході номер піну, до якого підключена кнопка, і повертає логічне значення, що відповідає стабілізованому стану кнопки.

### Апаратне усунення брязкоту

Апаратні методи усунення брязкоту контактів, як правило, більш ефективні та надійні порівняно з програмними. Це пов'язано з тим, що апаратні рішення не потребують витрат обчислювальних ресурсів мікроконтролера та забезпечують більш швидку реакцію на зміну стану контактів.

Одним із поширених апаратних методів є використання тригера Шмітта. Тригер Шмітта має два пороги спрацювання: верхній та нижній. Коли на вході тригера напруга перевищує верхній поріг, на виході з'являється логічна одиниця, і для скидання виходу напруга має знизитися нижче нижнього порогу. Завдяки гістерезису тригера Шмітта, випадкові коливання напруги, спричинені брязкотом контактів, не призводять до багаторазових перемикань виходу.

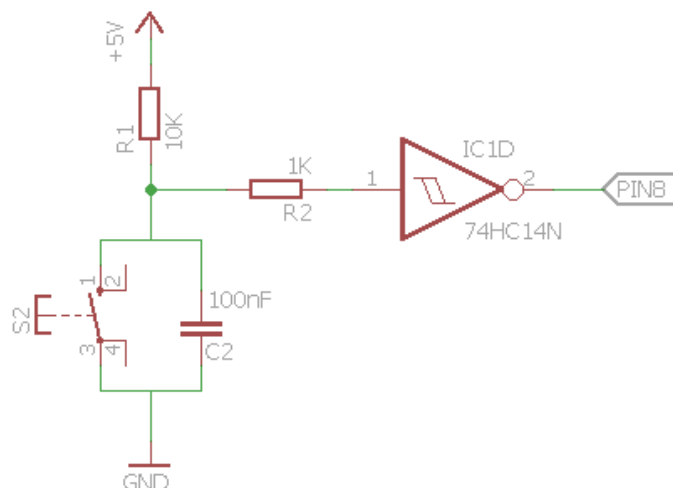


Рис. 2.9. Схема електрична принципова кола для усунення брязкоту

Замикання кола ініціює процес заряджання конденсатора. Як видно з графіка на рисунку 2.10, напруга на обкладках конденсатора зростає експоненціально, описуючись формулою (1), і досягає сталого значення 5 В.

$$U_c = U(1 - e^{-\frac{t}{RC}}) \quad (1)$$

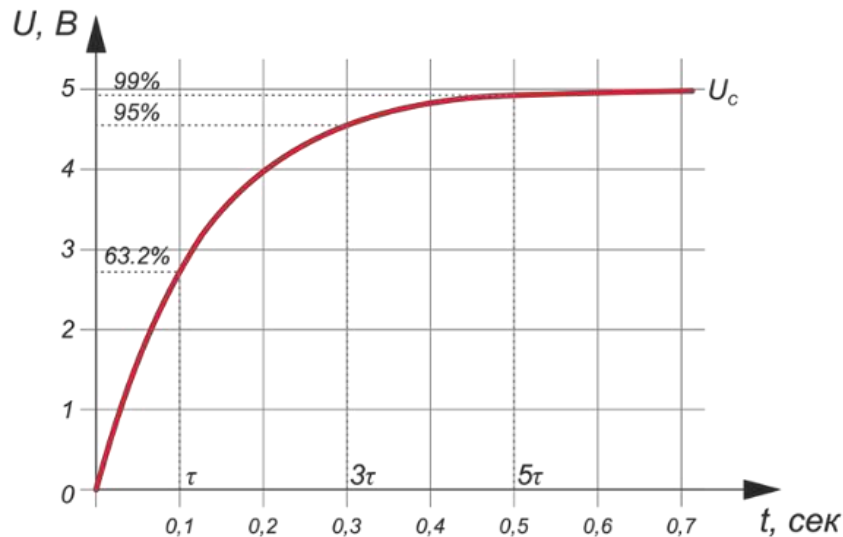


Рис. 2.10. Графік заряду конденсатора

У вимкненому стані на цифровому вході мікроконтролера підтримується напруга 5 В завдяки підтягувальному резистору. При натисканні кнопки, конденсатор розряджається через кнопку, тимчасово знижуючи напругу на вході мікроконтролера. Після відпускання кнопки, конденсатор починає заряджатися через резистор R2 зі сталою часу  $\tau = RC$ . Цей процес займає приблизно  $5\tau$ . Додатковий резистор R2 уповільнює розряд конденсатора, зменшуючи вплив брязкоту контактів. Інвертувальний тригер Шмітта 74НС14N перетворює плавну зміну напруги на конденсаторі в чіткий цифровий сигнал, який фіксується осцилографом (рис. 2.11).

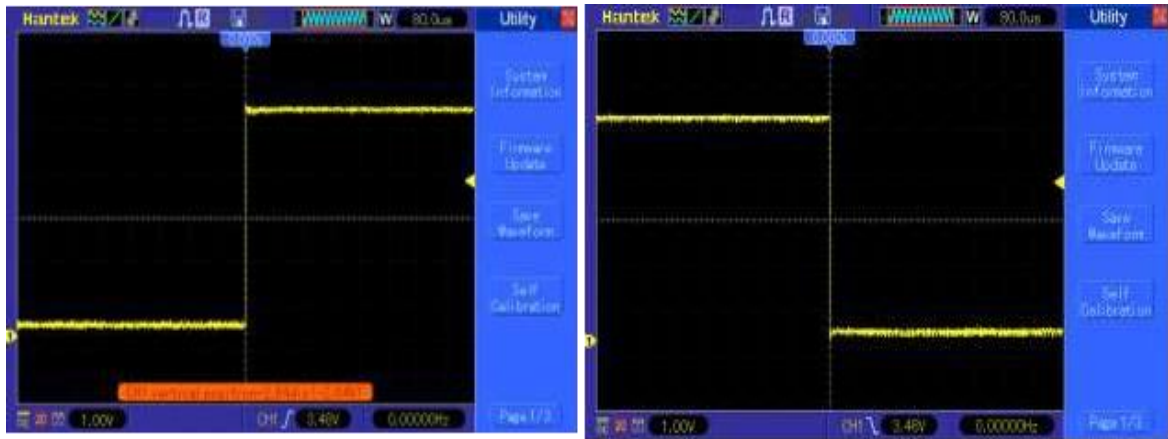


Рис. 2.11. Відсутність брязкоту у разі натискання кнопки, зафіксований осцилографом

### 2.3.2. Підключення декілька кнопок до одного аналогового входу

З метою оптимізації використання ресурсів мікроконтролера пропонується метод підключення декількох кнопок до єдиного аналогового входу. Суть методу полягає у формуванні унікального рівня напруги на аналоговому вході для кожної натиснутої кнопки за рахунок резистивного поділу напруги. Мікроконтролер, зчитуючи цей рівень напруги та порівнюючи його з еталонними значеннями, може однозначно визначити, яка саме кнопка була натиснута. Для реалізації цього методу використовуються два основні варіанти схемного рішення: послідовне та паралельне з'єднання резисторів.

#### Схема №1 – резистивне-послідовне підключення

Принципово схема резистивно-послідовного підключення може бути виконано за двома варіантами (рис.2.12):

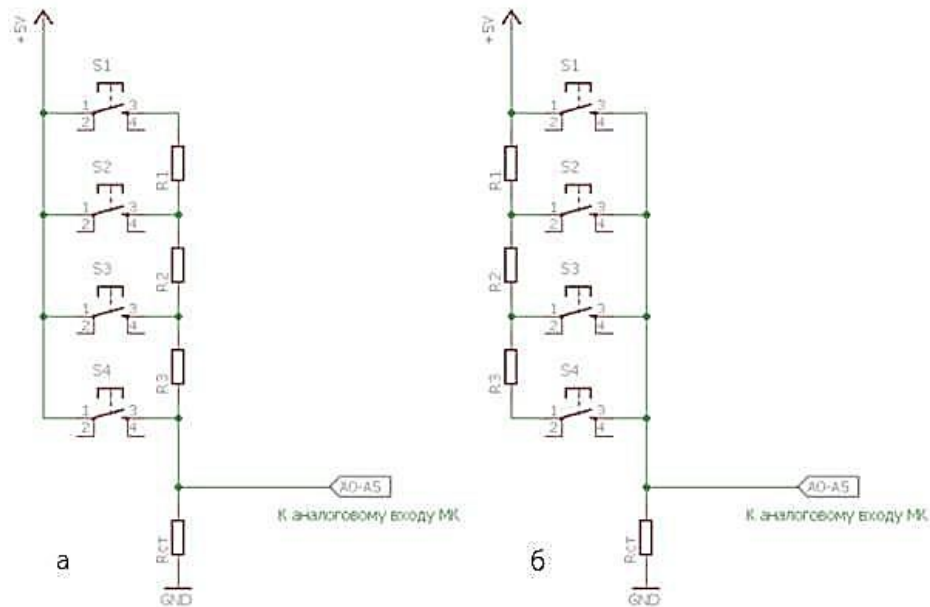


Рис. 2.12. Схеми електричні принципові комутації резисторів:

а – пряме підключення; б – зворотнє підключення

Основна відмінність між схемами, зображеними на рис. 2.12а та 2.12б, полягає в тому, як розподіляється напруга при натисканні різних кнопок. У схемі 2.12а, при натисканні кнопки  $S_4$ , на аналоговий вхід подається максимальна напруга 5 В, а при натисканні кнопки  $S_1$  - мінімальна, оскільки напруга ділиться на всіх резисторах кола. В схемі 2.12б ситуація протилежна: максимальна напруга відповідає кнопці  $S_1$ . Обидві схеми є прикладами послідовного з'єднання резисторів, загальний опір якого дорівнює сумі опорів всіх резисторів ( $R_{\text{заг}} = R_1 + R_2 + \dots + R_n$ ). При натисканні будь-якої кнопки утворюється типовий резистивний дільник напруги, як показано на рис. 2.13.

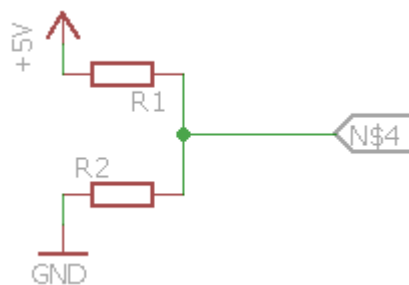


Рис. 2.13. Резистивний дільник напруги

Залежність напруги та опорів:  $U_{\text{вих}} = U_{\text{вх}} \times R_2 / (R_1 + R_2)$

З формули випливає, що залежність вихідної напруги  $U_{\text{вих}}$  від опорів  $R_1$  і  $R_2$  не лінійна, а гіперболічна (рис 2.14). Для прикладу:  $R_1 = 10 \text{ кОм}$ ,  $R_2 = x$ .

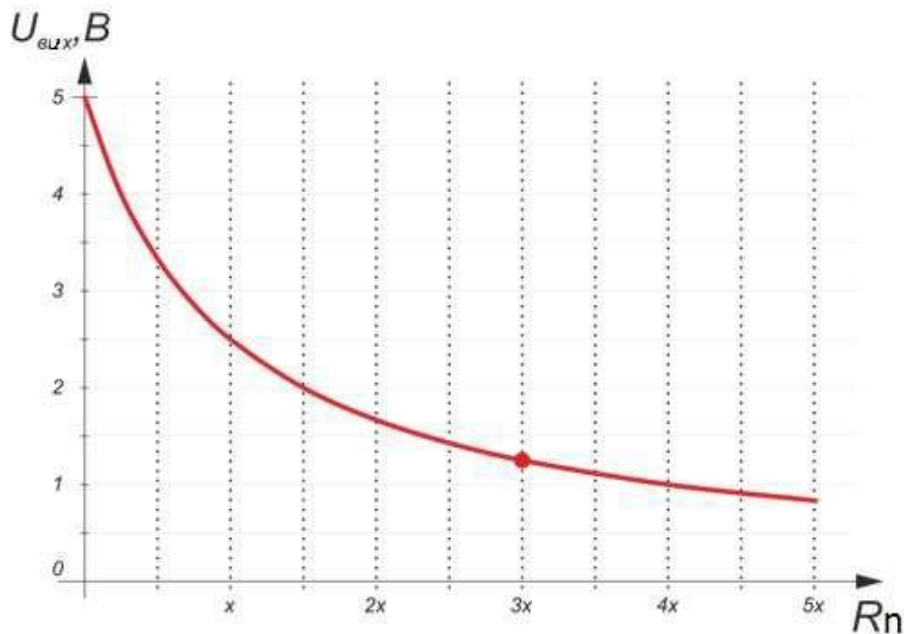


Рис. 2.14. Залежність напруги від опорів

Вибір номіналів резисторів для формування дискретних рівнів напруги при підключенні декількох кнопок до одного аналогового входу мікроконтролера є важливим завданням для забезпечення точності та ефективності системи.

Існує два основних підходи до розрахунку номіналів резисторів.

Формування рівномірних інтервалів напруги передбачає підбір таких номіналів резисторів, щоб при натисканні кожної кнопки на аналоговому вході формувалася напруга, яка відрізняється від попередньої на фіксовану величину. Такий метод є особливо ефективним при кількості кнопок, що дорівнює степеню двійки ( $2^n$ , де  $n \geq 2$ ), оскільки дозволяє використовувати бітовий зсув для швидкого визначення натиснутої кнопки. Завдяки цьому мінімізується вплив похибки аналого-цифрового перетворення (АЦП) та допусків резисторів.

Використання резисторів однакового номіналу спрощує розрахунок схеми, але може призвести до нерівномірного розподілу напруги на аналоговому вході. Для компенсації цієї неточності можна використовувати програмні методи, такі як таблиці перетворення або інтерполяція.

Як показано на рисунку 2.15, при формуванні рівномірних інтервалів напруги, номінали резисторів, що відповідають низьким рівням напруги, зростають експоненціально з збільшенням кількості кнопок. Це може призвести

до труднощів у підборі стандартних номіналів резисторів і збільшення загального опору схеми.

Для усунення впливу похибки АЦП при використанні нерівномірних інтервалів напруги, можна застосувати програмну корекцію, тобто ввести таблицю перетворення, яка пов'язує значення, отримані від АЦП, з відповідними номерами кнопок.

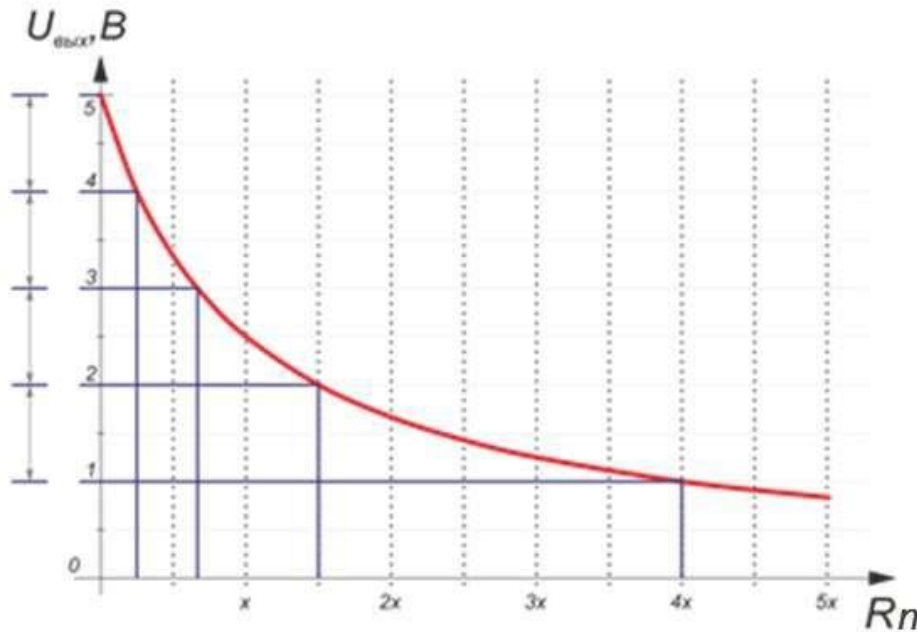


Рис. 2.15. Залежність з однаковими інтервалами для напруги

Для створення рівномірної шкали напруги при роботі з  $k$  кнопками, використовується формула, яка дозволяє розрахувати необхідні номінали резисторів. При цьому за основу береться номінал підтягувального резистора  $R_{ст}$ , що спрощує процес розрахунку для різних конфігурацій схеми.

$$R_n = \frac{U_{вх} \times R_{ст}}{u \times (k - n)} - R_{ст} - \sum_{i=1}^{n-1} R_i, \text{ при } n \in \mathbb{Z} \mid 1 \leq n \leq k - 1$$

$u = \frac{U_{вх}}{k}$  — крок дискретності вихідної напруги

Оскільки до першої кнопки резистор не під'єднано, можна вважати, що до неї підключено резистор з нульовим опором. Таким чином, кількість резисторів, які фактично впливають на розподіл напруги, менша за загальну кількість кнопок на одиницю ( $n \leq k-1$ ). Коли перша кнопка натиснута, на аналоговий вхід

подається максимальна напруга, рівна напрузі живлення ( $U_{вх} = 5 \text{ В}$ ).

Розглянемо конкретний приклад розрахунку номіналів резисторів для схеми з 5 кнопками та 4 резисторами. Припустимо, що опір підтягувального резистора  $R_{ст}$  дорівнює 10 кОм.

$$u = \frac{U_{ex}}{k} = \frac{5\text{В}}{5} = 1\text{В}$$

$$R_1 = \frac{5\text{В} \times 10\text{кОм}}{1\text{В} \times (5 - 1)} - 10\text{кОм} - 0\text{кОм} = 2,5\text{кОм}$$

$$R_2 = \frac{5\text{В} \times 10\text{кОм}}{1\text{В} \times (5 - 2)} - 10\text{кОм} - (2,5\text{кОм}) = 4,167\text{кОм}$$

$$R_3 = \frac{5\text{В} \times 10\text{кОм}}{1\text{В} \times (5 - 3)} - 10\text{кОм} - (2,5\text{кОм} + 4,167\text{кОм}) = 8,333\text{кОм}$$

$$R_4 = \frac{5\text{В} \times 10\text{кОм}}{1\text{В} \times (5 - 4)} - 10\text{кОм} - (2,5\text{кОм} + 4,167\text{кОм} + 8,333\text{кОм}) = 25\text{кОм}$$

Для даної схеми доцільно використовувати резистори номіналом близько 2,5 кОм, 4,167 кОм, 8,333 кОм та 25 кОм. Точний допуск резисторів не є критичним.

Як альтернативу можна застосувати резистори однакового номіналу, меншого за опір підтягувального резистора. В такому випадку найбільші зміни напруги спостерігаються при натисканні перших кнопок. Діапазон змін напруги визначається формулою, що залежить від значень  $R_{ст}$  та  $R$ :

$$U_{вих_n} = U_{вх} * \frac{R_{ст}}{R * n + R_{ст}}$$

При такому способі підключення спостерігається зменшення роздільної здатності за напругою. Для уникнення цього явища, особливо коли необхідна висока точність вимірювань, рекомендується збільшувати номінали резисторів, починаючи з тих, що відповідають найменшим значенням напруги. Це дозволить збільшити різницю між сусідніми рівнями напруги, як показано на рисунку 2.16.



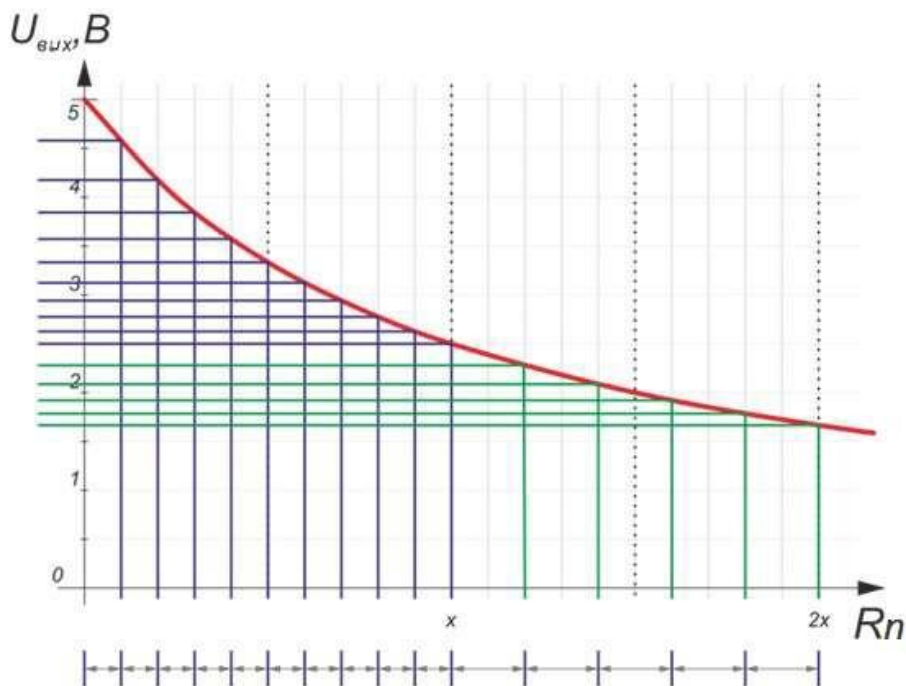


Рис. 2.16. Залежність з однаковими номіналами резисторів

Ключова особливість резистивно-послідовної схеми полягає в тому, що вона розпізнає натискання тільки однієї кнопки. Незалежно від того, скільки кнопок натиснуто одночасно, схема визначає лише ту, яка підключена через найменший опір. (рис. 2.17)

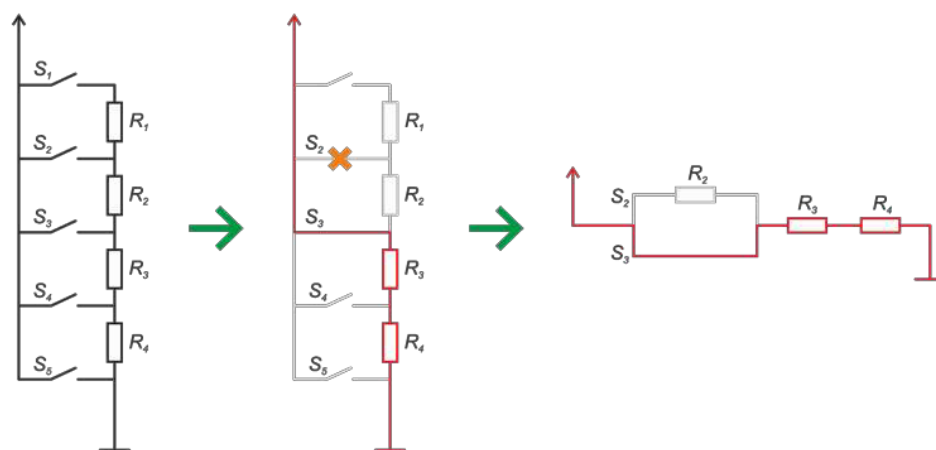


Рис. 2.17. Приклад реалізації комутації сигналу

Резистивно-послідовне підключення є одним із поширених методів формування напругових рівнів у електронних схемах.

Така схема має ряд переваг, які роблять її привабливою для використання в багатьох застосуваннях:

- передбачувана поведінка: робота схеми базується на простих законах Ома, що забезпечує високу стабільність та передбачуваність результатів.
- порівняно легкі розрахунки: розрахунок номіналів резисторів для отримання необхідних рівнів напруги є відносно простим завданням, яке може бути виконане за допомогою базових знань з електротехніки.
- передбачуваний результат при одночасному натисканні кількох кнопок: навіть якщо кілька кнопок будуть натиснуті одночасно, схема завжди видасть сигнал, що відповідає кнопці з найменшим опором, що робить поведінку схеми легко прогнозованою.
- гнучкість реалізації: існує два основних підходи до реалізації резистивно-послідовної схеми: з рівномірним кроком зміни вихідної напруги та з однаковими значеннями опорів резисторів. Кожен з цих підходів має свої переваги та може бути обраний залежно від конкретних вимог до системи.
- Разом з тим, резистивно-послідовне підключення має і свої недоліки:
- накопичення похибки: похибки у значеннях опорів резисторів можуть призвести до неточностей у вихідних напругах, особливо при використанні великої кількості резисторів.
- складність розрахунків для рівномірної дискретності: досягнення рівномірної дискретності значень напруги при великій кількості рівнів може вимагати складних розрахунків і підбору нестандартних номіналів резисторів.
- обмеження діапазону напруги: зазвичай, резистивно-послідовні схеми використовуються для формування напруг в діапазоні до 1 В. Для отримання більших значень напруги можуть знадобитися додаткові каскади підсилення.

- принципний порядок розташування резисторів: порядок розташування резисторів у схемі впливає на її функціонування. Зміна порядку може призвести до зміни вихідних напруг.

## **Схема №2 - резистивне-паралельне підключення**

Пропонована схема підключення кнопок характеризується індивідуальним заданням рівня вихідної напруги для кожної кнопки за рахунок підбору відповідного номіналу резистора. Розрахунок номіналів резисторів здійснюється на основі принципів резистивно-послідовного з'єднання з використанням різних значень опорів.

Варто зазначити, що при одночасному натисканні кількох кнопок відбувається паралельне з'єднання відповідних резисторів [47], що призводить до формування нового, індивідуального значення вихідної напруги. Таким чином, схема дозволяє генерувати множину дискретних рівнів напруги, кожен з яких відповідає певній комбінації натиснутих кнопок.

$$\frac{1}{R_{\text{общ}}} = \frac{1}{R_1} + \frac{1}{R_2} + \dots + \frac{1}{R_n}$$

або

$$R_{\text{общ}} = \frac{1}{\frac{1}{R_1} + \frac{1}{R_2} + \dots + \frac{1}{R_n}}$$

Схема ефективно розпізнає одночасне натискання невеликої кількості кнопок. Однак, при збільшенні кількості кнопок, передбачення поведінки схеми ускладнюється через зростання кількості можливих комбінацій. Розрахунок значень опорів для цієї схеми аналогічний розрахунку для типової резистивно-послідовної схеми, але з однією важливою відмінністю: кожен наступний опір розраховується незалежно від попередніх, без необхідності корекції.

$$R_n = \frac{U_{\text{вх}} \times R_{\text{ст}}}{U_{\text{вих}_n}} - R_{\text{ст}}$$

Розглянемо приклад схеми з трьома комутаціями, яка повинна генерувати напруги 1 В, 2,5 В та 4 В при напрузі живлення 5 В та номіналі підтягувального резистора 10 кОм. Для визначення необхідних номіналів резисторів виконуємо

розрахунок за такою схемою:

$$R_1 = \frac{5\text{В} \times 10_{\text{КОМ}}}{1\text{В}} - 10_{\text{КОМ}} = 40_{\text{КОМ}}$$

$$R_2 = \frac{5\text{В} \times 10_{\text{КОМ}}}{2,5\text{В}} - 10_{\text{КОМ}} = 10_{\text{КОМ}}$$

$$R_3 = \frac{5\text{В} \times 10_{\text{КОМ}}}{4\text{В}} - 10_{\text{КОМ}} = 2,5_{\text{КОМ}}$$

Унікальна особливість даної схеми полягає в тому, що при одночасному натисканні кількох кнопок аналогово-цифровий перетворювач (АЦП) мікроконтролера фіксує нове, інтегральне значення напруги. Це дозволяє значно збільшити кількість можливих комбінацій в порівнянні з випадком, коли розглядається лише натискання однієї кнопки. Наприклад, для схеми з трьома кнопками можливі такі комбінації: 1+2, 1+3, 2+3 та 1+2+3. Загальна кількість можливих комбінацій для  $n$  кнопок визначається за формулою числа сполучень  $C_n^k$  [48].

$$C_n^k = \frac{n!}{(n-k)! \times k!}$$

Оскільки схема не розглядає ситуації, коли натиснута лише одна кнопка (це базовий стан), та допускає одночасне натискання будь-якої кількості кнопок, загальна кількість можливих комбінацій визначається шляхом підсумовування кількості сполучень по дві кнопки, по три кнопки і так далі.

$$C_{n_{\text{обш}}} = C_n^2 + C_n^3 + \dots + C_n^n$$

$$C_{n_{\text{обш}}} = \sum_{k=2}^n \frac{n!}{(n-k)! \times k!}$$

Розглянемо приклад з п'ятьма кнопками. Крім п'яти окремих натискань, можливі різні комбінації одночасного натискання декількох кнопок. Для визначення загальної кількості комбінацій почнемо з підрахунку комбінацій по дві кнопки: 1+2, 1+3, 1+4, 1+5, 2+3 і так далі.

$$C_5^2 = \frac{5!}{(5-2)! \times 2!} = 10$$

Кількість комбінацій натиснень 3 кнопок (1+2+3, 1+2+4, 1+2+5, 2+3+4, тощо):

$$C_5^3 = \frac{5!}{(5-3)! \times 3!} = 10$$

Кількість комбінацій для натиснень 4 кнопок (1+2+3+4, 1+2+3+5, тощо):

$$C_5^4 = \frac{5!}{(5-4)! \times 4!} = 5$$

Одже кількість одночасних натискань 5 кнопок - 1. Разом:

$$C_{n_{\text{общ}}} = 10 + 10 + 5 + 1 = 26$$

Таким чином, крім п'яти основних станів (коли натиснута лише одна кнопка), для нашої схеми отримуємо додаткові 26 комбінацій. Аналогічно, для системи з шістьма кнопками, крім шести основних станів, існує 57 унікальних комбінацій одночасного натискання.

Переваги резистивно-паралельного підключення очевидні: простота підбору необхідних значень опорів, незалежність роботи схеми від порядку розташування резисторів та можливість розширення функціоналу за рахунок різноманітних інтерпретацій комбінацій натиснутих кнопок.

Однак, дана схема має і свої обмеження. При збільшенні кількості кнопок понад п'ять, передбачити поведінку схеми при одночасному натисканні кількох кнопок стає досить складно.

### **2.3.3. Збільшення аналогових входів завдяки мультиплексору CD4051**

CD4051 - це інтегральна мікросхема, яка виконує функції 8-канального аналогового мультиплексора/демультиплексора на основі CMOS технології. Вона є ефективним рішенням для розширення кількості аналогових входів у вашому проєкті. Завдяки використанню лише одного аналогового входу і трьох цифрових сигналів управління, ця мікросхема дозволяє отримати доступ до восьми різних аналогових сигналів по чергово.

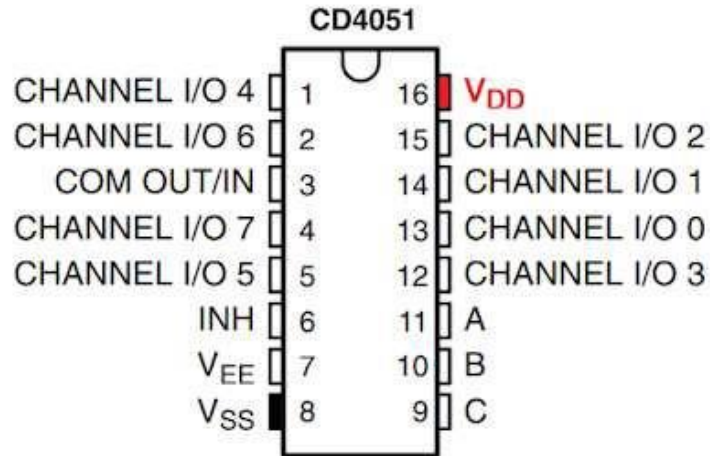


Рис. 2.18. Мультиплексор CD4051 [49]

Мультиплексор - це електронний пристрій, який дозволяє підключати до одного виходу декілька джерел сигналу по черзі [49]. Демультиплексор виконує протилежну функцію: він розподіляє сигнал з одного входу на кілька виходів. Мікросхема CD4051 об'єднує в собі можливості як мультиплексора, так і демультиплексора. Принцип її роботи детально проілюстровано на рисунку 2.19.

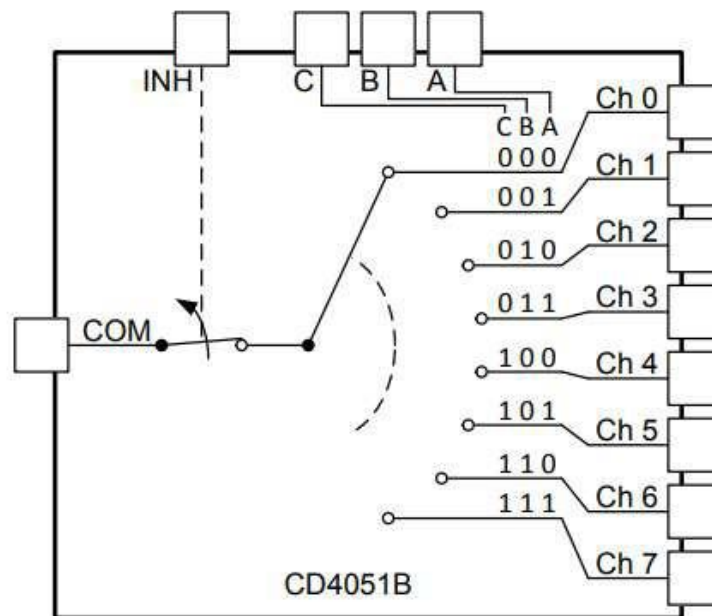


Рис. 2.19. Функціональна схема CD4051 [49]

Основна функція мікросхеми CD4051 полягає в комутації аналогових сигналів. Вона дозволяє підключати до одного виходу (COM) різні входи (0-7) залежно від комбінації сигналів на входах управління (A, B, C). Для того, щоб комутація відбулася, на вході заборони (INH) має бути низький логічний рівень.

Практичне застосування CD4051 є досить широким. Наприклад, її можна

використовувати для керування 8 світлодіодами, зчитування даних з 8 потенціометрів або для розширення кількості підключених до Arduino п'єзoelementів з метою створення MIDI барабанної установки.

Один із типових прикладів використання CD4051 - це підключення 8 фоторезисторів до Arduino. Завдяки цій мікросхемі, можна по чергово зчитувати значення з кожного фоторезистора, використовуючи лише один аналоговий вхід Arduino. Схема такого підключення детально показана на рисунку 2.20.

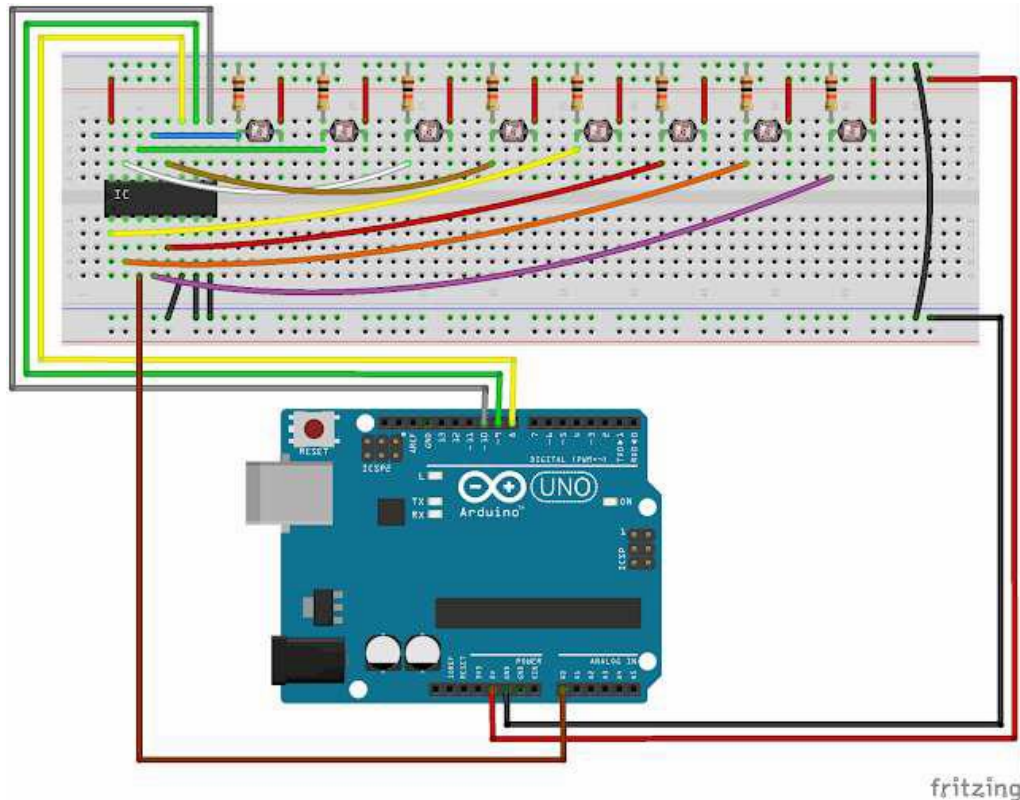


Рис. 2.20. Схема підключення 8 фоторезисторів до Ардуіно

Вивід INH (інгібітор) контролює, чи може відбуватися передача даних через мікросхему CD4051. Якщо цей вивід з'єднати з землею (тобто встановити логічний нуль), то передача даних буде завжди дозволена. Вивід COM підключений до аналогового входу A0 мікроконтролера Arduino, а виводи управління A, B і C - до цифрових виходів 8, 9 і 10 відповідно. Для швидшої зміни стану цих цифрових виходів замість використання функції `digitalWrite` можна безпосередньо працювати з регістром B, який відповідає за виводи з 8 по 13.

Лістинг програми для роботи із мультиплексором CD4051

```
byte AnalogIn = A0; // CD4051 pin 3 (Common in/out)
```

```

void setup() { Serial.begin(9600);
pinMode(8, OUTPUT); // CD4051 pin 11 (A)
pinMode(9, OUTPUT); // CD4051 pin 10 (B) pinMode(10, OUTPUT); //
CD4051 pin 9 (C)
}
void loop() {
for (byte i = 0; i < 8; i++) {
PORTB = (PORTB & B11111000) ^ i;
Serial.print(i); Serial.print(": ");
Serial.println(analogRead(AnalogIn));
}
Serial.println(); delay(1000); }

```

Якщо завантажити скетч в Arduino, то в моніторі порту ми побачимо значення, які зчитуються з фоторезисторів. Завдяки CD4051 можна підключити більше фоторезисторів до Arduino, ніж передбачено за замовчуванням. Крім того, можна об'єднати кілька мікросхем CD4051, щоб отримати ще більше аналогових входів.

#### **2.3.4. Розширення кількості цифрових виходів за допомогою зсувного регістра 74НС595**

Для збільшення кількості доступних цифрових виходів у мікроконтролері Arduino було використано зсувний регістр 74НС595 [50]. Цей вибір обумовлений його високою функціональністю та широким застосуванням у подібних задачах.

74НС559 являє собою восьмирозрядний синхронний зсувний регістр з можливістю паралельного або послідовного виведення даних. Ключовою особливістю цього компонента є його здатність приймати послідовний потік даних і паралельно виводити їх на вісім незалежних виходів. Це дозволяє значно збільшити кількість керованих пристроїв, використовуючи при цьому обмежену кількість виводів мікроконтролера.

Принцип роботи 74НС595 заснований на синхронній послідовній передачі даних. Біти даних послідовно подаються на вхід регістра, а синхронізуючий



імпульс фіксує значення бітів у внутрішніх тригерах. Після прийому восьми бітів дані стають доступними на паралельних виходах. Такий підхід дозволяє ефективно керувати великою кількістю зовнішніх пристроїв за допомогою невеликої кількості сигналів управління.

Варто зазначити, що 74НС595 підтримує три стани на виході: логічний нуль, логічна одиниця та високий імпеданс. Останній стан дозволяє відключати окремі виходи від схеми, що може бути корисним у деяких додатках.

Для розширення кількості керованих виходів кілька зсувних регістрів можуть бути з'єднані послідовно. У такому разі дані з першого регістра передаються на вхід наступного, дозволяючи керувати ще більшою кількістю зовнішніх пристроїв.

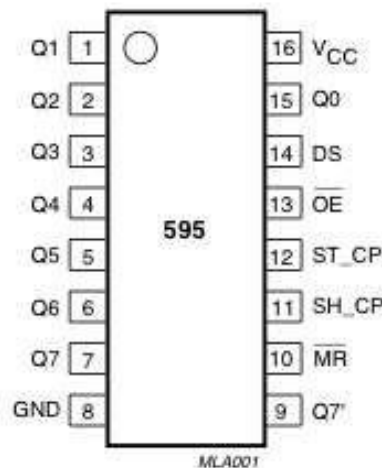


Рис. 2.21. Зсувний регістр 74НС595

#### Входи / виходи регістра

- ✓ Вивід 1-7, 15: Q0 - Q7 Паралельні виходи
- ✓ Вивід 8: GND Земля
- ✓ Вивід 9: Q7 Вихід для послідовного з'єднання регістрів
- ✓ Вивід 10: MR Скидання значень регістра.
- ✓ Вивід 11: SH\_CP Вхід для тактових імпульсів
- ✓ Вивід 12: ST\_CP Сінхронізація виходів
- ✓ Вивід 13: OE Вхід для перемикання стану виходів з високоомного в робочий стан
- ✓ Вивід 14: DS Вхід для послідовних даних

## Вивід 16: Vcc Живлення

Схема підключення з одним регістром наведено на рисунку 2.22

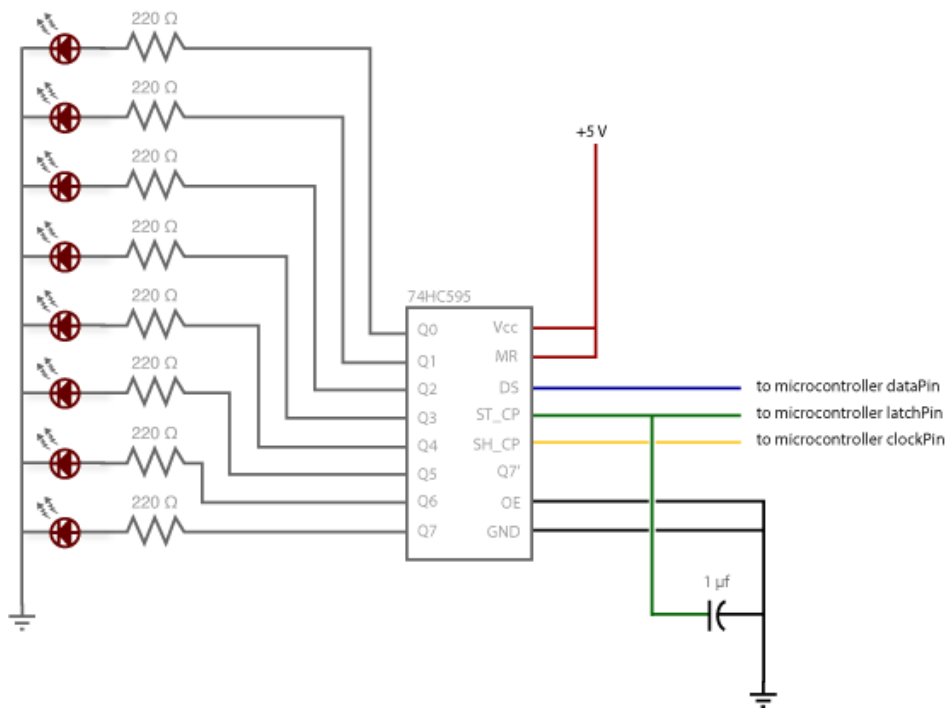


Рис. 2.22. Підключення регістру до Арудіно

- GND (вивід 8) на землю
- Vcc (вивід 16) до живлення 5В
- OE (вивід 13) на землю
- MR (вивід 10) до живлення 5В
- DS (вивід 14) з 11-м цифровий вихід Arduino (на схемі синій провід)
- SH\_CP (вивід 11) з 12-им цифровим виходом (жовтий провід)
- ST\_CP (вивід 12) с 8-им (зелений провід)

При використанні регістрів відмінних від 74HC595 слід зверитися з документацією і перевірити схему підключення.

Лістинг програми для роботи із зсувний регістром

```
// Пін підключений до ST_CP входу 74HC595 int latchPin = 8;  
// Пін підключений до SH_CP входу 74HC595 int clockPin = 12;  
// Пін підключений до DS входу 74HC595 int dataPin = 11;  
void setup () {  
    // встановлюємо режим OUTPUT pinMode (latchPin, OUTPUT); pinMode  
(clockPin, OUTPUT); pinMode (dataPin, OUTPUT);
```

```

}
void loop () {
// відраховуємо від 0 до 255 і відображаємо значення на світлодіоді
for (int numberToDisplay = 0; numberToDisplay <256; numberToDisplay ++ )
{
// встановлюємо синхронізацію "засувки" на LOW digitalWrite (latchPin,
LOW);

// передаємо послідовно на dataPin
shiftOut (dataPin, clockPin, MSBFIRST, numberToDisplay);

// закриваємо регістр, тим самим встановлюючи значення на виходах
digitalWrite (latchPin, HIGH);

// пауза перед наступною ітерацією delay (500); } }

```

Функціонування зсувного регістру 74HC595 детально ілюструється часовою діаграмою (рис. 2.23) та таблицею станів (рис. 2.24). Процес запису даних відбувається наступним чином: при переході сигналу на вході clockPin з низького логічного рівня (LOW) у високий (HIGH) регістр зчитує значення з входу даних DS. Зчитане значення запам'ятовується у внутрішньому регістрі. Для виведення записаних даних на виходи регістра необхідно подати позитивний фронт на вхід latchPin. Таким чином, дані фіксуються на виходах, що дозволяє керувати зовнішніми пристроями, такими як світлодіоди.

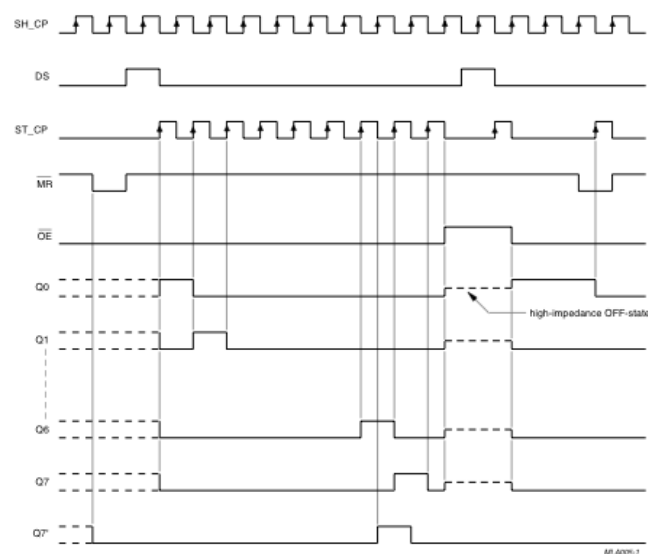


Рис. 2.23. Тимчасова діаграма сигналів регістру

**FUNCTION TABLE**

See note 1..

INPUT					OUTPUT		FUNCTION
SH_CP	ST_CP	OE	MR	DS	Q7'	Qn'	
X	X	L	L	X	L	n.c.	a LOW level on MR only affects the shift registers
X	↑	L	L	X	L	L	empty shift register loaded into storage register
X	X	H	L	X	L	Z	shift register clear; parallel outputs in high-impedance OFF-state
↑	X	L	H	H	Q6'	n.c.	logic high level shifted into shift register stage 0; contents of all shift register stages shifted through, e.g. previous state of stage 6 (internal Q6') appears on the serial output (Q7')
X	↑	L	H	X	n.c.	Qn'	contents of shift register stages (internal Qn') are transferred to the storage register and parallel output stages
↑	↑	L	H	X	Q6'	Qn'	contents of shift register shifted through; previous contents of the shift register is transferred to the storage register and the parallel output stages

**Note**

1. H = HIGH voltage level;  
 L = LOW voltage level;  
 ↑ = LOW-to-HIGH transition;  
 ↓ = HIGH-to-LOW transition;  
 Z = high-impedance OFF-state;  
 n.c. = no change;  
 X = don't care.

Рис. 2.24. Таблиця логіки регістру [50]

**Приклад використання каскаду зсувних регістрів**

Для розширення кількості цифрових виходів можна з'єднувати зсувні регістри в каскади. Наприклад, для отримання 16 виходів достатньо підключити два регістри 74HC595. Другий регістр під'єднується до джерела живлення та загальної землі аналогічно першому. Вхід даних (DS) другого регістра з'єднується з виходом Q7' першого регістра (синій провід). Сигнали синхронізації (SH\_CP і ST\_CP) обох регістрів з'єднуються паралельно (жовтий і зелений дроти відповідно). Детальна схема такого з'єднання представлена на рисунку 2.25.

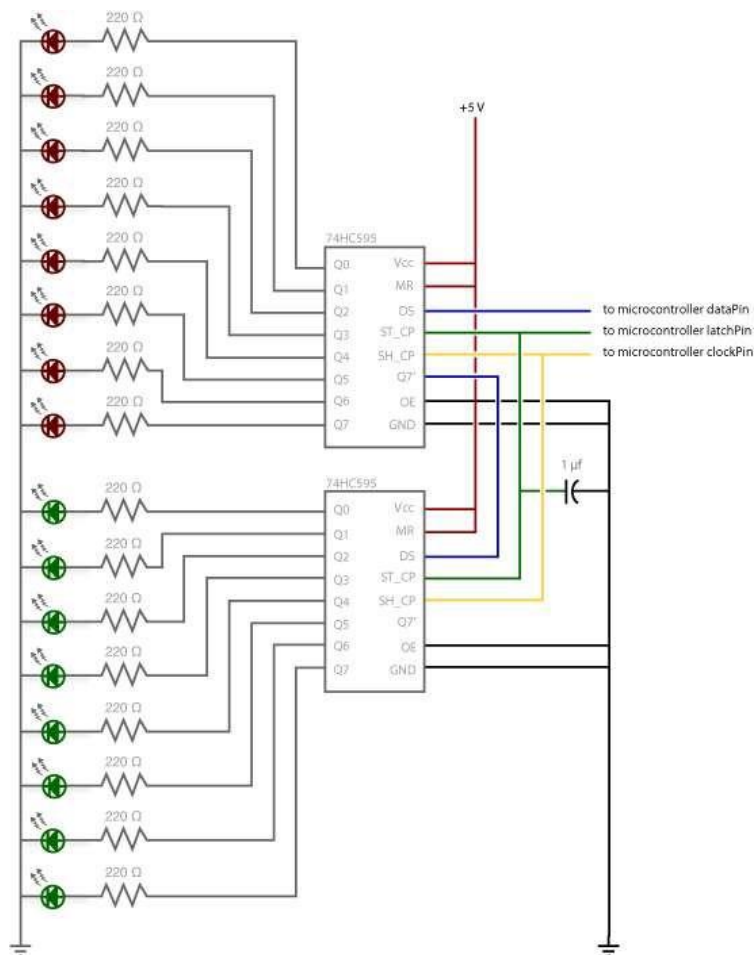


Рис. 2.25. Схема підключення каскаду регістрів

## Висновки до розділу 2

Проведений аналіз додаткових бібліотек для Arduino IDE продемонстрував їхню ефективність у вирішенні різноманітних завдань. Було розроблено та реалізовано програмні та апаратні методи усунення брязкоту контактів, що дозволило підвищити стабільність роботи електронних пристроїв. Для збільшення кількості цифрових входів було запропоновано метод мультиплексування, який дозволяє використовувати один аналоговий вхід мікроконтролера для зчитування даних з декількох цифрових входів. Крім того, було детально розглянуто застосування зсувних регістрів для розширення кількості цифрових виходів, що є ефективним рішенням для багатьох електронних проєктів.

## РОЗДІЛ 3. РЕАЛІЗАЦІЯ ПРИСТРОЮ ДЛЯ КОНТРОЛЬОВАНОГО НАГРІВАННЯ ВОДИ, ЯК ЕЛЕМЕНТ ІНТЕРНЕТУ РЕЧЕЙ

### 3.1. Конструювання пристрою контрольованого нагрівання води

Для створення системи контрольованого нагрівання води було обрано набір електронних компонентів, кожен з яких виконує свою специфічну функцію:

*Потенціометр:* дозволяє користувачеві вручну задавати бажану температуру води.

*Фоторезистор:* реагує на зміну освітленості, що може бути використано для автоматичного регулювання температури залежно від часу доби.

*Датчик температури DS18B20:* забезпечує точне вимірювання температури води.

*Датчик тиску MPXV5050:* контролює рівень води в системі.

*Інфрачервоний модуль:* може використовуватися для дистанційного керування системою або виявлення перешкод.

*Твердотільне реле:* керує потужними навантаженнями, такими як нагрівальний елемент.

*Драйвер двигуна:* керує роботою водяної помпи.

*Сервопривід:* може використовуватися для відкривання/закривання клапанів або керування іншими механізмами.

*RGB світлодіод:* служить для візуальної індикації стану системи.

*Модуль звуку:* відтворює звукові сигнали для сповіщення користувача.

*LCD дисплей:* відображає інформацію про температуру, тиск та інші параметри системи.

*Модуль Bluetooth:* забезпечує бездротове з'єднання зі смартфоном або іншим пристроєм для дистанційного керування.

*Тактові кнопки та перемикачі:* служать для взаємодії користувача з системою.

*Мікроконтролер Arduino Uno:* є "серцем" системи, обробляє дані від датчиків, керує виконавчими пристроями і забезпечує взаємодію з користувачем.

Всі компоненти були з'єднані згідно зі схемою, представленою на рисунку 3.1.



Рис. 3.1. Компоненти для конструювання пристрою

### 3.1.1. Схема електричних з'єднань пристрою

Функціональні елементи розробленої системи з'єднані відповідно до електричної принципової схеми, представленої на рисунку 3.2. Ця схема деталізує взаємозв'язки між мікроконтролером та периферійними пристроями, визначаючи таким чином архітектуру системи.



Рис. 3.2. Схема електрична принципова

### 3.1.2. Прототипування на макетній платі

Перед остаточним монтажем електронного пристрою було проведено його прототипування на безпайковій макетній платі. Це дозволило перевірити правильність розробленої схеми, налаштувати параметри роботи компонентів та виявити потенційні помилки. На рисунку 3.3 наведено приклад підключення тактових кнопок та потенціометрів до мікроконтролера на макетній платі. Такий

підхід є стандартною практикою в електроніці, оскільки він дозволяє швидко і легко вносити зміни в схему та проводити експерименти.

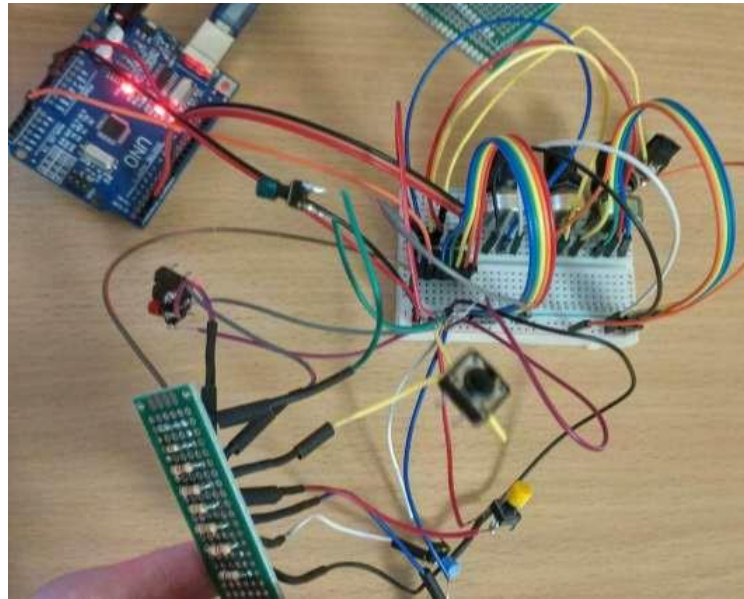


Рис. 3.3. З'єднання компонентів на макетній платі

Після успішного тестування підключення основних компонентів на макетній платі, можна переходити до остаточного складання пристрою. Всі елементи схеми з'єднуються між собою відповідно до принципової схеми, зображеної на рисунку 3.4. Особливу увагу слід приділити з'єднанню загальної землі всіх компонентів. Загальна земля забезпечує єдиний потенціал для всіх елементів схеми, що є необхідною умовою для їх синхронної роботи та коректної передачі сигналів.

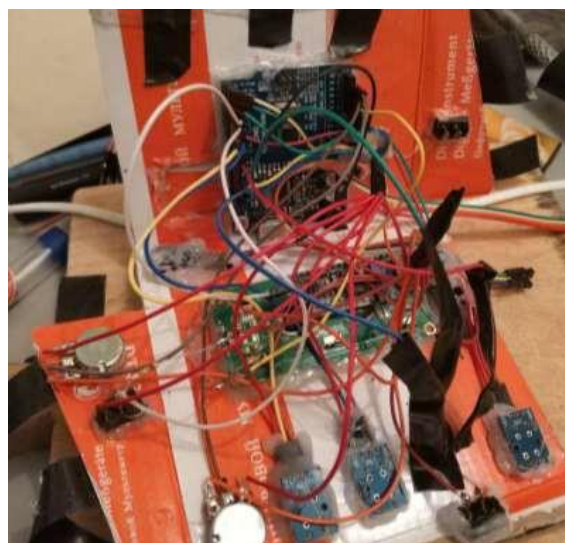


Рис. 3. 4. З'єднання усіх електричних компонентів між собою



### 3.1.3. Макет для корпусу пристрою

Функціональний макет системи контролюваного нагрівання води виконаний у вигляді корпусу з ПВХ паперу чорного кольору (127x105x35 мм, 192 г). На передній панелі корпусу розміщені елементи керування: тактові кнопки, потенціометри, перемикачі та фоторезистор. Для підключення живлення та завантаження програмного забезпечення передбачені відповідні роз'єми. Візуальну індикацію стану системи забезпечує RGB світлодіод. Зовнішній вигляд готового макету представлено на рисунку 3.5.



Рис. 3.5. Зовнішній вигляд макета пристрою

### 3.1.4. Приклад програмної реалізації

Для реалізації алгоритмів управління пристроєм було розроблено програмне забезпечення на мові C++ в середовищі Arduino IDE. Програмний код, представлений в Додатку А, забезпечує взаємодію між апаратними компонентами та реалізує необхідні функції. Перед завантаженням програми в мікроконтролер було проведено її детальну дебагінг процедуру в середовищі Arduino IDE з метою виявлення та усунення можливих помилок.

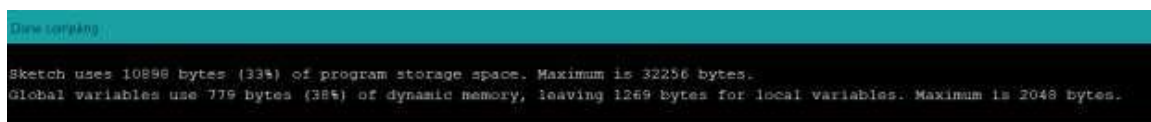


Рис. 3.6. Консоль компілятора Arduino IDE

### 3.1.5. Принцип функціонування пристрою для контролюваного нагрівання води

Розроблений пристрій забезпечує точний контроль над процесом

нагрівання та рівнем води. Він оснащений датчиками, які дозволяють вимірювати температуру з точністю до 1°C та рівень води з кроком 100 мл у реальному часі. Для безпечної експлуатації реалізовано систему блокування, яка запобігає включенню пристрою за відсутності чайника або при недостатньому рівні води.

Пристрій підтримує кілька режимів роботи:

1. Нагрівання до кипіння. При активації цього режиму нагрівальний елемент працює до досягнення температури 100°C, після чого автоматично відключається.
2. Нагрівання до заданої температури. Користувач може встановити бажану температуру за допомогою потенціометра. Нагрівання триває до досягнення заданого значення.
3. Підтримання заданої температури. Після досягнення заданої температури система підтримує її на постійному рівні, включаючи нагрівальний елемент при зниженні температури.
4. Наповнення до заданого рівня. Користувач встановлює необхідний об'єм води за допомогою потенціометра. Насос працює до досягнення заданого рівня.
5. Підтримання заданого рівня. Система автоматично підтримує заданий рівень води, доливаючи її за необхідності.
6. Аварійне відключення. Цей режим дозволяє примусово зупинити будь-який активний процес.

Пристрій також забезпечує додаткові функції:

- Регулювання яскравості дисплея. Яскравість дисплея може регулюватися вручну або автоматично залежно від освітленості навколишнього середовища.
- Звукова індикація. Пристрій видає звукові сигнали для сповіщення користувача про завершення процесів нагрівання або наповнення.
- Візуальна індикація. RGB-світлодіод змінює колір залежно від температури води, що дозволяє візуально оцінити стан системи.

Запропоновані режими роботи та додаткові функції роблять цей пристрій зручним та ефективним інструментом для контролю процесу нагрівання води.

### **3.2. Розробка мобільного додатку для управління системою контрольованого нагрівання води**

Для забезпечення дистанційного керування розробленою системою було створено мобільний додаток, сумісний з операційною системою Android [51]. Як платформа для розробки було обрано візуальне середовище програмування MIT App Inventor 2, яке завдяки своєму інтуїтивному інтерфейсу та блоковому підходу дозволяє створювати функціональні мобільні додатки без глибоких знань в області програмування [52].

#### **3.2.1. Середовище розробки MIT App Inventor 2**

MIT App Inventor 2 використовує візуальну мову програмування, засновану на принципах, схожих до мов Scratch та StarLogoTNG. Це спрощує процес розробки та робить його доступним для широкого кола користувачів. Створений у Google, App Inventor базується на фреймворку GNU Kawa, який забезпечує трансляцію візуальних блоків у байт-код для Android. Варто зазначити, що App Inventor є безкоштовним онлайн-сервісом, який користується значною популярністю серед розробників, особливо серед освітніх установ.

Завдяки використанню MIT App Inventor 2 вдалося реалізувати мобільний додаток, який дозволяє користувачеві здійснювати повний контроль над системою контрольованого нагрівання води зі свого смартфона. Цей додаток забезпечує зручний та інтуїтивний інтерфейс, що дозволяє користувачеві легко налаштовувати режими роботи, відстежувати параметри системи та отримувати необхідну інформацію.

#### **3.2.2. Реалізація програмного забезпечення для мобільного додатку**

Процес розробки мобільного додатку для керування системою контрольованого нагрівання води здійснювався в середовищі MIT App Inventor 2. Спочатку було створено візуальний інтерфейс користувача (ВІК) у відповідному розділі середовища (рис. 3.7). ВІК додатку був розроблений з урахуванням інтуїтивності та зручності використання, забезпечуючи

користувачеві простий доступ до основних функцій системи.7.

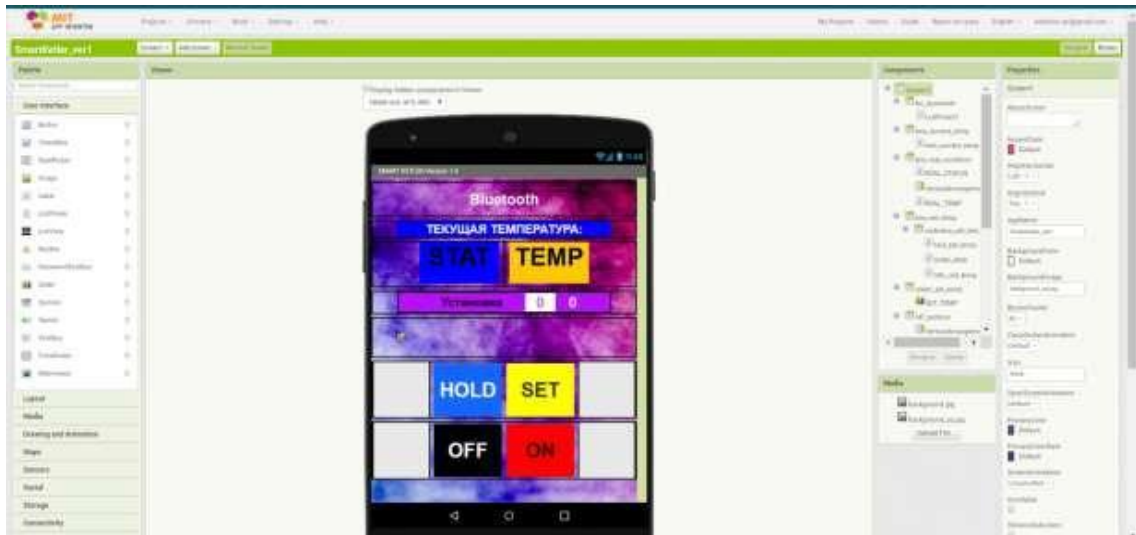


Рис. 3.7. Середовище розробки App Inventor 2

Логіка роботи додатку була реалізована за допомогою блокової мови програмування, доступної в розділі Blocks. Цей підхід дозволив візуально відобразити алгоритми управління та спростити процес розробки. На рисунку 3.8 представлений фрагмент блокового коду, який ілюструє взаємодію між елементами ВІК та функціями системи контролюваного нагрівання води.

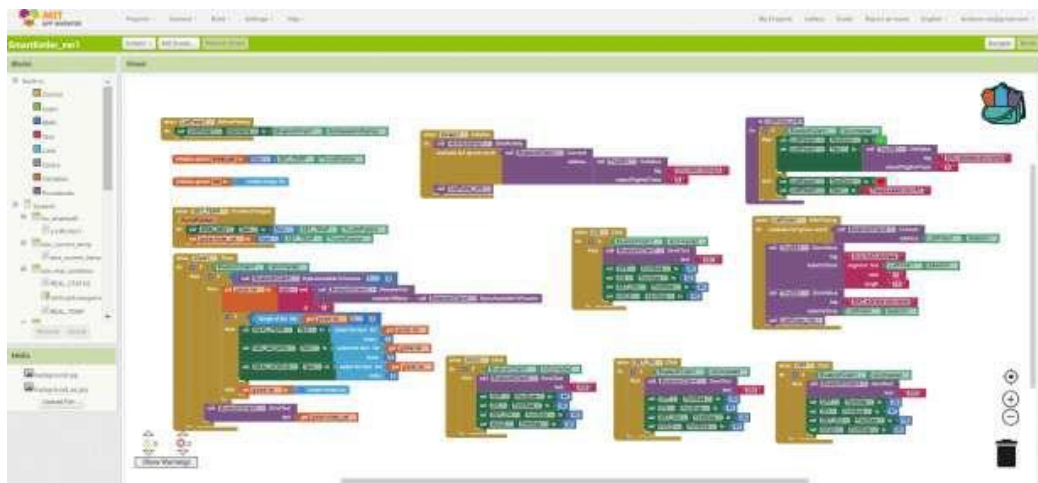


Рис. 3.8. Програмний код мобільного додатку із блоків

Для перевірки працездатності розробленого додатку було скомпільовано його в файл формату .apk. Отриманий файл було інстальовано на смартфон, після чого було встановлено з'єднання з Bluetooth-модулем системи. Результати тестування, що підтверджують коректну роботу додатку, представлені на рисунку 3.9.



Рис. 3.9. Мобільний додаток для керування пристроєм

### 3.2.3. Принцип дії програми

Мобільний додаток функціонує як повний дублікат фізичного інтерфейсу пристрою. Вся взаємодія між додатком та пристроєм здійснюється за допомогою бездротового протоколу Bluetooth. При запуску додаток автоматично виявляє та підключається до попередньо налаштованого пристрою.

Графічний інтерфейс додатку містить елементи керування, аналогічні фізичним кнопкам та потенціометрам на пристрої.

Повзунки використовуються для встановлення значень температури та рівня води, замінюючи відповідні потенціометри на пристрої.

Кнопки ініціюють виконання певних операцій:

"Start" запускає процес нагрівання води до 100°C.

"Set Temp" встановлює та підтримує задану температуру води.

"Hold Temp" підтримує постійну задану температуру води.

"Set Water" встановлює та підтримує заданий рівень води.

"Hold Water" підтримує постійний заданий рівень води.

"Stop" примусово зупиняє всі активні процеси.

Таким чином, користувач може керувати системою контролюваного

нагрівання води як за допомогою фізичних кнопок на пристрої, так і за допомогою мобільного додатку, що забезпечує додаткову зручність та гнучкість у використанні.

### **Висновки до розділу**

У розділі детально описана структура та функціональні можливості розробленої системи контрольованого нагрівання води. Система побудована на базі мікроконтролера Arduino Uno та оснащена необхідними датчиками та виконавчими пристроями. Для забезпечення зручного керування системою розроблений мобільний додаток для Android, який дозволяє користувачеві віддалено налаштовувати та контролювати роботу системи. Запропонована система не лише автоматизує процес нагрівання води, але й забезпечує гнучке керування та моніторинг усіх параметрів. Програмне забезпечення системи розроблене на мові C++ в середовищі Arduino IDE. Запропоноване рішення відрізняється простотою використання, високою ефективністю та може знайти широке застосування в побуті та промисловості.

## ВИСНОВКИ

У рамках магістерської роботи було успішно розроблено та впроваджено інноваційну систему контрольованого нагрівання води, яка інтегрується в концепцію Інтернету речей.

Проведене дослідження дозволило вирішити наступні завдання:

1. Створено фізичний прототип системи на основі мікроконтролера Arduino Uno, оснащеної необхідними датчиками та виконавчими пристроями для забезпечення точного вимірювання та контролю параметрів процесу нагрівання.
2. Розроблено мікропрограмне забезпечення для мікроконтролера, яке забезпечує управління системою та взаємодію з периферійними пристроями. Додатково, створено мобільний додаток для смартфонів на платформі Android, що дозволяє користувачеві здійснювати дистанційне керування системою.
3. Здійснено ряд досліджень та експериментів для оптимізації роботи системи, таких як усунення шуму в аналогових сигналах, розширення кількості цифрових входів та виходів мікроконтролера.
4. Система інтегрується в концепцію Інтернету речей завдяки можливості дистанційного керування через мобільний додаток.

Загалом, результати проведеної роботи свідчать про успішне вирішення поставлених завдань та створення функціональної та перспективної системи контрольованого нагрівання води. Система забезпечує точний контроль температури та рівня води, автоматичне управління процесом нагрівання та зручний інтерфейс користувача, може бути легко адаптована до різних потреб завдяки модульній конструкції та відкритому програмному коді.

Розроблена система має великий потенціал для подальшого розвитку, зокрема, шляхом інтеграції з іншими системами "розумного дому" та використання більш потужних мікроконтролерів.

## СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Osborne, Adam (1980). An Introduction to Microcomputers. Volume 1: Basic Concepts (2nd ed.). Berkeley, California: Osborne-McGraw Hill
2. Krishna Kant Microprocessors And Microcontrollers: Architecture Programming And System Design, PHI Learning Pvt. Ltd., 2007
3. History of the Microprocessor URL: <https://meetingtomorrow.com/content-library/history-of-the-microprocessor>
4. National Semiconductor Corporation History URL: <http://www.fundinguniverse.com/company-histories/national-semiconductor-corporation-history/>
5. Hewlett-Packard history URL: <https://www.siliconvalleyhistorical.org/hewlett-packard-history>
6. RISC — архитектура процессора. История и принципы работы. URL: <http://www.xtechx.ru/c40-visokotehnologichni-spravochnik-hitech-book/risc-architecture-processor/>
7. Що таке мікроконтролер Ардуіно URL: <https://learn.sparkfun.com/tutorials/what-is-an-arduino>
8. Різновид мікроконтролерів Arduino URL: <https://store.arduino.cc/arduino-genuino/most-popular>
9. Протокол обмена STK500 URL: <http://microsin.net/programming/avr/avr068-stk500-communication-protocol.html>
10. Atmel ATmega640 Datasheet URL: [https://ww1.microchip.com/downloads/en/devicedoc/atmel-2549-8-bit-avr-microcontroller-atmega640-1280-1281-2560-2561\\_datasheet.pdf](https://ww1.microchip.com/downloads/en/devicedoc/atmel-2549-8-bit-avr-microcontroller-atmega640-1280-1281-2560-2561_datasheet.pdf)
11. Потенциометры. Виды и устройство. Работа и особенности URL: <https://electrosam.ru/glavnaja/slabotochnye-seti/oborudovanie/potentsiometry/>
12. Фоторезистор. Принцип работы, характеристики URL: <http://www.joyta.ru/7603-fotorezistor-osnovnaya-informaciya/>
13. DS18B20 – датчик температуры URL: <http://mypractic.ru/ds18b20-datchik-temperature-s-interfejsom-1-wire-opisanie-na-russkom-yazyke.html>



14. Цифрове вимірювання рівня гарячої води на мікропроцесорній платформі Arduino // International Scientific And Practical Conference «Technical Sciences: History, The Present Time, The Future, EU Experience», 2019 р.,-С.69-72
15. Медь - коррозионная стойкость URL: <http://delta-grup.ru/bibliot/41/21.htm>
16. Атмосферное давление // Большая советская энциклопедия : [в 30 т.] / гл. ред. А. М. Прохоров. — 3-е изд. — М., 1969—1978
17. Закон Бойля — Мариотта // Физическая энциклопедия / Гл. ред. А. М. Прохоров. — М/, 1988. — Т. 1. — 704 с.
18. Ir Remote And Receiver On An Arduino URL: <http://www.circuitbasics.com/arduino-ir-remote-receiver-tutorial/>
19. Электромагнитные реле. Виды и работа. Устройство и применение. URL: <https://electrosam.ru/glavnaja/jelektrooborudovanie/rozetki-vykljuchateli/elektromagnitnye-rele/>
20. Твердотельные реле URL: <http://electricalschool.info/spravochnik/apparaty/1450-tverdotelnye-rele.html>
21. Н-мост и схема работы для управления двигателями URL: <https://arduino-master.ru/uroki-arduino/shema-raboty-n-mosta-dlya-upravleniya-dvigatelyami/>
22. Драйвера мотора на L298N URL: <https://robotchip.ru/obzor-drayvera-motora-na-l298n/>
23. Servo Motor Control using Arduino URL: <https://circuitdigest.com/microcontroller-projects/arduino-servo-motor-control-code-and-circuit>
24. Восприятие цветов человеком URL: <http://rosdesign.com/design/kolorofdesign.htm>
25. Смешение цветов URL: <https://haircolor.org.ua/koloristika/100-smeshenie-tsvetov.html>
26. Пищалка – пьезодинамик Ардуино URL: <https://arduino-master.ru/uroki-arduino/pishhalka-pe zodinamik-arduino/>

27. Construction and Working Principle of LCD Display URL:  
<https://www.elprocus.com/ever-wondered-lcd-works/>
28. Bluetooth URL: [http://www.smartphone.ua/w\\_bluetooth.html](http://www.smartphone.ua/w_bluetooth.html)
29. Arduino IDE software. URL: <https://www.arduino.cc/en/main/software>
30. Библиотеки Arduino URL: <https://doc.arduino.ua/ru/prog/Libraries>
31. Библиотека LiquidCrystal\_I2C URL:  
[https://wiki.iarduino.ru/page/Working\\_with\\_character\\_LCD\\_displays](https://wiki.iarduino.ru/page/Working_with_character_LCD_displays)
32. Библиотека OneWire URL: <https://xakep.ru/2015/05/10/arduino-digital-temp-wire/>
33. Библиотека ServoSmooth URL: <https://alexgyver.ru/servosmooth/>
34. Библиотека GyverButton URL: <https://alexgyver.ru/gyverbutton/>
35. Интерфейсы последовательной связи URL:  
<https://studfile.net/preview/357307/page:23/>
36. Интерфейс SPI URL: <http://s-engineer.ru/interfejs-spi/>
37. Интерфейс I2C URL: [http://itt-ltd.com/reference/ref\\_i2c.html](http://itt-ltd.com/reference/ref_i2c.html)
38. Интерфейс UART URL: <https://voltiq.ru/wiki/uart-interface/>
39. Стягивающие и подтягивающие резисторы URL: <http://funnydiy.ru/1010>
40. Постоянная времени RC-цепи URL: [https://www.lcard.ru/lexicon/rc\\_const](https://www.lcard.ru/lexicon/rc_const)
41. Datasheet SN74HC14 URL: <http://www.ti.com/lit/ds/symlink/sn74hc14.pdf>
42. Аналогово-цифровой преобразователь в микроконтроллерах URL:  
<http://narodstream.ru/avr-urok-22-izuchaem-acp-chast-1/>
43. Последовательное соединение резисторов URL:  
<http://hightolow.ru/resistor3.php>
44. Делитель напряжения на резисторах URL: <http://www.joyta.ru/7328-delitel-napryazheniya-na-rezistorax-raschet-onlajn/>
45. О битовых операциях URL: <https://tproger.ru/translations/bitwise-operations/>
46. Стягивающие и подтягивающие резисторы URL: <http://funnydiy.ru/1010>
47. Последовательное и параллельное соединение резисторов URL:  
<http://hightolow.ru/resistor3.php>
48. Комбинаторика. Сочетания URL:

<https://mathematics.ru/courses/algebra/content/chapter4/section2/paragraph3>

49. Мультиплексоры и демультиплексоры: схемы, принцип работы URL:

<https://pue8.ru/silovaya-elektronika/908-multipleksory-i-demultipleksory.html>

50. Сдвиговый регистр 74HC595 URL: <https://cxem.net/arduino/arduino166.php>

51. MIT App Inventor URL: <https://appinventor.mit.edu/>

## ДОДАТОК А

### Лістинг коду

```
#include <LiquidCrystal_I2C.h>
#include <OneWire.h> LiquidCrystal_I2C lcd(0x3F,16,2);
OneWire ds(12); //жовтий провід
boolean red_but, blue_but, yellow_but, black_but; //поточний стан кнопок
boolean red_flag=0, blue_flag=0, yellow_flag=0, black_flag=0; //Попередні
стан кнопок
unsigned long red_last, blue_last, yellow_last, black_last, last_serial,
last_buzzer; //останнє натискання цих кнопок
int level_p; // Рівень потенціометра для lcd екрану int level_f; // Рівень світла
з фоторезистора
int lcd_stat; // стан перемикача між потенціометром(0) та
фоторезистором(1)
int brightness; // Яскравість екрану
int serial_val; // для зберігання значень Serial.parseInt
int temp_stat; // вибір між аналоговим (0) і цифровим (1) установкою
температури int analog_temp; // Встановлення температури з потенціометра
int digit_temp; // Встановлення температури з телефону
int level_temp; // значення аналогової/цифрової установки температури,
залежно стану
int digit_stat; // буфер для посилення on(1) та off(0) на телефон
int condition = 0; //Стан системи int ketler_stat;
int red, blue, green; // Значення ШІМ для RGB int fake_temp;
int temperature = 0; // Глобальна змінна для зберігання значення
температури датчика DS18B20
long lastUpdateTime = 0; // Змінна для зберігання часу останнього
зчитування з датчика
const int TEMP_UPDATE_TIME = 1000; // Визначаємо періодичність
перевірок
byte simvol[8] =
{B00111,B00101,B00111,B00000,B00000,B00000,B00000,B00000,}; // Символ
градуса
void setup() { Serial.begin(9600); Serial.setTimeout(10); lcd.init();
lcd.backlight();// Включаємо підсвічування дисплея pinMode(red_pin,
INPUT_PULLUP); pinMode(blue_pin, INPUT_PULLUP); pinMode(yellow_pin,
INPUT_PULLUP); pinMode(black_pin, INPUT_PULLUP); pinMode(red_led,
OUTPUT);
pinMode(green_led, OUTPUT); pinMode(blue_led, OUTPUT);
pinMode(lcd_pin, OUTPUT); pinMode(lcd_knopka,INPUT);
pinMode(poten_lcd,INPUT); pinMode(poten_temp,INPUT);
pinMode(fotor_pin,INPUT); pinMode(chainik,INPUT);
digitalWrite(rele, HIGH); //про всяк випадок вимикаю реле (чайник)
}
```

```

void loop() {
  if (temperature < 5) ketler_stat = 1; else ketler_stat = 1;
  //КІД ДЛІЯ ВІБОРУ між автояркістю LCD ЕКРАНА і ручного
налаштування lcd_stat = digitalRead(lcd_knopka);
  level_p = analogRead(poten_lcd); level_f = analogRead(fotor_pin); level_p =
map (level_p,0,1024,0,256); level_f = map (level_f,500,1024,255,0); level_f =
constrain(level_f, 0, 255); switch (lcd_stat) {
    case 0: analogWrite(lcd_pin,level_p); brightness = map (level_p, 0, 255, 0, 100);
    break;
    case 1: analogWrite(lcd_pin,level_f); brightness = map (level_f, 0, 255, 0, 100);
break; }
  // РОБОТА З ТЕМПЕРАТУРАМИ
  switch (temp_stat) {
    case 0: level_temp = analog_temp; break;
    case 1: level_temp = digit_temp; break; }
  analog_temp = analogRead(poten_temp); analog_temp = map
(analog_temp,5,1024,0,100);
  detectTemperature(); // Визначаємо температуру від датчика DS18B20
  // fake_temp = analogRead(chainik);
  // Temperature = map (fake_temp, 0,1024, 0,165);
  //КОД для прийому даних з блютуз if (Serial.available()>0)
  {
    serial_val = Serial.parseInt();
    if (serial_val >= 0 && serial_val <= 100)
    {digit_temp=serial_val; } else if (serial_val == 301)
    {condition = 1;}
    else if (serial_val == 300)
    {condition = 0;}
    else if (serial_val == 302)
    {condition = 2;}
    else if (serial_val == 303)
    {condition = 3;}
    else if (serial_val == 304)
    {condition = 4;}
    else if (serial_val == 305)
    {condition = 5;}
  }
  //КОД ДЛІЯ ЧИТАННЯ КНОПОК
  red_but = !digitalRead(red_pin); // Вважати поточне положення кнопки
blue_but = ! DigitalRead (blue_pin);
  yellow_but = !digitalRead(yellow_pin); black_but = !digitalRead(black_pin)
  //ВКЛ
  if (red_but == 1 && red_flag == 0 && millis() - red_last > 100) { red_flag = 1;
condition = 1; red_last = millis();
  if (red_but == 0 && red_flag == 1) { red_flag = 0; }

```

```

//аналогова ПОСТІЙНА ПІДТРИМКА
if (blue_but == 1 && blue_flag == 0 && millis() - blue_last > 100) { blue_flag
= 1;
temp_stat = 0;
condition = 3; blue_last = millis(); }
if (blue_but == 0 && blue_flag == 1) { blue_flag = 0; }
//аналогова ВСТАНОВЛЕННЯ ОДИН РАЗ
if (yellow_but == 1 && yellow_flag == 0 && millis() - yellow_last > 100) {
yellow_flag = 1;
temp_stat = 0;
condition = 2; yellow_last = millis(); }
if (yellow_but == 0 && yellow_flag == 1) { yellow_flag = 0; }
//ВИКЛ
if (black_but == 1 && black_flag == 0 && millis() - black_last > 100) {
black_flag = 1;
condition = 0; black_last = millis(); }
if (black_but == 0 && black_flag == 1) {
black_flag = 0;
//КОД для вибору стану чайника (основна частина) switch (ketler_stat) {
case 1
switch (condition) {
case 0: digitalWrite(rele, HIGH); lcd.setCursor(0, 1); lcd.print("stat = OFF");
temp_stat = 0; digit_stat = 0;
break; case 1:
if (temperature < 97 ) {
digitalWrite(rele, LOW); lcd.setCursor(0, 1); lcd.print("stat = ON"); digit_stat =
1;
}
else if (temperature = 98) {
condition = 0; digitalWrite(rele, HIGH);
digitalWrite(red_led, HIGH); digitalWrite(blue_led, HIGH);
digitalWrite(green_led, HIGH);
else if (temperature > 98) { condition = 0;
}
break case 2:
if (temperature < analog_temp ) {
digitalWrite(rele, LOW); lcd.setCursor(0, 1); lcd.print("stat = ~ON~");
digit_stat = 1;
} else { condition = 0;
}
break; case 3:
if (temperature < analog_temp ) {
digitalWrite(rele, LOW); lcd.setCursor(0, 1); lcd.print("stat = HOLD");
digit_stat = 1;
} else {

```

```

        digitalWrite(rele, HIGH); lcd.setCursor(0, 1); lcd.print("stat = WAIT");
digit_stat = 2;
    }
    break; case 4:
    if (temperature < digit_temp ) {
        digitalWrite(rele, LOW); lcd.setCursor (0, 1); lcd.print("stat = d.ON");
temp_stat = 1; digit_stat = 1;
    } else {
        condition = 0; temp_stat = 0; digit_stat = 0;
    }
    break; case 5:
    if (temperature < digit_temp ) {
        digitalWrite(rele, LOW); lcd.setCursor(0, 1); lcd.print("stat=d.HOLD");
temp_stat = 1; digit_stat = 1;
    } else {
        digitalWrite(rele, HIGH); lcd.setCursor(0, 1); lcd.print("stat=d.WAIT");
digit_stat = 2;
    }
    break;
}
//КОД ДЛЯ RGB СВИТЛЮДІЮДА
if (temperature < 30 ) { red = 0; blue = 0;
if (temperature > 30 && temperature < 50) { red = 135; green = 0; } //ЖОВТИЙ
if (temperature > 70 & temperature < 90) { red = 0; green = 15;
switch (digit_stat) {
case 0: digitalWrite(red_led, LOW);
case 1: analogWrite(red_led, red);
break
case 2: digitalWrite(red_led, LOW); digitalWrite(blue_led, LOW);
analogWrite(green_led, 80);
else digitalWrite(green_led, LOW);
//КОД ДЛЯ ВІДОБРАЗЕННЯ НА LCD ЕКРАНІ
lcd.createChar(1, simvol); lcd.setCursor(0, 0);
if (temperature > 99){ lcd.setCursor(6, 0);
else if (temperature < 10) { lcd.setCursor(6, 0); ); lcd.print(temperature);
lcd.setCursor(9,0); lcd.print("\1");
lcd.setCursor(10,0); lcd.print("C|");
if (level_temp < 10 ){ lcd.setCursor(12, 0); }
lcd.setCursor(14,0); lcd.print("\1");
lcd.setCursor(11, 1); lcd.print("|");
else if (brightness < 10) { lcd.setCursor(12, 1); lcd.print(brightness);
"); }
else { lcd.setCursor(12, 1); lcd.print(brightness); %"); //ЯРКІСТЬ
ПІДСВІТЛЕННЯ ЕКРАНА
break; // брейк для свитч кетлет стат кейз 1

```

```

analogWrite(red_led, 50);
//КІД ДЛЯ ВИСНОВКУ В SERIAL PORT
if (millis() - last_serial > 500) {
  Serial.print(temperature); Serial.print("C");
  switch (digit_stat) {
    case 0: Serial.print("OFF");
    case 1: Serial.print("ON");
    case 2: Serial.print("WAIT");
  }
  last_serial = millis();
}
}

int detectTemperature(){
  byte data[2]; // Місце для значення температури
  ds.reset(); // Починаємо взаємодію зі скидання всіх попередніх команд і
параметрів ds.write(0xCC); // Даємо датчику DS18B20 команду пропустити
пошук за адресою.
  ds.write(0x44); // Даємо команду виміряти температуру.
  // Саме значення температури ми ще отримуємо.
  if (millis() - lastUpdateTime > TEMP_UPDATE_TIME)

  {
    lastUpdateTime = millis();
    ds.reset(); // Тепер готуємось отримати значення виміряної температури
ds.write(0xCC);
    ds.write(0xBE); // Просимо передати нам значення регістрів зі значенням
температури
    data[0] = ds.read(); // Читаємо молодший байт значення температури data[1]
= ds.read();
    // Формуємо значення
    temperature = (data[1] << 8) + data[0];
  }
}

```