Дрогобицький державний педагогічний університет
імені Івана Франка
кафедра фізики та інформаційних систем

Роман ЛЕШКО, Ольга ЛЕШКО

РЕАЛІЗАЦІЯ STEM-ПРОЄКТІВ НА БАЗІ ARDUINO

Методичні рекомендації до виконання індивідуальних проєктних завдань

Дрогобич
2024

УДК 004.9:37.091.313(072)
Р31

Рекомендовано вченою радою Дрогобицького державного педагогічного університету імені Івана Франка
(протокол № 12 від 29.11.2024 р.)

Рецензенти:

- **Віктор БРИТАН**, доцент кафедри фізики та інформаційних систем Дрогобицького державного педагогічного університету імені Івана Франка, кандидат фізико-математичних наук.
- **Володимир ПОПОВИЧ,** доцент кафедри технологічної та професійної освіти Дрогобицького державного педагогічного університету імені Івана Франка, кандидат фізико-математичних наук.

Відповідальний за випуск –
Юрій УГРИН, кандидат фізико-математичних наук, доцент кафедри фізики та інформаційних систем Дрогобицького державного педагогічного університету імені Івана Франка.

Роман Лешко, Ольга Лешко
Реалізація STEM-проєктів на базі Arduino : методичні рекомендації до виконання індивідуальних проєктних завдань. Дрогобич : Дрогобицький державний педагогічний університет імені Івана Франка, 2024. 44 с.

Методичні рекомендації **“Реалізація STEM-проєктів на базі Arduino”** написані відповідно до робочих програм навчальних дисциплін “**Інформаційно-керуючі системи та STEM-технології**” для підготовки фахівців магістерського рівня вищої освіти спеціальностей 014 Середня освіта (Фізика) та 104 Фізика та астрономія, 122 Комп’ютерні науки, затверджених вченою радою Дрогобицького державного педагогічного університету імені Івана Франка.

Подано основні рекомендації, що дають змогу ефективно реалізувати STEM-проєкти. Наведено приклади з детальним описом, як виконувати згадані проєкти.

Бібліографія 12 назв

З М І С Т

ПЕРЕДМОВА **4**

1. STEM-проєкт “RGB-нічник з керуванням його кольором за допомогою руху рук” **5**

2. STEM-проєкт “Вимірювання відстані” **14**

3. STEM-проєкт “Парктроник” **18**

4. STEM-проєкт “Дистанційний вмикач світла” **23**

5. STEM-проєкт “Розумна оранжерея” **28**

6. STEM-проєкт “Радіочастотна ідентифікація” **35**

ПІСЛЯМОВА **40**

Список використаних джерел **43**

ПЕРЕДМОВА

Сучасний світ все більше зосереджений на інноваціях та технологічному розвитку. Одним із ключових напрямів освіти, який готує молоде покоління до викликів майбутнього, є **STEM** (Science, Technology, Engineering, Mathematics). Він об'єднує науки, технології, інженерію та математику, дозволяючи студентам не лише здобувати теоретичні знання, а й застосовувати їх на практиці. Одним із ефективних інструментів для реалізації **STEM**-проєктів є **Arduino** – універсальна платформа для створення електронних пристроїв та систем.

Arduino відкриває перед учнями та студентами безмежні можливості для творчості, експериментів і розв'язання реальних інженерних задач. Завдяки доступності апаратної та програмної частини навіть початківці можуть легко створювати різноманітні проєкти: від простих світлодіодних схем до складних роботизованих систем. У поєднанні з іншими компонентами, такими як датчики, двигуни та дисплеї, **Arduino** дає змогу створювати практичні й актуальні рішення для різних галузей – від домашньої автоматизації до наукових досліджень.

Цей навчальний посібник має на меті познайомити читачів із основами роботи з платформою **Arduino**, а також навчити їх проєктувати та реалізовувати **STEM**-проєкти різної складності. Він допоможе розвинути навички в програмуванні, електроніці та системному мисленні, що є необхідними для сучасних інженерів і науковців.

Книга орієнтована на студентів, викладачів, інженерів та всіх, хто цікавиться практичним застосуванням технологій для виконання реальних завдань. Крок за кроком ми розглянемо основні компоненти та можливості Arduino, опануємо базові принципи побудови електронних схем і написання програмного коду.

1. STEM-проєкт “RGB-нічник з керуванням його кольором за допомогою руху рук”

Мета:

Метою проєкту є створення нічника з трьома кольоровими лампами (червона, зелена і синя), який можна вмикати за допомогою руху рук.

Обладнання:

1. Макетна плата.
2. Платформа **Arduino**.
3. Три світлодіоди (червоний, зелений і синій), які можна замінити на три реле (якщо використовувати справжні лампи).
4. Ультразвуковий датчик відстані.
5. Інфрачервоний пульт з приймачем.
6. З'єднувальні проводи.

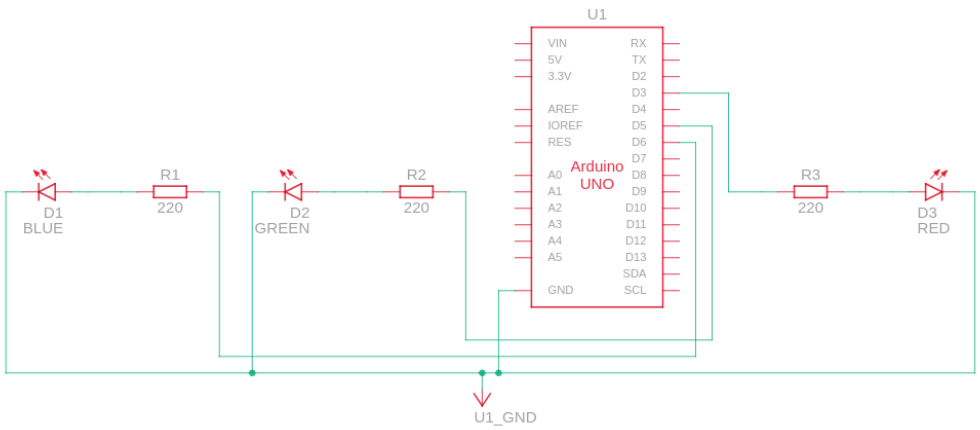
РЕАЛІЗАЦІЯ ПРОЄКТУ

Для реалізації розглянемо спочатку як працюють окремі елементи, що будуть використані у проєкті.

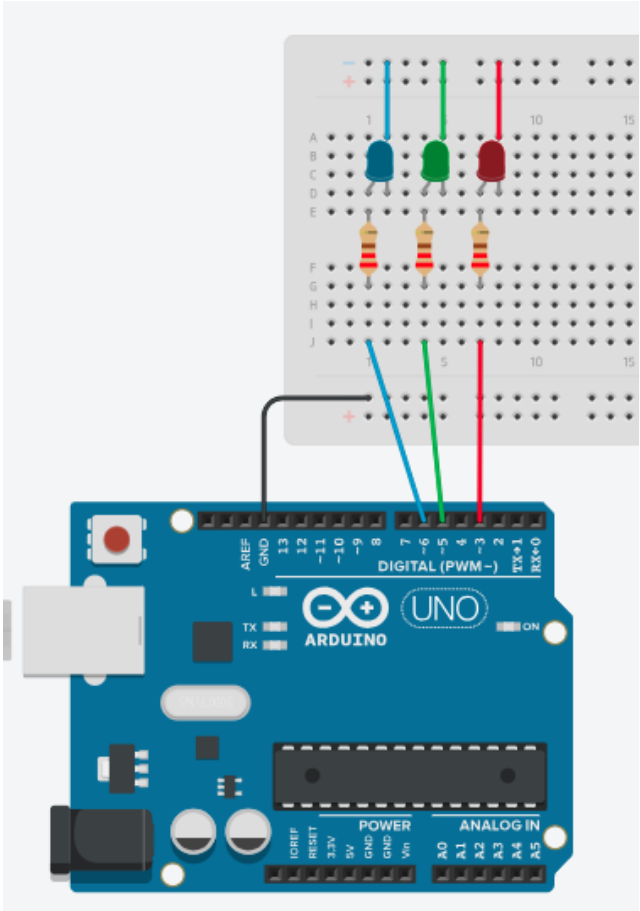
Для моделювання використовується макетна плата **BREADBOARD MB-10**, особливості функціонування якої можна знайти в [1, 2]. З сімейств Arduino використовується платформа **Arduino-UNO**, основі принципи роботи якої також описано в [1–3].

Розглянемо спочатку підключення трьох світлодіодів. Схема їхнього підключення передбачає ще 3 резистори, що з'єднані послідовно з кожним світлодіодом. Позитивні контакти від кожного зі світлодіодів під'єднаємо до цифрових виходів, які підтримують ШІМ (широтно-імпульсну модуляцію). На платформі **Arduino** вони позначені ~ (тильдою).

Схема підключення має вигляд



або візуально

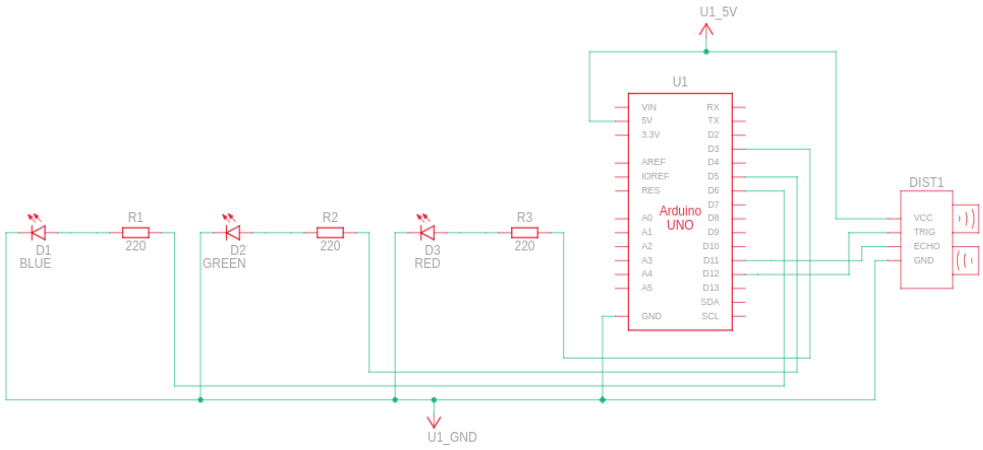


Тут використано резистори на 220 Ом. Код, що дає змогу засвітити ці світлодіоди, має вигляд:

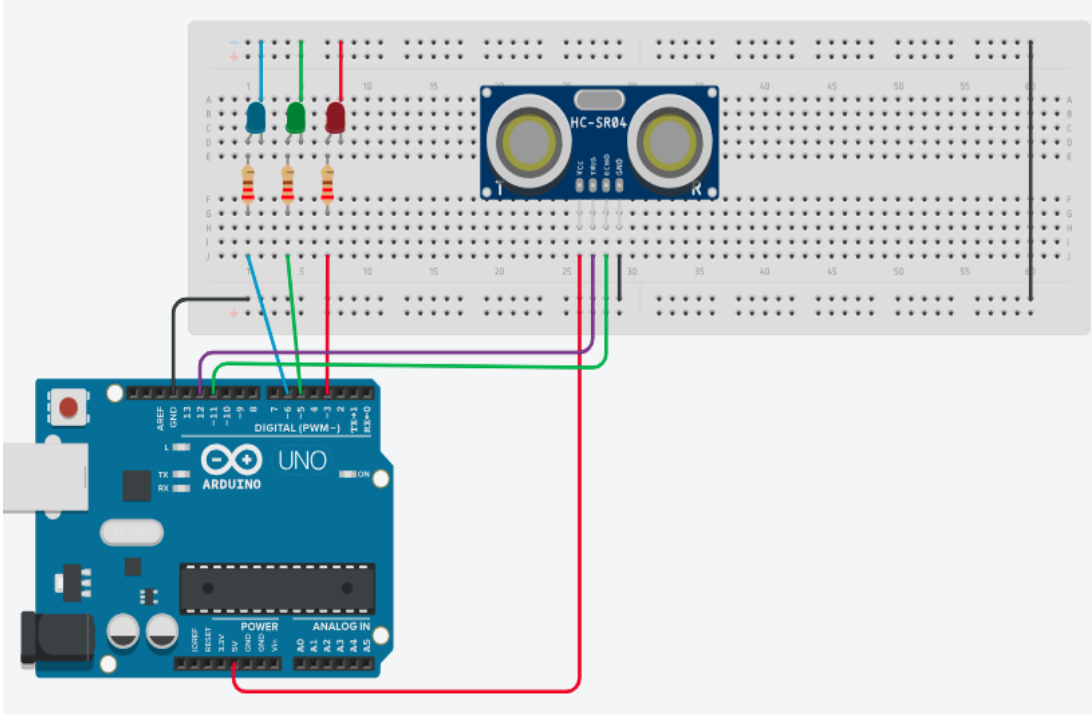
```
const byte redA = 3;
const byte greenA = 5;
const byte blueA = 6;

void setup() {
  pinMode(redA, OUTPUT);
  pinMode(greenA, OUTPUT);
  pinMode(blueA, OUTPUT);
}

void loop() {
  analogWrite(redA, 100);
  analogWrite(greenA, 100);
  analogWrite(blueA, 5);
}
```

Використання ШІМ виходів дасть змогу керувати яскравістю світлодіодів. У прикладі коду вище це реалізовано за допомогою функції **analogWrite(_pin_, value)**. Тут **value** можу набувати значення від 0 до 255 (0 і 5 В відповідно). Для прикладу тут вказано 100.

Тепер дослідимо роботу ультразвукового датчика відстані. Ми використовуємо модель HC-SR04. Це точний датчик відстані, який має змогу вимірювати дистанції у діапазоні (0,1500 мм). Похибка відстані становить 3 мм. Підключимо цей датчик до попередньої схеми.



Візуально це виглядає так:

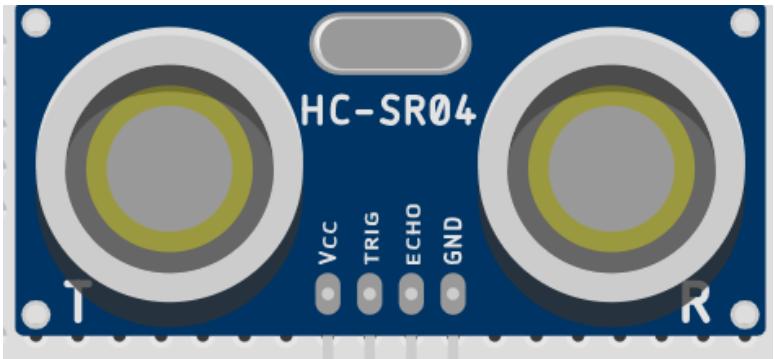


Код, що дає змогу визначати відстань, набуде вигляду:

```
const byte redA = 3;
const byte greenA = 5;
const byte blueA = 6;

                    const byte echoPin = 11;
                    const byte trigPin = 12;

void setup() {
  pinMode(redA, OUTPUT);
   pinMode(greenA, OUTPUT);
  pinMode(blueA, OUTPUT);

                     pinMode(echoPin, INPUT);
                     pinMode(trigPin, OUTPUT);
                     Serial.begin(9600);
}

void loop() {

                     long duration, cm;
                     digitalWrite(trigPin, LOW);
                     delayMicroseconds(2);
                     digitalWrite(trigPin, HIGH);
```

```
                    delayMicroseconds(10);
                    digitalWrite(trigPin, LOW);

                    duration = pulseIn(echoPin, HIGH);
                    Serial.println(durationToCentimeters(duration));

   analogWrite(redA, 100);
  analogWrite(greenA, 100);
  analogWrite(blueA, 5);
}


long durationToCentimeters(long duration)
{
  // швидкість звуку становить 343 м/с або  1/343 = 0,0029 с/м = 29 мкс/см
  // ми ділимо на 2, бо враховуємо час звучу до об'єкта і назад.
  return duration / 29 / 2;
}
```

Фрагменти коду, що ми додали у loop, змістили праворуч. Також додали функцію **durationToCentimeters**. Проаналізуємо ці зміни. Як видно з рисунка, ультразвуковий датчик має 4 виходи:



1) **Vcc** – підключається напруга 5В;

2) **GND** – підключається на “землю”;

3) **TRIG** – контакт, який має працювати у режимі **OUTPUT**. На цей контакт подається напруга, яка перетворюється на ультразвуковий сигнал, що випромінюється датчиком;

4) **ECHO** – контакт, що працює в режимі **INPUT**. Цей контакт передає на платформу Arduino інформацію, про відбитий звуковий сигнал.

Це у коді реалізовано

const byte echoPin = 11

const byte trigPin = 12

та

pinMode(echoPin, INPUT)

pinMode(trigPin, OUTPUT).

Також додано **Serial.begin(9600)**, щоб ініціалізувати виведення інформації через **Serial Monitor**.

У блоці **loop** оголошено змінні duration, cm та відправляються сигнали з інтервалом 2 і 10 мікросекунд.

```
digitalWrite(trigPin, LOW)
delayMicroseconds(2)
digitalWrite(trigPin, HIGH)
delayMicroseconds(10)
digitalWrite(trigPin, LOW).
```

Отримавши на контакті **trigPin** значення **HIGH**, ультразвуковий датчик випромінює звукову хвилю. Далі використовуємо конструкцію

duration = pulseIn(echoPin, HIGH).

Функція зчитує тривалість імпульсу (**HIGH** або **LOW**) на виведенні. Наприклад, якщо задане значення (значення) – **HIGH**, функція **PulseIn**() очікує виявлення на виведенні сигналу **HIGH**, потім засікає час і очікує перемикання виводу в стан **LOW**, після чого зупиняє відлік часу. Функція повертає тривалість імпульсу в мікросекундах, або 0 у разі відсутності імпульсу протягом певного таймауту. Отже, результатом функції є час, за який звук проходить відстань від датчика до об'єкта і назад.

Наступним кроком виводимо значення відстані у сантиметрах.

Serial.println(durationToCentimeters(duration)).

Функція durationToCentimeters приймає агрумент тривалість часу у мікросекундах, а повертає відстань. Логіка роботи функції полягає у

перетворенні часу проходження звукового сигналу у відстань, знаючи швидкість звуку.

Провівши випробовування, зазначимо, що у порт виводиться відстань, як показано на рисунку

Тепер організуємо вмикання світла при русі рукою. Ідея, яку ми закладемо у код, полягає у тому, що при потраплянні об'єкта на відстань, меншу 10 сантиметрів, має спрацьовувати вимикач. І ще потрібно, щоб стан (діоди увімкнуті або вимкнуті) змінювався тільки один раз при наближенні об'єкта до сенсора. Світлодіоди мають залишатися у цьому стані доти, доки об'єкт розміщений у діапазоні, і змінити стан лише після того, як об'єкт покине діапазон, і потім знову з'явиться у ньому. Робочу версію коду, що це реалізовує, подано нижче:

```
const byte redA = 3;
const byte greenA = 5;
const byte blueA = 6;

const byte echoPin = 11;
const byte trigPin = 12;

byte light = false;       // Стан світла
```

```
bool objectPresent = false; // Чи є об'єкт у діапазоні
byte redLevel = 250;
byte greenLevel = 250;
byte blueLevel = 250;

const int threshold = 10;   // Поріг діапазону (10 см)

void setup() {
  pinMode(redA, OUTPUT);
  pinMode(greenA, OUTPUT);
  pinMode(blueA, OUTPUT);
  pinMode(echoPin, INPUT);
  pinMode(trigPin, OUTPUT);
  Serial.begin(9600);
}

void loop() {
  long duration, cm;

  // Відправляємо ультразвуковий імпульс
  digitalWrite(trigPin, LOW);
  delayMicroseconds(2);
  digitalWrite(trigPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin, LOW);

  // Вимірюємо тривалість сигналу та обчислюємо відстань
  duration = pulseIn(echoPin, HIGH);
  cm = durationToCentimeters(duration);  // Відстань в см

  if (cm < threshold && !objectPresent) {
    // Якщо об'єкт потрапив у діапазон і не був у ньому раніше
    light = !light;  // Перемикаємо стан світла
    objectPresent = true;  // Вказуємо, що об'єкт зараз у діапазоні
  }
  else if (cm >= threshold && objectPresent) {
    // Якщо об'єкт вийшов за межі діапазону
    objectPresent = false;  // Вказуємо, що об'єкта більше немає в діапазоні
  }

  // Контроль світла
  if (light) {
    analogWrite(redA, redLevel);
    analogWrite(greenA, greenLevel);
    analogWrite(blueA, blueLevel);
  } else {
    analogWrite(redA, 0);
    analogWrite(greenA, 0);
    analogWrite(blueA, 0);
  }
```

```
  delay(100);  // Невелика затримка для стабільності
}

long durationToCentimeters(long duration) {
  // Швидкість звуку: 343 м/с або 29 мкс/см (ділимо на 2 через зворотній шлях звуку)
  return duration / 29 / 2;
}
```

Також робочу версію проєкту можна переглянути за посиланням: https://drive.google.com/file/d/1BRT1GhFj1nXb83EAR0oIHDG-y_rHIv_9

Якщо замість світлодіодів використати реле, то можна підключити реальні лампи.

2. STEM-проєкт “Вимірювання відстані”

Мета:

Метою проєкту є створення прототипу приладу, що вимірює відстань та виводить її на рідкокристалічний екран.

Обладнання:

1. Макетна плата.
2. Платформа Arduino.
3. Ультразвуковий датчик.
4. Рідкокристалічний дисплей.
5. Модуль I2C.
6. З'єднувальні проводи.

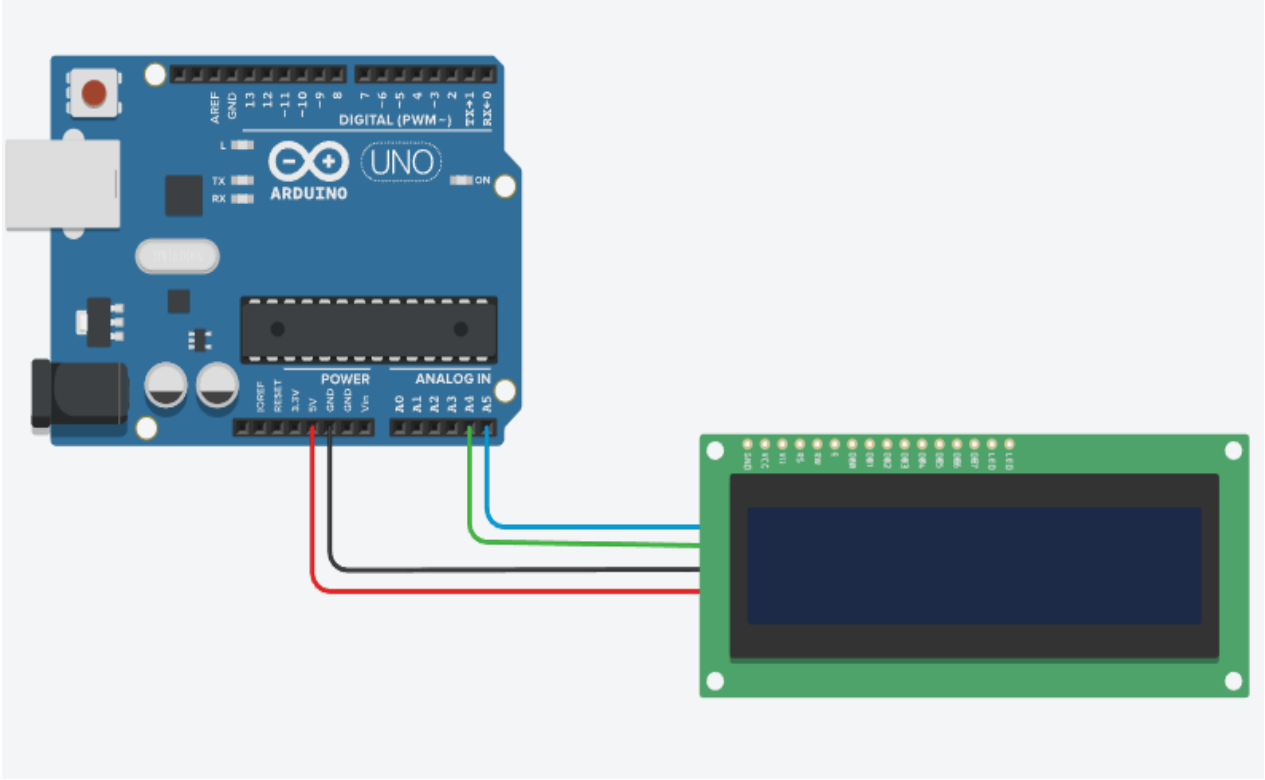
Реалізація проєкту

Для реалізації розглянемо спочатку, як працюють окремі елементи, що будуть використані у проєкті.

Для моделювання використовується макетна плата **BREADBOARD MB-10**, платформа **Arduino-UNO**, ультразвуковий датчик, як і в проєкті 1. Для виведення відстані на екран використаємо рідкокристалічний **LCD 1602 I2C** символьний дисплей 16x2. Особливості роботи безпосередньо з дисплеєм наведено у [2]. Однак для зручності роботи ми використаємо модуль **I2C**.

Модуль **I2C** для рідкокристалічного дисплея (**LCD**) дозволяє легко підключити дисплей до Arduino через шину I2C, використовуючи всього два проводи (**SDA** і **SCL**), замість 6–8 проводів, необхідних для звичайного підключення. Він полегшує використання дисплеїв з контролерами **HD44780** і зменшує кількість контактів, що використовуються Arduino. Модуль має вбудований потенціометр для налаштування контрастності

дисплея. Завдяки бібліотеці **LiquidCrystal_I2C** можна просто відправляти текстові рядки на дисплей. **I2C**-адресу модуля можна змінювати, що дозволяє підключати декілька дисплеїв одночасно. Візуально схема підключення має вигляд:

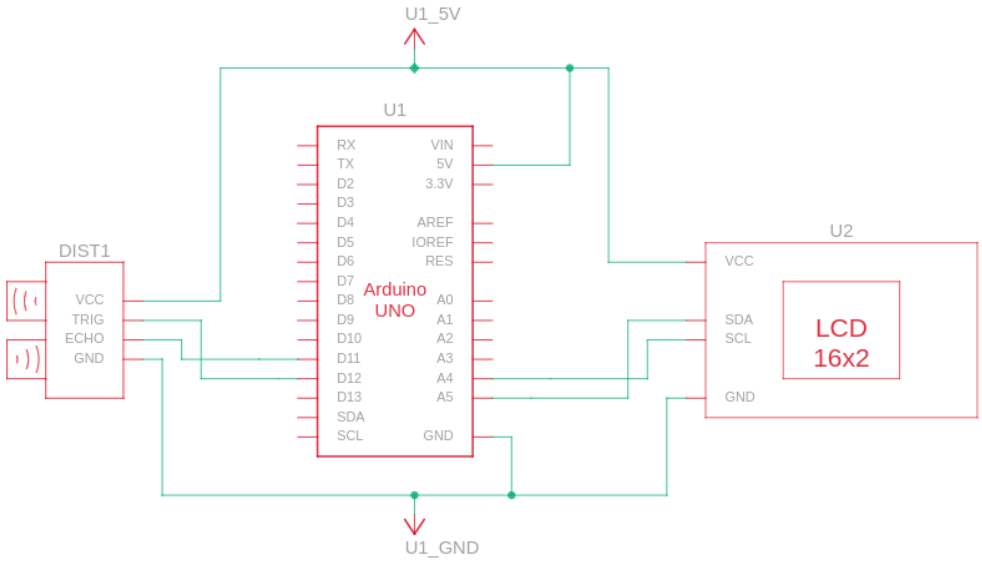


Контакти треба з'єднати так, як показано у таблиці.

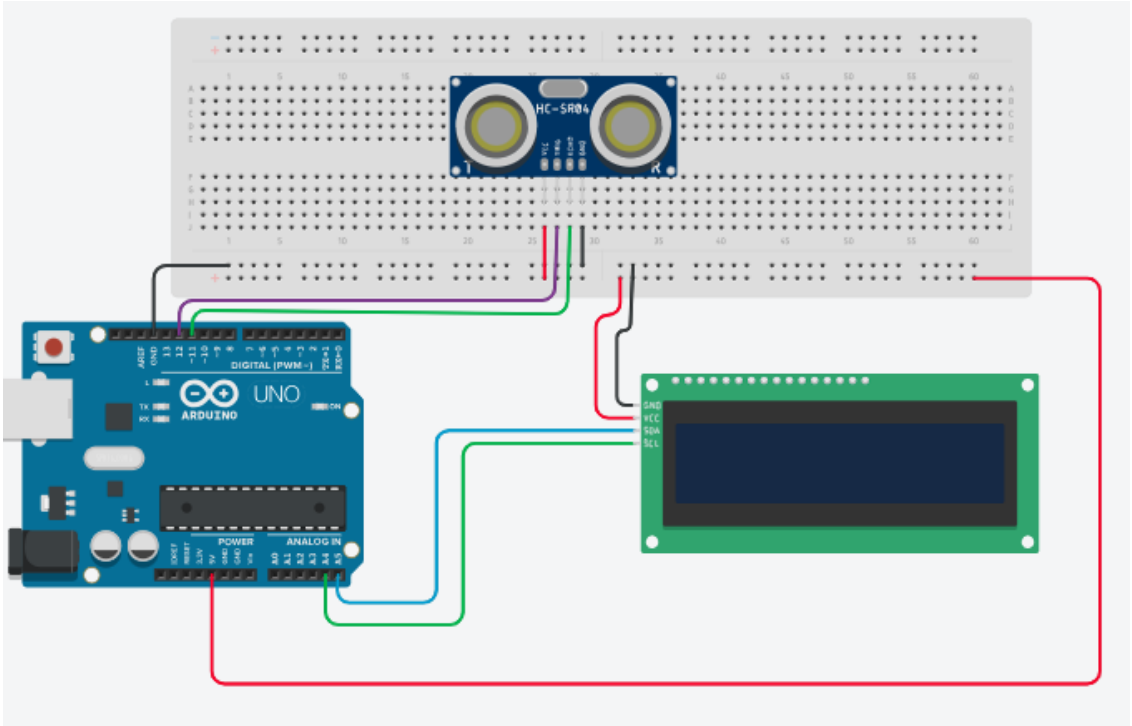
Arduino Uno	**LCD1602 I2C**
Живлення 5V	5V
GND (“земля”)	GND
Аналоговий контакт A4 (SCL)	SCL
Аналоговий контакт A5 (SDA)	SDA

Відповідні позначення є на модулі **I2C**. Також для роботи з модулем потрібна бібліотека **LiquidCrystal_I2C**, яку можна завантажити з доступних джерел (наприклад, **github**).

Принципова схема підключення має вигляд:



А візуально це виглядає так:



Щодо програмної реалізації коду, то частина, що стосується ультразвукового датчика, є аналогічною до першого проєкту. Інша частина коду стосується виведення на екран. Загалом програмний код має вигляд:

```
#include "LiquidCrystal_I2C.h" // підключення бібліотеки

LiquidCrystal_I2C lcd(0x27,16,2);

const byte echoPin = 11;
const byte trigPin = 12;


void setup() {
  pinMode(echoPin, INPUT);
  pinMode(trigPin, OUTPUT);

  lcd.init();     // ініціалізація дисплею
  lcd.backlight();
}

void loop() {
  long duration, cm;

  // Відправляємо ультразвуковий імпульс
  digitalWrite(trigPin, LOW);
  delayMicroseconds(2);
  digitalWrite(trigPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin, LOW);

  // Вимірюємо тривалість сигналу та обчислюємо відстань
  duration = pulseIn(echoPin, HIGH);
  cm = durationToCentimeters(duration);  // Відстань в см

  lcd.clear();
  lcd.print(String("Distance:" + String(cm) + " cm" ));

  delay(1000);  // затримка для стабільності
}

long durationToCentimeters(long duration) {
  // Швидкість звуку: 343 м/с або 29 мкс/см (ділимо на 2 через зворотній шлях звуку)
  return duration / 29 / 2;
}
```

Відеорезультат реалізації можна переглянути на відео за посиланням: https://drive.google.com/file/d/1BSx4mAYCoXS1qrOMMFI_dQu2vctpKUoF .

3. STEM-проєкт “Парктроник”

Мета:

Метою проєкту є створення прототипу парктроника, який дає видає звуковий сигнал і засвічуються світлодіоди при наближенні до об'єкта.

Обладнання:

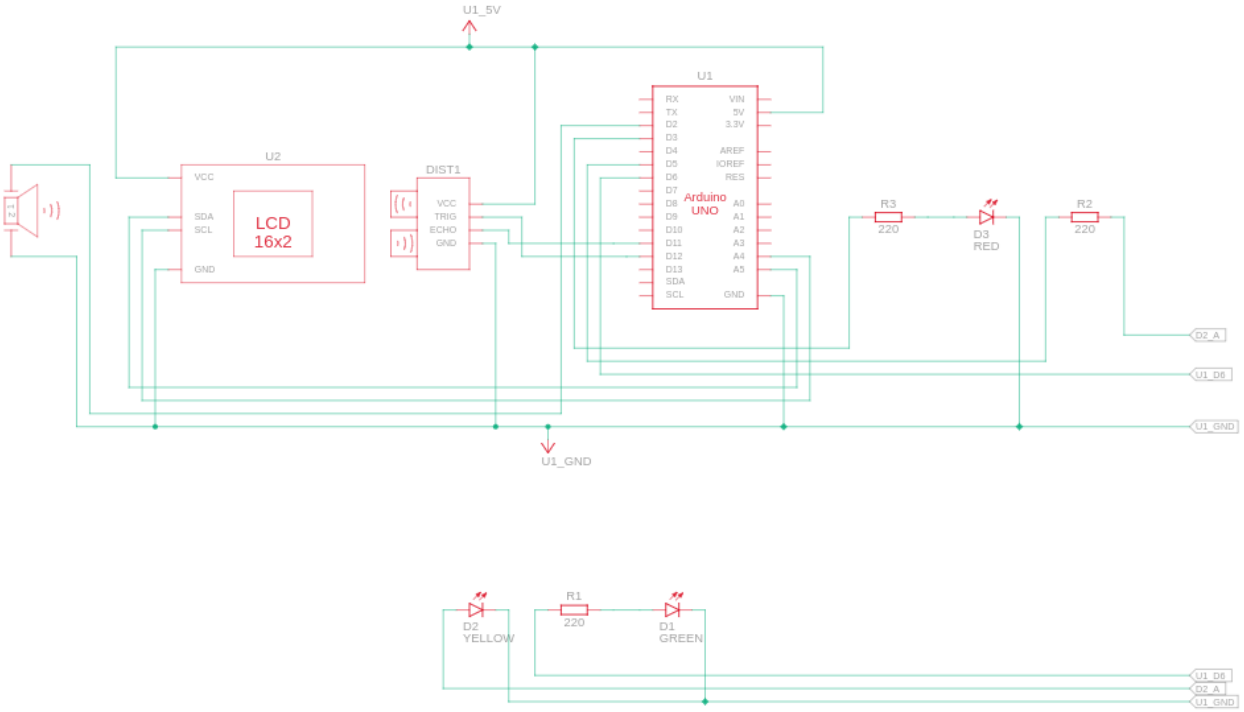
1. Макетна плата.
2. Платформа Arduino.
3. Ультразвуковий датчик.
4. Рідкокристалічний дисплей.
5. Модуль I2C.
6. Три світлодіоди.
6. Елемент **piezo**.
7. З'єднувальні проводи.

Реалізація проєкту

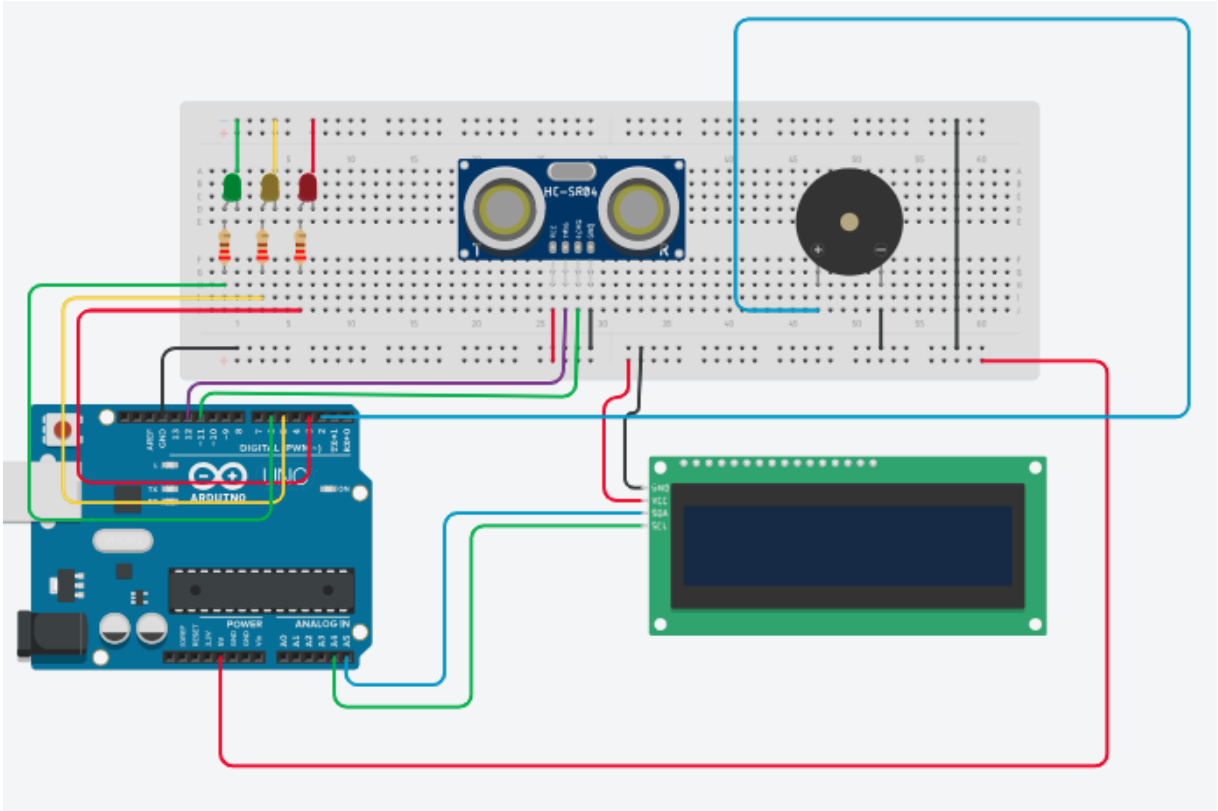
Для реалізації розглянемо спочатку, як працюють окремі елементи, що будуть використані у проєкті.

Для моделювання використовується макетна плата **BREADBOARD MB-10**, платформа **Arduino-UNO**, ультразвуковий датчик, три світлодіоди, які підключимо через резистори, як і в проєкті 1 і 2. Для організації звукового сигналу використаємо елемент **piezo**, який описано у [2].

Використаємо схеми підключення світлодіодів, резисторів, дисплею та ультразвукового датчика з проєктів 1 і 2. Елемент **piezo** додамо, “-” підключивши на “землю”, а “+” підключимо на цифровий контакт 2 Arduino. Принципова схема підключення має вигляд:



А візуально так:



Програмну реалізацію подано нижче:

```
#include "LiquidCrystal_I2C.h"

LiquidCrystal_I2C lcd(0x27,16,2);

const byte echoPin = 11;
const byte trigPin = 12;
const byte redA = 3;
const byte yellowA = 5;
const byte blueA = 6;
const byte piezo = 2;

long prevCm = 0;          // Для зберігання попереднього значення відстані
unsigned long prevMillis = 0; // Час останнього оновлення екрана
const long updateInterval = 200; // Інтервал оновлення екрана (200 мс)

void setup() {
  pinMode(echoPin, INPUT);
  pinMode(trigPin, OUTPUT);

  pinMode(redA, OUTPUT);
  pinMode(yellowA, OUTPUT);
  pinMode(blueA, OUTPUT);

  pinMode(piezo, OUTPUT);

  lcd.init();
  lcd.backlight();
  lcd.setCursor(0, 0);
  lcd.print("Distance: ");
}

void loop() {
  long duration, cm;

  // Відправляємо ультразвуковий імпульс
  digitalWrite(trigPin, LOW);
  delayMicroseconds(2);
  digitalWrite(trigPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin, LOW);

  // Вимірюємо тривалість сигналу та обчислюємо відстань
  duration = pulseIn(echoPin, HIGH);
  cm = durationToCentimeters(duration);  // Відстань в см

  unsigned long currentMillis = millis();  // Поточний час

  // Оновлюємо екран, якщо минуло більше ніж updateInterval та зміна більше ніж 2 см
  if ((currentMillis - prevMillis >= updateInterval) && (abs(cm - prevCm) > 2)) {
```

```
   lcd.setCursor(10, 0); // Зміщення курсора до значення відстані
   lcd.print("     ");    // Очищаємо попереднє значення
   lcd.setCursor(10, 0);  // Встановлюємо курсор на початок поля для відстані
   if(cm > 50){
     lcd.print("50+");
   }else{
     lcd.print(cm);
   }
   lcd.print(" cm");
   prevCm = cm;  // Зберігаємо нове значення відстані
   prevMillis = currentMillis;  // Оновлюємо час останнього оновлення
  }

  // Управління світлодіодами
  if(cm < 10){
   digitalWrite(redA, HIGH);
   digitalWrite(yellowA, HIGH);
   digitalWrite(blueA, HIGH);
   tone(piezo, 1000);  // Піщання на високій частоті
  } else if (cm < 20){
   digitalWrite(blueA, HIGH);
   digitalWrite(yellowA, HIGH);
   digitalWrite(redA, LOW);
   tone(piezo, 500);   // Піщання на нижчій частоті
  } else {
   digitalWrite(blueA, HIGH);
   digitalWrite(yellowA, LOW);
   digitalWrite(redA, LOW);
   noTone(piezo);      // Вимикаємо звук
  }

  delay(10);  // Невелика затримка для стабільної роботи (можна прибрати при
необхідності)
}

long durationToCentimeters(long duration) {
  // Швидкість звуку: 343 м/с або 29 мкс/см (ділимо на 2 через зворотній шлях звуку)
  return duration / 29 / 2;
}
```

У коді використано ідеї:

1) умовне оновлення дисплея: екран оновлюється тільки тоді, коли зміна відстані перевищує 2 см (**abs**(cm - prevCm) > 2);

2) інтервал оновлення: використовується таймер на основі функції **millis()**, щоб оновлювати екран не частіше, ніж кожні 200 мс (**updateInterval**).

3) відстані: якщо відстань більше 50 см, ми не враховуємо подальшу зміну, тобто виводимо на екран “50+”.

Щоб п'єзоелемент змінював інтенсивність писку залежно від відстані, зроблено так:

1) для відстані менше ніж 10 см п'єзоелемент повинен пищати сильніше (більша частота);

2) для відстані від 10 до 20 см п'єзоелемент повинен пищати слабше (менша частота);

3) якщо відстань більша за 20 см, п'єзоелемент не повинен працювати. Для цього використано функцію **tone()** для керування частотою звуку п'єзоелемента та функцію **noTone()** для вимкнення звуку.

Аналогічно для тих самих відстаней реалізовано світіння світлодіода:

1) світяться усі світлодіоди (синій, жовтий і червоний), якщо відстань менша 10 см;

2) світяться жовтий і синій світлодіоди, якщо відстань у діапазоні (10 см, 20 см);

3) світить тільки синій світлодіод у всіх інших випадках.

Переглянути відео робочого проєкту можна за посиланням: https://drive.google.com/file/d/1FHwrBEtbR0YQAsm2vK50eo-W6Fp0Cm9w/

Цей проєкт можна використовувати як парктроник для автомобіля, що допомагає водію визначати відстань до перешкод позаду. Ультразвуковий датчик вимірює відстань до об'єкта, а світлодіоди і п'єзоелемент сигналізують про наближення. Коли відстань менше 10 см, лунає гучний сигнал і загоряються всі світлодіоди. Якщо відстань між 10 і 20 см, сигнал слабший, а частина світлодіодів вимикається. При відстані більше 20 см сигнал не подається, що означає безпечну відстань. Для реальних ситуації з автомобілем ці відстані можна змінити на більш практичні значення.

4. STEM-проєкт “Дистанційний вмикач світла”

Мета:

Метою проєкту є створення прототипу дистанційного вмикача світла, який працює з використанням пульта. Також за допомогою пульта включаються різні лампи.

Обладнання:

1. Макетна плата.
2. Платформа Arduino.
3. Інфрачервоний пульт з приймачем
4. Два світлодіоди, два резистори.
5. З'єднувальні проводи.

Реалізація проєкту

Для реалізації розглянемо спочатку, як працює інфрачервоний пульт. Інфрачервоний (ІЧ) пульт для **Arduino** уможливлює бездротове керування різними пристроями або модулями за допомогою ІЧ-сигналів. Тобто цей пульт використовує ІЧ промені для передачі команд від пульта до приймача, підключеного до **Arduino**. ІЧ-пульт використовує інфрачервоне випромінювання для передачі даних. Коли натискається кнопка на пульті, він генерує модуляційний ІЧ-сигнал, який кодує конкретну команду. ІЧ-ресивер підключений до **Arduino** приймає цей сигнал, розшифровує його і дозволяє Arduino виконувати відповідні дії.

Щоб підключити ІЧ-ресивер до **Arduino**, треба

1) **VCC** ІЧ-ресивера підключити до 5V на **Arduino**.

2) **GND** ІЧ-ресивера підключити до **GND** на **Arduino**.

3) **OUT** ІЧ-ресивера підключити до цифрового піну (наприклад, **D8**) на **Arduino**.

Для спрощення роботи з ІЧ-сигналами використовується бібліотека **IRremote**. Вона дає змогу легко приймати та відправляти ІЧ-сигнали. Однак різні пульти можуть по-різному кодувати / декодувати сигнали. Тому для того, щоб визначити коди сигналів, треба спочатку у тестовому режимі підключити приймач і за допомогою серійного порту **Arduino** визначити їх коди.

Нижче наведено детальне пояснення того, як працює ІЧ-пульт з **Arduino**, включаючи необхідні компоненти, принципи роботи та приклади коду.

```
#include <IRremote.hpp> // підключення бібліотеки


const byte IR_RECEIVE_PIN = 8;

void setup() {
      IrReceiver.begin(IR_RECEIVE_PIN); // запускаємо аналіз даних з інфрачервоного давача, підключеного до 8 піну
      Serial.begin(9600); // ініціалізація серійного порту
}

void loop() {

   if (IrReceiver.decode()) { // Розшифровуємо сигнал пульта
       IrReceiver.resume(); // Продовжуємо отримувати сигнали
       int command = IrReceiver.decodedIRData.command; // та зберігаємо його у десятковому форматі у змінну command
       Serial.println(command);
   }
}
```

У нашому випадку, використовуючи пульт, отримано такі коди клавіш пульта: “1” – код 69, “2” – код 70, “4” – код 68, “5” – код 64. Використаємо їх для налаштування вмикання та вимикання світла за допомогою пульта. Доб'ємося такого ефекту: при натисканні на

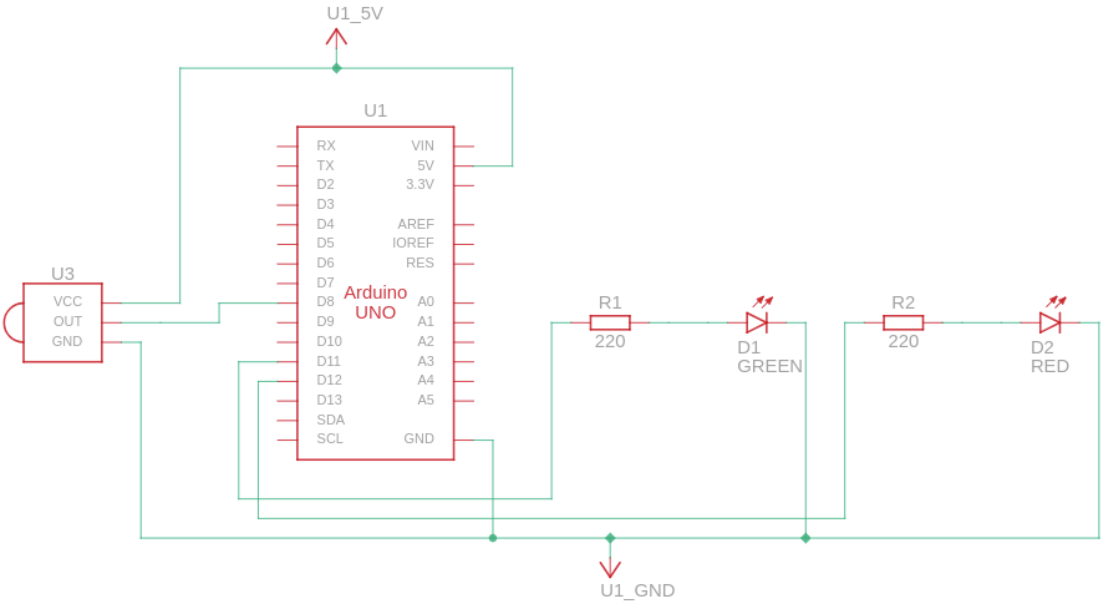
“1” – засвічується червоний світлодіод,

“2” – засвічується зелений світлодіод,

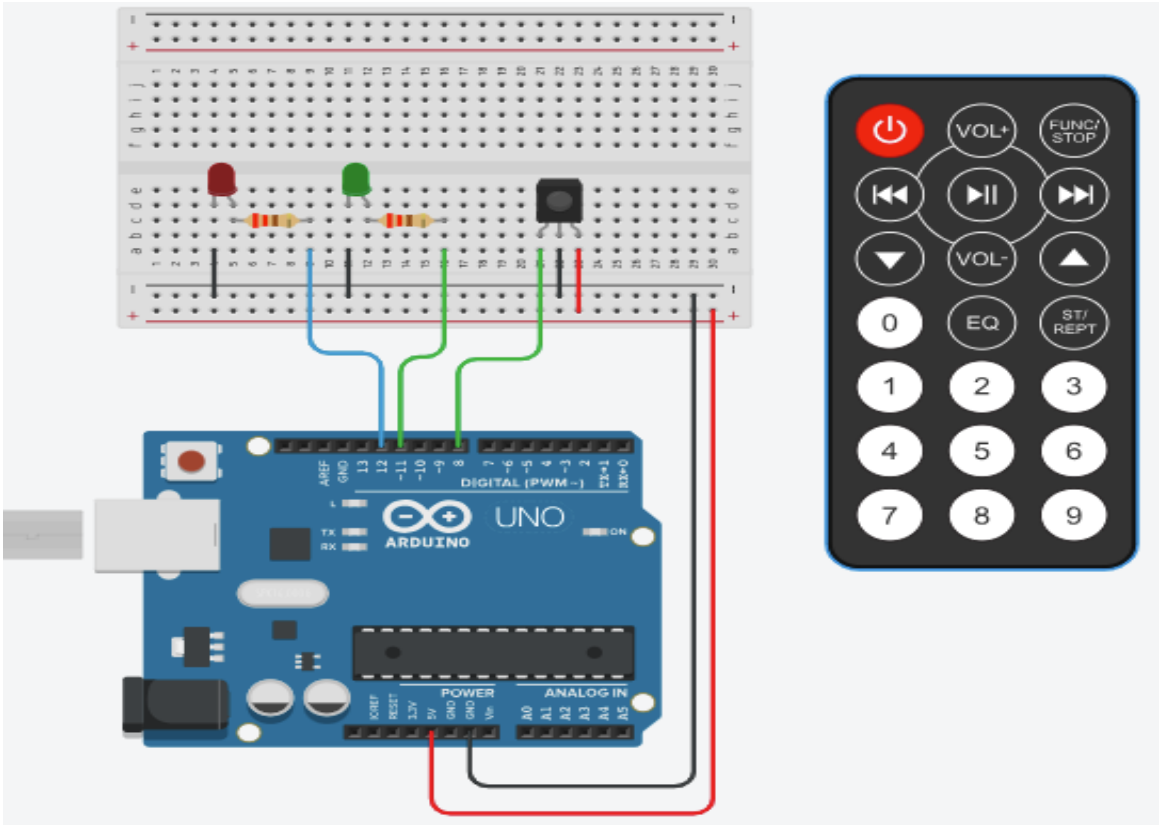
“4” – гасне червоний світлодіод,

“5” – гасне зелений світлодіод.

Для реалізації використано принципову схему, що наведено нижче:



А візуально схема виглядає так:



Програмна реалізація описаного способу керування світлодіодами має вигляд:

```
#include <IRremote.hpp>

const byte IR_RECEIVE_PIN = 8;
const byte RED_LED = 12; // константа  RED_LED і її значення 12
const byte GREEN_LED = 11; // константа GREEN_LED і її значення 11

int redLedState = LOW; // у змінній redLedState задаємо режим роботи світлодіода
int greenLedState = LOW; // у змінній greenLedState задаємо режим роботи світлодіода

void setup() {
  IrReceiver.begin(IR_RECEIVE_PIN); // запускаємо аналіз даних з інфрачервоного давача, підключеного до 8 піну
  pinMode(RED_LED, OUTPUT); // конфігуруємо 12 пін як вихід
  pinMode(GREEN_LED, OUTPUT); // конфігуруємо 11 пін як вихід
  Serial.begin(9600);
}

void loop() {

  if (IrReceiver.decode()) { // Розшифровуємо сигнал пульта
    IrReceiver.resume(); // Продовжуємо отримувати сигнали
    int command = IrReceiver.decodedIRData.command; // та зберігаємо його у десятковому форматі у змінну command
    Serial.println(command);
    switch (command) { //Перевіряємо, які команди пульта відповідають потрібним нам кнопкам. Задаємо дії відповідно до отриманого сигналу, що провіяв

      case 69: { // Виконується, коли command дорівнює 69 (число 1)
        redLedState = HIGH; // Переключаємо стан світлодіоду redLedState на HIGH
        digitalWrite(RED_LED, redLedState); // Оновлюємо стан світлодіоду RED_LED
        break; //  break - це вихід із оператора case; використовується в кінці кожного case
      }

      case 70: { // Виконується, коли command дорівнює  70 (число 2)
        greenLedState = HIGH; // Переключаємо стан світлодіоду на HIGH
        digitalWrite(GREEN_LED, greenLedState); // Оновлюємо стан світлодіоду GREEN_LED
        break;
      }

      case 68: { // Виконується, коли command дорівнює 68 (число 4)
        redLedState = LOW; // Переключаємо стан світлодіоду redLedState на HIGH
        digitalWrite(RED_LED, redLedState); // Оновлюємо стан світлодіоду RED_LED
        break; //  break - це вихід із оператора case; використовується в кінці кожного case
      }
```

```
    case 64: { // Виконується, коли command дорівнює  64 (число 5)
      greenLedState = LOW; // Переключаємо стан світлодіоду на HIGH
      digitalWrite(GREEN_LED, greenLedState); // Оновлюємо стан світлодіоду
GREEN_LED
      break;
    }
  }
 }
}
```

Запропонована модель керування світлодіодами може бути використана для керування дистанційно освітленням, за умови використання реле замість світлодіодів.

Переглянути відео робочого проєкту можна за посиланням: https://drive.google.com/file/d/1qiAMKss2QF-1NOt5rd1WGhPBWmAc4536.

5. STEM-проєкт “Розумна оранжерея”

Мета:

Метою проєкту є створення прототипу системи управління оранжереєю.

Обладнання:

1. Макетна плата.
2. Платформа **Arduino**.
3. Світлодіод, резистор.
4. П’єзоелемент.
5. Дисплей на основі **I2C**.
5. Датчик вологості ґрунту.
6. Датчик температури і вологості
7. Реле.
8. Фоторезистор.
9. З'єднувальні проводи.

Реалізація проєкту

Розумна оранжерея працює завдяки комплексній системі датчиків та компонентів, що взаємодіють через платформу **Arduino**. Фоторезистор постійно відстежує рівень освітлення в теплиці. Якщо рівень освітленості падає нижче за оптимальний, фоторезистор надсилає сигнал на Arduino. Мікроконтролер аналізує цей сигнал і дає команду на замикання реле, що вмикає освітлення в теплиці. Це забезпечує постійну оптимальну кількість світла для рослин. Паралельно датчики температури та вологості повітря вимірюють ці параметри і надсилають дані на платформу **Arduino**. Значення температури і вологості виводяться на дисплей, що дає можливість у реальному часі контролювати мікроклімат у теплиці. Якщо

параметри виходять за межі встановлених норм, система може подати сигнал для регулювання. Окрім того, в ґрунт вмонтований датчик вологості, який постійно вимірює рівень вологи в субстраті. Коли рівень вологості ґрунту стає занадто низьким, датчик передає сигнал на **Arduino**. Мікроконтролер активує червоний світлодіод, який служить індикатором низької вологості. Також задіюється п’єзоелемент, що подає звуковий сигнал, нагадуючи про необхідність поливу. Таким чином, система не лише підтримує оптимальні умови для росту рослин, але й забезпечує автоматичний контроль за ними. Завдяки цим елементам оранжерея може функціонувати майже автономно. Людина отримує необхідні сигнали лише тоді, коли виникає потреба у втручанні. Це спрощує догляд за рослинами та зменшує ризик помилок, пов'язаних з недостатнім або надмірним поливом, чи поганими умовами освітлення.

У попередніх проєктах ми розглядали способи підключення і світлодіодів, і п’єзодинаміка, і дисплея. Зараз розглянемо роботу датчика температури і вологості. Це реалізовано в одному пристрої, що вимірює і температуру, і вологість повітря. Датчик температури та вологості **DHT11**, реалізовано у вигляді модуля на платі. Його можна підключати безпосередньо до Arduino, без необхідності додаткового резистора (бо він уже вмонтований у плату).

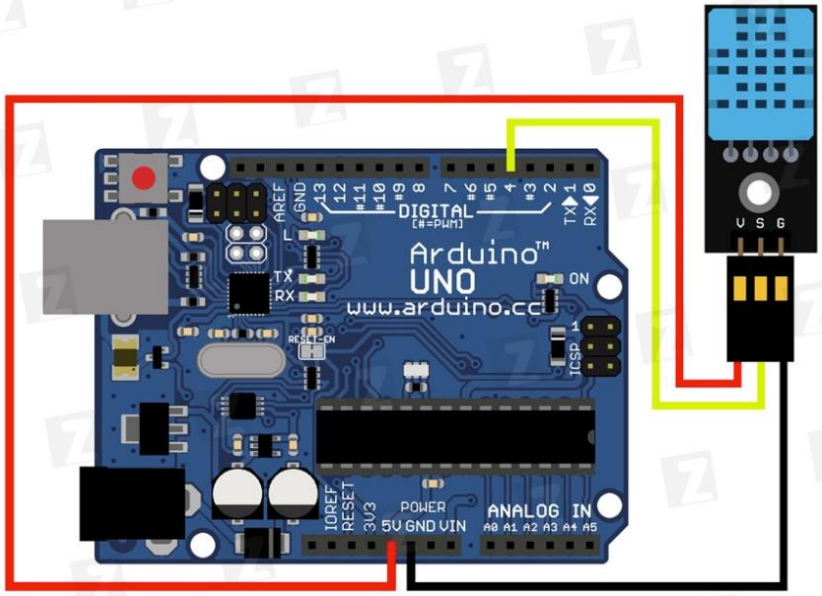
Модуль обладнаний триконтактним роз'ємом стандарту 2.54мм

G – підключається до виводу **GND**

V – підключається до виводу **+5V**

S – підключається до цифрового виводу (у прикладі **D4**)

Далі необхідно підключити відповідну бібліотеку, яка опрацьовує сигнали з датчика і перетворює їх у зрозумілі параметри вологості і температури.

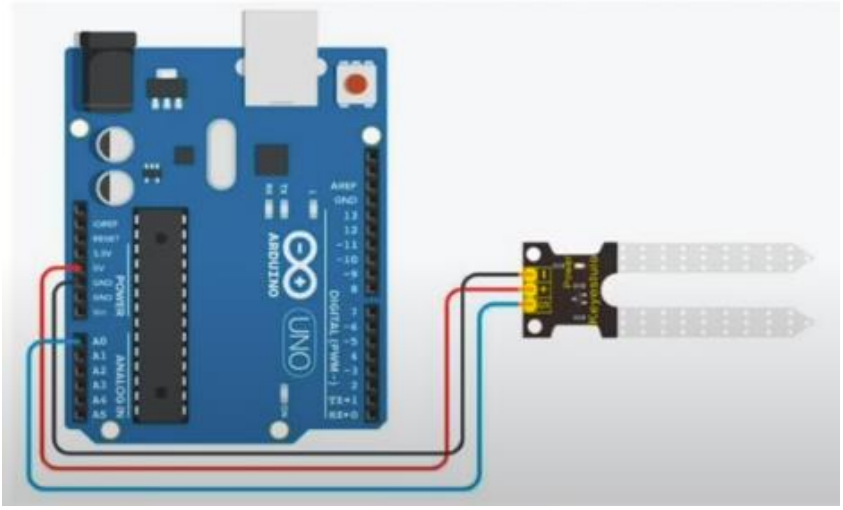


Приклад програми наведено нижче

```
#include "DHT.h"
#define DHT11_PIN 4     // підключаємо до 4 цифрового піну
#define DHTTYPE DHT11   // тип датчика DHT 11
DHT dht(DHTPIN, DHTTYPE); ///


void setup(){
   Serial.begin(9600); // ініціалізація серійного порту
   Serial.println(DHT11LIB_VERSION); // Версія бібліотеки
   Serial.println(); // порожній рядок
}

void loop(){
   int chk;
   chk = DHT.read(DHT11_PIN); // Зчитуємо дані з датчика
    // Перевіряємо чи все правильно опрацьовується. Якщо так, то нічого не виводимо
   switch (chk){
     case DHTLIB_OK:
        break;
     case DHTLIB_ERROR_CHECKSUM:
        Serial.println("Checksum error, \t");
        break;
     case DHTLIB_ERROR_TIMEOUT:
        Serial.println("Time out error, \t");
        break;
     default:
        Serial.println("Unknown error, \t");
        break;
    }
// Виводимо поточні значення температури і вологості повітря
     Serial.print("Humidity = ");
```

```
    Serial.print(DHT.humidity, 1); // вологість

    Serial.print(", Temp = ");
    Serial.println(DHT.temperature,1); // температури у цельсіях

    delay(1000);
}
```

Тепер опишемо принцип роботи датчика вологості ґрунту, який для **Arduino** працює за принципом вимірювання електропровідності ґрунту між двома контактами. Коли рівень вологи в ґрунті високий, провідність збільшується, і датчик генерує вищий аналоговий сигнал. Якщо ґрунт сухий, провідність зменшується, і сигнал стає нижчим. **Arduino** отримує цей сигнал та аналізує його, визначаючи рівень вологості. На основі цього значення система може приймати рішення, наприклад, активувати полив або подати сигнал про необхідність втручання. Візуально схема підключення датчика подана на рисунку



З рисунка видно, що схема підключення досить проста, тому на ній зупинятися не будемо.

Розглянемо принцип роботи реле. Реле для **Arduino** працює як електронний перемикач, який дає змогу керувати високовольтними або потужними пристроями за допомогою низьковольтного сигналу від

мікроконтролера. Коли на вхідну котушку реле подається сигнал від **Arduino**, утворюється магнітне поле, яке замикає або розмикає контакти реле. Це дозволяє керувати підключеними до реле приладами, такими як лампи чи насоси, через **Arduino**, не перевантажуючи його струмами високої потужності. Таким чином, реле забезпечує безпечне управління потужними пристроями у системах автоматизації. Подаючи високий або низький сигнал ми вмикаємо або розмикаємо високовольтні контакти реле.

Робота і принцип підключення світлодіода розглянуто у [2].

Отже, перейдемо безпосередньо до програмної реалізації проєкту

```
#include "LiquidCrystal_I2C.h"
LiquidCrystal_I2C lcd(0x27,20,4);


const int piezo = 2;
const int lightDiod = A0;
const int led = 3;
const int rele = 8;
float grunt = A1;


const float rate = 700; // рівень, який ми вважаємо достатнім для вмикання світла
const float gruntValue = 900;


#include "DHT.h"
#define DHTPIN 4     // 4 пін для отримання даних про температуру і вологість
#define DHTTYPE DHT11   // DHT 11
DHT dht(DHTPIN, DHTTYPE);


#include "notes.h"
// notes in the melody:
int melody[] = {
      NOTE_C4, NOTE_G3, NOTE_G3, NOTE_A3, NOTE_G3, 0, NOTE_B3, NOTE_C4
};
// note durations: 4 = quarter note, 8 = eighth note, etc.:
int noteDurations[] = {
     4, 8, 8, 4, 4, 4, 4, 4
};


byte isLed = 0;
```

```
void setup()
{
  lcd.init();lcd
  lcd.backlight();

  Serial.begin(9600);
  pinMode(piezo, OUTPUT);
  pinMode(lightDiod, INPUT);
  pinMode(led, OUTPUT);
  pinMode(rele, OUTPUT);
  pinMode(grunt, INPUT);
  dht.begin();

}

void loop()
{
  float light = analogRead(lightDiod);
  if(light < rate){
      digitalWrite(rele,0);
    }else{
      digitalWrite(rele,HIGH);
  }


  if(isLed == 1){
    digitalWrite(led,HIGH);
  }else{
    digitalWrite(led,LOW);
  }
  if(gruntValue < analogRead(grunt)){
    // сухо
    digitalWrite(led,HIGH);
    isLed = 1;
    playNotes();
  }else{
    // мокро
    digitalWrite(piezo,0);
    isLed = 0;
  }

   delay(200);

  float h = dht.readHumidity();
  float t = dht.readTemperature();

  if (isnan(t) || isnan(h)) {
    Serial.println("Failed to read from DHT");
  } else {
    lcd.clear();
```

```
   lcd.print(String("Humidity:" + String(h) + "%" ));
   lcd.setCursor(0,1);
   lcd.print(String("Temperat:" + String(t) + "*C" ));
   lcd.setCursor(0,2);
  }
}

void playNotes() {
  // Проходимо по кожній ноті в мелодії (всього 8 нот)
  for (int thisNote = 0; thisNote < 8; thisNote++) {
    // Для обчислення тривалості ноти беремо 1 секунду (1000 мс) і ділимо на тип ноти.
   // Наприклад, для чверті ноти: 1000 / 4, для восьмої ноти: 1000 / 8 і т.д.
   int noteDuration = 1000 / noteDurations[thisNote];
   // Видаємо сигнал на п'єзоелемент з відповідною частотою мелодії та тривалістю
ноти.
   tone(piezo, melody[thisNote], noteDuration);
   // Щоб відрізняти ноти одну від одної, встановлюємо мінімальний інтервал між
ними.
   // Тривалість паузи між нотами становить 30% від тривалості ноти:
   int pauseBetweenNotes = noteDuration * 1.30;
      // Затримка перед наступною нотою:
   delay(pauseBetweenNotes);
   // Зупиняємо програвання тону після кожної ноти:
   noTone(piezo);
  }
}
```

Для урізноманітнення видачі сигналу п’єзоелементом ми підключили бібліотеку, що дає змогу відтворювати звуки різної частоти і тривалості, імітуючи ноти. Відповідні частоти і тривалості відмічені у бібліотеці у вигляді нот. Також додана функція **playNotes**, що відтворює ці ноти.

Запропонований проєкт можна удосконалити шляхом підключення до реле лампи. Також можна організувати світіння цієї лампи у різні пори доби, не тільки керуючи освітленням. Ще одним важливим моментом удосконалення проєкту може бути відправлення сповіщень у реальному часі на мобільний пристрій через **wi-fi** адаптер або **bluetooth** адаптер.

6. STEM-проєкт “Радіочастотна ідентифікація”

Мета:

Метою проєкту є створення прототипу домофону.

Обладнання:

1. Макетна плата.
2. Платформа **Arduino**.
3. Світлодіод, резистор.
4. Модуль **RFID**.

Реалізація проєкту

Принцип організації домофону за допомогою **RFID**-модуля на основі Arduino полягає у створенні системи доступу з використанням **RFID**-карток або брелоків. Коли користувач підносить **RFID**-картку до зчитувача, модуль зчитує унікальний ідентифікатор (**UID**) картки. Arduino обробляє цей **UID** та порівнює його з **UID** карток, збережених у пам'яті системи. Якщо **UID** знайдено у базі дозволених карток, система активує реле, яке, зі свого боку, розблоковує замок дверей. Якщо **UID** не збігається із жодним записом у базі, замок не розблоковується, і доступ відхиляється.

RFID-модуль, підключений до **Arduino**, уможливлює безконтактне зчитування даних, що підвищує зручність і швидкість доступу. Для зберігання **UID** дозволених карток можна використовувати **EEPROM Arduino** або зовнішній модуль пам'яті. Система може бути розширена за допомогою додаткового дисплея, який відображає статус доступу, або звукового сигналу для підтвердження зчитування картки. Така схема дає змогу створити просту й надійну систему контролю доступу для домофону, яка може бути легко інтегрована в різні типи дверей. Розглянемо детальніше модуль.

RFID (Radio Frequency Identification) – це технологія, яка використовує радіохвилі для автоматичної ідентифікації об'єктів. **RFID** система складається з двох основних компонентів:

1. **RFID мітка** (мікрочип, який зберігає дані).
2. **RFID зчитувач** (пристрій, який взаємодіє з мітками).

RFID мітки можуть бути пасивними або активними. Пасивні мітки не мають власного джерела живлення і використовують енергію від радіохвиль, випромінюваних зчитувачем, тоді як активні мітки мають власне джерело живлення.

RFID-RC522 – це популярний зчитувач на базі чипа **MFRC522**, який працює на частоті 13.56 МГц і підтримує протокол ISO **14443A** (стандарт для безконтактних карток). Він використовується для зчитування **RFID**-карток, брелоків і інших міток.

3.3V – живлення напругою 3.3В;

RST – Скидання модуля (Reset);

GND – земля (Ground);

IRQ – Переривання (Interrupt);

MISO – Майстер-читач (Master In Slave Out), SPI;

MOSI – Майстер-записувач (Master Out Slave In), SPI;

SCK – Серійний годинник (Serial Clock), SPI;

SDA/SS – Вибір пристрою (Slave Select), SPI.

RFID модуль **RC522** є відмінним вибором для багатьох проєктів з ідентифікації, і завдяки простому інтерфейсу підключення він легко інтегрується з мікроконтролерами для реалізації різноманітних застосувань.

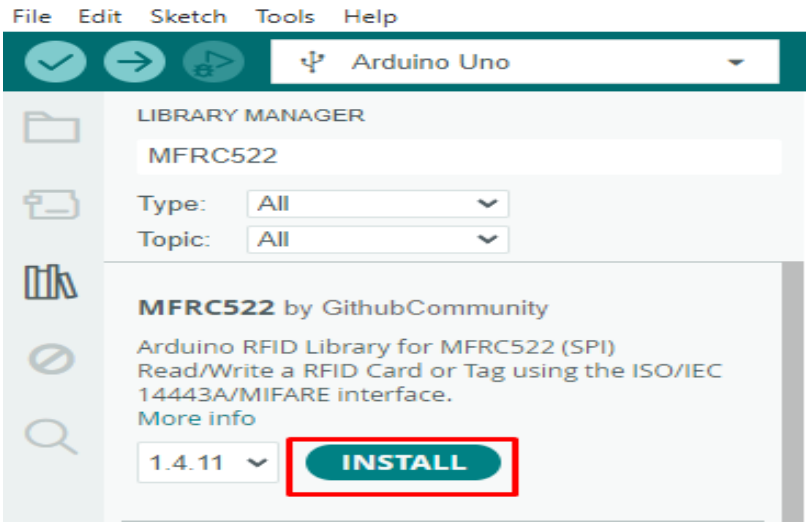
Принцип роботи

Після ініціалізації зчитувач починає періодично випромінювати радіосигнали на частоті 13.56 МГц. Якщо RFID мітка потрапляє у зону дії зчитувача, вона активується і надсилає збережену на ній інформацію через електромагнітне поле. Дані з мітки передаються у зчитувач, де вони обробляються і відправляються в мікроконтролер для подальшої обробки або виконання дій (наприклад, відкриття дверей, засвічення світлодіода тощо).

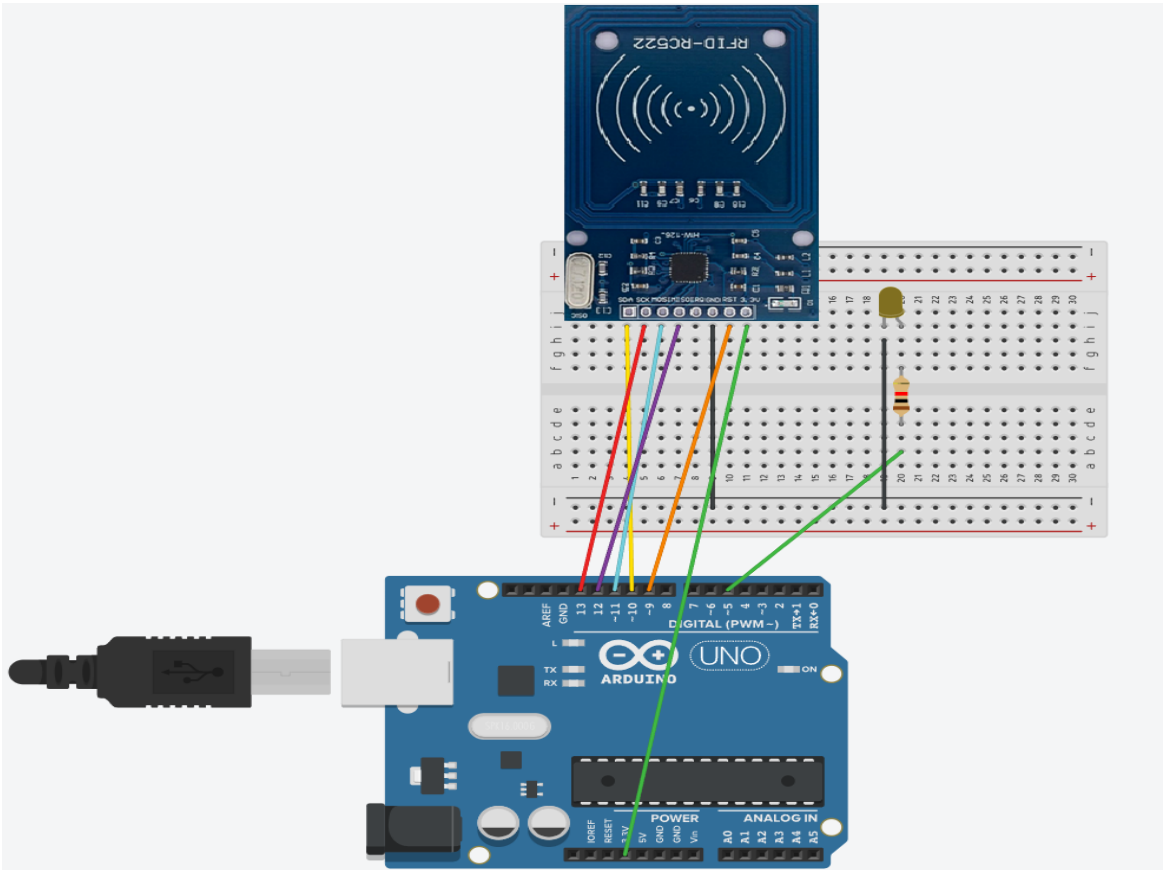
Приклад підключення такого **RFID** модуля до **Arduino Uno** подано в таблиці нижче.

Модуль MFRC522	UNO/NANO
RST	D9
SDA (SS)	D10
MOSI	D11
MISO	D12
SCK	D13

Для роботи з модулем потрібна бібліотека **MFRC522**, яку можна встановити безпосередньо з середовища **Arduino IDE.**



Підключимо **RFID**-модуль та світлодіод до плати Arduino і “змусимо” його блимати у разі спрацювання **RFID**-модуля. Для цього зберемо схему, як подано на рисунку. Опір резистора встановимо 220 Ом



```
#include <SPI.h>
#include <MFRC522.h>

#define RST_PIN         9       // Пін rfid модуля RST
#define SS_PIN          10      // Пін rfid модуля SS
#define LED_PIN 5               // Пін для світлодіода

MFRC522 rfid(SS_PIN, RST_PIN);   // Об'єкт rfid модуля

void setup() {
  Serial.begin(9600);           // Ініціалізація Serial
  SPI.begin();                  // Ініціалізація SPI
  rfid.PCD_Init();              // Ініціалізація модуля
  pinMode(LED_PIN, OUTPUT);     // Встановлюємо пін для світлодіода як вихід
}

void loop() {
  if (!rfid.PICC_IsNewCardPresent()) {    // Перевіряємо наявність нової картки
    return;
  }
  if (!rfid.PICC_ReadCardSerial()) {      // Перевіряємо, чи можна її прочитати
    return;
  }
  digitalWrite(LED_PIN, HIGH);            // Увімкнемо світлодіод
  delay(5000);
  digitalWrite(LED_PIN, LOW);             // Вимикаємо світлодіод
  rfid.PICC_HaltA();                      // Зупиняємо зв'язок із карткою
}
```

У запропонованому проєкті введено індикатор, що фіксує чи відчинені двері. Також передбачено, що індикотор горить деякий час (5 секунд).

Описаний вище проєкт можна удосконалити шляхом підключення до реле електромагніта, який буде замикати двері. Також можна організувати рідкокристалічний дисплей, що покаже привітання, коли правильний ключ до домофону і виведе повідомлення, коли неправильний.

ПІСЛЯМОВА

STEM-проєкти, які ми реалізували, демонструють широкий спектр можливостей використання мікроконтролерів та різноманітних датчиків у побутових і навчальних цілях. Проєкт "RGB-нічник з керуванням його кольором за допомогою руху рук" дає змогу користувачу інтуїтивно змінювати кольори світла без фізичного контакту, використовуючи датчик руху. Це розширює ідею взаємодії з електронними пристроями через жести, що може стати основою для створення більш складних систем управління, наприклад, розумних інтерфейсів для освітлення у будинку. Водночас проєкт підкреслює важливість навчання основ програмування і електроніки через прості та зрозумілі задачі. Такі STEM-проєкти також розвивають творче мислення й інженерні навички.

Проєкт "Вимірювання відстані" є чудовим прикладом використання ультразвукових датчиків для точного визначення відстаней. Це особливо корисно для навчання основ сенсорних технологій та їхнього застосування у реальних умовах. Система може бути розширена, включивши в неї інші сенсори для створення більш складних роботів або систем безпеки. Така концепція є важливою для розуміння, як пристрої на основі Arduino можуть забезпечувати безконтактні вимірювання у різних умовах. Це також надає основи для створення подібних проєктів у сфері інженерії чи автоматизації.

Парктроник, третій проєкт, фокусується на тому, як технології допомагають водіям керувати транспортними засобами, запобігаючи зіткненням. Використовуючи ті ж ультразвукові датчики, що й у попередньому проєкті, ця система активно вказує на перешкоди, допомагаючи користувачеві зменшити ризики під час паркування. Упровадження таких технологій показує, як невеликі системи на основі мікроконтролерів можуть застосовуватися у складних середовищах з

високими вимогами до точності. Це може слугувати стартовою точкою для розробки подібних систем у масштабніших індустріях, наприклад, у сфері автономних транспортних засобів.

Наступний проєкт, "Дистанційний вмикач світла", демонструє ефективність бездротових технологій у повсякденному житті. Керування освітленням на відстані через інфрачервоний пульт чи Bluetooth дає змогу спростити взаємодію з побутовими приладами. Це важливий крок у розвитку концепції "розумного дому", де всі пристрої можуть взаємодіяти через прості команди. Дистанційний вмикач також може бути основою для більш складних систем автоматизації, які працюють на основі часу або зовнішніх умов. Такий проєкт не лише підвищує комфорт, але й сприяє економії енергії.

Проєкт "Розумна оранжерея" виводить STEM-проєкти на новий рівень, зосереджуючись на автоматизації середовища для рослин. Використання датчиків вологості, температури та світла дає можливість підтримувати оптимальні умови для росту рослин без постійного нагляду. Це особливо корисно для аграрної індустрії або для міського фермерства. Така система може бути розширена через додавання нових датчиків і функцій, наприклад, для автоматичного поливу або контрольованого живлення рослин. Проєкт підкреслює важливість автоматизації у сфері екології та сільського господарства.

Ідеї для майбутніх STEM-проєктів можуть базуватися на подальшому розвитку автоматизації та сенсорних технологій. Наприклад, можна створити "Розумну систему вентиляції", яка контролює якість повітря та автоматично регулює вентиляцію у приміщенні. Інший проєкт – це "Система для контролю якості води", яка б за допомогою датчиків аналізувала хімічний склад води та попереджала про можливе забруднення. Ще одним цікавим напрямом може бути створення "Системи для моніторингу здоров’я", яка збирала б дані про пульс і температуру

людини і надавала рекомендації на основі цих даних. Такі ідеї розширять межі використання технологій у повсякденному житті. Крім того, можна розробити "Систему розумного поливу", яка враховувала б погодні умови та прогнози для оптимізації витрат води. Цікавим проєктом буде "Розумна охоронна система", яка за допомогою камер та датчиків руху відслідковувала б підозрілу активність і надсилала сповіщення користувачеві. Ще одна ідея – "Система автоматичного годування тварин", яка дозує їжу залежно від часу або потреб тварин. "Система відстеження викидів CO_2" може стати важливою для контролю за екологією на промислових підприємствах. А також можна створити "Розумний холодильник", який моніторить наявні продукти та рекомендує рецепти або покупки на основі терміну їхнього зберігання. Ці ідеї показують, як технології можуть інтегруватися в повсякденні процеси, спрощуючи їх та роблячи їх більш ефективними.

Таким чином, STEM-проєкти не лише допомагають освоїти основи електроніки та програмування, але й стимулюють креативне мислення і розвиток інновацій. Кожен проєкт є кроком до ширшого розуміння того, як сучасні технології можуть поліпшити життя і зробити його більш автоматизованим і ефективним. Спираючись на вже реалізовані проєкти, можна впроваджувати нові ідеї, які б розв’язували актуальні проблеми. STEM-проєкти дають можливість не лише навчатися, але й створювати реальні рішення для майбутнього.

Список використаних джерел

1. Arduino.ua [Електронний ресурс]. Режим доступу: https://arduino.ua/ .
2. Лешко Р. Інформаційно-керуючі системи та STEM-технології : методичні рекомендації до виконання лабораторних робіт. Дрогобич : ДДПУ ім. І. Франка, 2023. 40 с.
3. Arduino blog [Електронний ресурс]. Режим доступу: https://blog.arduino.cc/.
4. Arduino-DIY [Електронний ресурс]. Режим доступу: http://arduinodiy.com/.
5. Margolis M. Arduino Cookbook. O'Reilly Media, 2011. 662 p.
6. Perea F. Arduino Essentials, 2015. 206 p.
7. Tinkercad [Електронний ресурс]. Режим доступу: https://www.tinkercad.com/.
8. Microsoft C++. [Електронний ресурс]. Режим доступу: https://learn.microsoft.com/uk-ua/cpp/?view=msvc-170.
9. Грабко В.В., Розводюк М.П., Грабко Вал. В. Мікропроцесорні системи керування електроприводами. Вінниця : ВНТУ, 2012. 97 с.
10. Користувацькі посібники по роботі із платформою Arduino. [Електронний ресурс]. Режим доступу: https://www.arduino.cc/en/Tutorial/HomePage.
11. Програма технічного конструювання. Програми з позашкільної освіти науково-технічний напрям (інформаційно-технічний профіль). Київ, 2014. С. 15–32.
12. Ресурс habr.com [Електронний ресурс]. Режим доступу: https://habr.com/search/?q=Arduino#h.

Електронне навчально-методичне видання

Роман ЛЕШКО, Ольга ЛЕШКО

РЕАЛІЗАЦІЯ STEM-ПРОЄКТІВ НА БАЗІ ARDUINO

Дрогобицький державний педагогічний університет
імені Івана Франка

Редактор
Ірина Невмержицька
Технічний редактор
ІринаАртимко

Здано до набору 07.11.2024 р. Формат 60х90/16. Гарнітура Times. Ум. друк. арк. 2,75. Зам. 112.

Дрогобицький державний педагогічний університет імені Івана Франка. (Свідоцтво про внесення суб'єкта видавничої справи до державного реєстру видавців, виготівників та розповсюджувачів видавничої продукції ДК № 5140 від 01.07.2016 р.). 82100, Дрогобич, вул. Івана Франка, 24, к. 203