Національний технічний університет України «КПІ ім. Ігоря Сікорського»
Факультет Інформатики та Обчислювальної Техніки
Кафедра інформаційних систем та технологій

Лабораторна робота №4

з дисципліни «Програмування мікропроцесорних систем»

на тему

«Програмування мікропроцесорних систем. Робота з серво та кроковим двигунами.»

Виконав:
студент групи ІП-11
Дякунчак І.

Викладач:
доц. Голубєв Л. П.

Київ – 2024

Зміст

1. Постановка задачі

Мета: навчити роботі з серво та кроковим двигунами, за допомогою мікроконтролера Arduino.

Завдання до роботи:

В кожній з робіт потрібно розробити схеми та заставити її працювати за правилами, що викладені в задачах. До кожної із задач у звіті повинні бути намальовані відповідні схеми.

За допомогою сервісу **tinkercad.com** створити наступні проекти:

В середовищі tinkercad створити на малий макетній платі принципіальну схему пристрою, а в розділі «Текст» написати текст програми.

1. **Робота з серводвигуном**. Створити проект швидкого повороту серводвигуна на кут

$$\alpha = \begin{cases} n * 5 + 20, n \le 10 \\ n * 4 + 20, \wedge 10 < x \le 20 \\ n * 3 + 20, \wedge x > 20 \end{cases}$$

і повільного його повернення в початкове положення (з затримкою n * 10).

Значення кута ввести:

- в моніторі порту.
- за допомогою потенціометра.

2. **Робота з кроковим двигуном** (**виконується за допомогою інтернет-сервісу WokWi)** Написати програму повороту крокового двигуна на кут, заданий в п.1 за допомогою бібліотеки Stepper.

На першому натисканні на кнопку він повертається за годинниковою стрілкою, при другому - проти.

Додати можливість змінювати швидкість обертання (за допомогою потенціометра).

3. **Робота з двигуном постійного струму**. Режими обертання:

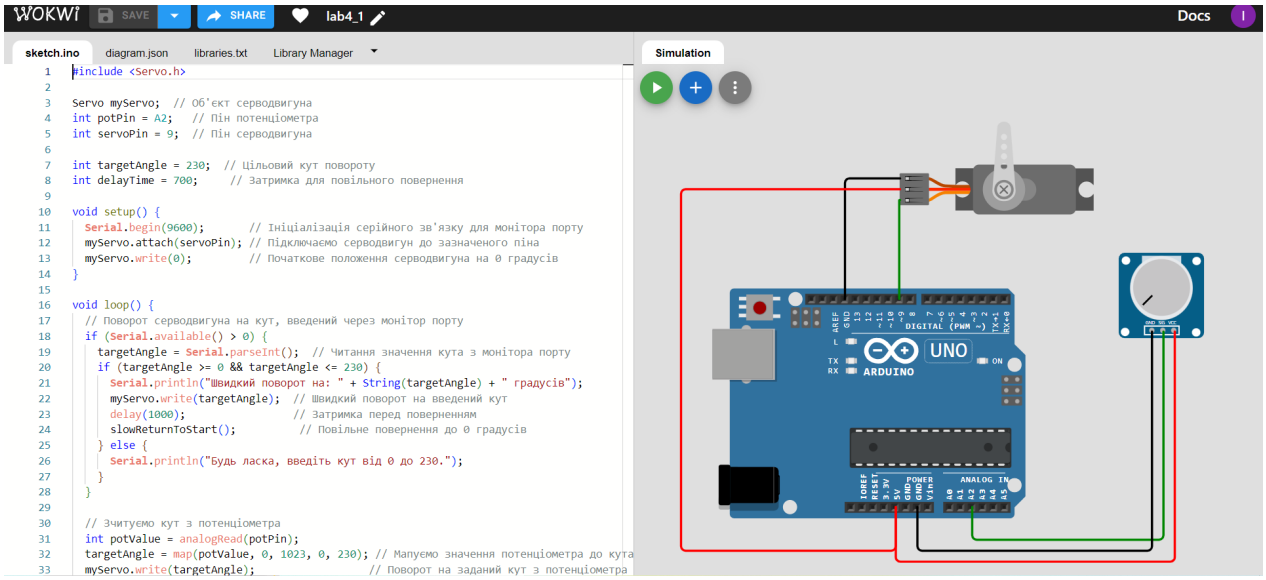
- за годинниковою стрілкою;
- проти годинникової стрілки;
- збільшення швидкості обертання;
- зменшення швидкості обертання;

Вибір режимів роботи здійснити за допомогою пульта дистанційного керування.

Змінювати швидкість обертання за допомогою ШІМ.

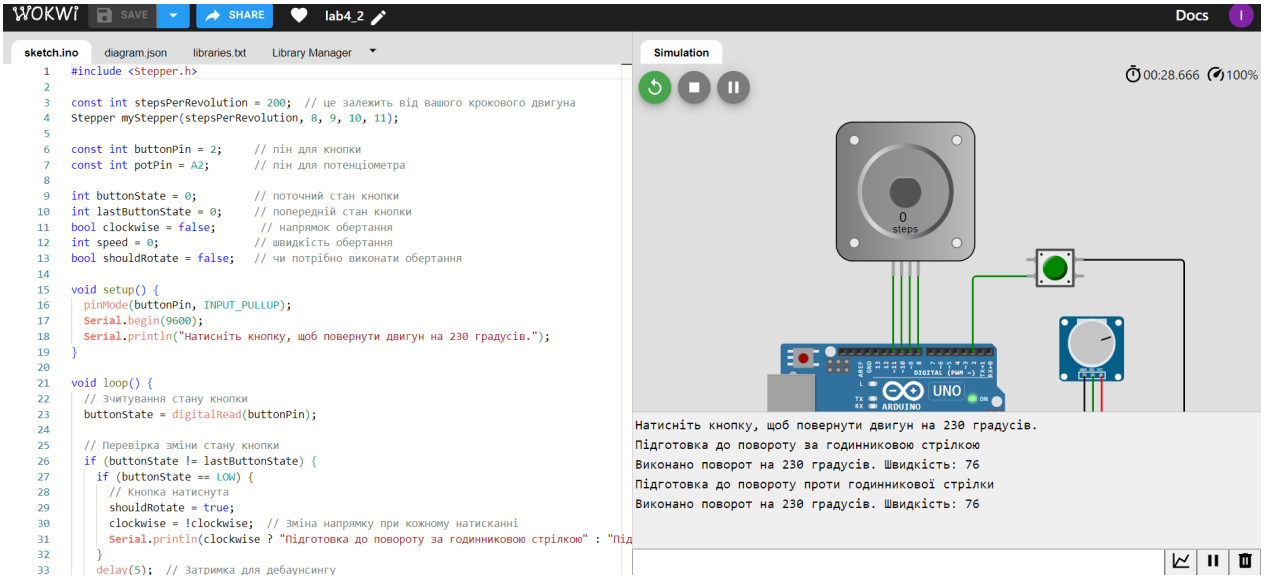
2. Виконання

Спочатку потрібно було створити перший проект швидкого повороту серводвигуна на кут, заданий в моніторі порту або за допомогою потенціометра, код та знімки екрану наведені нижче:



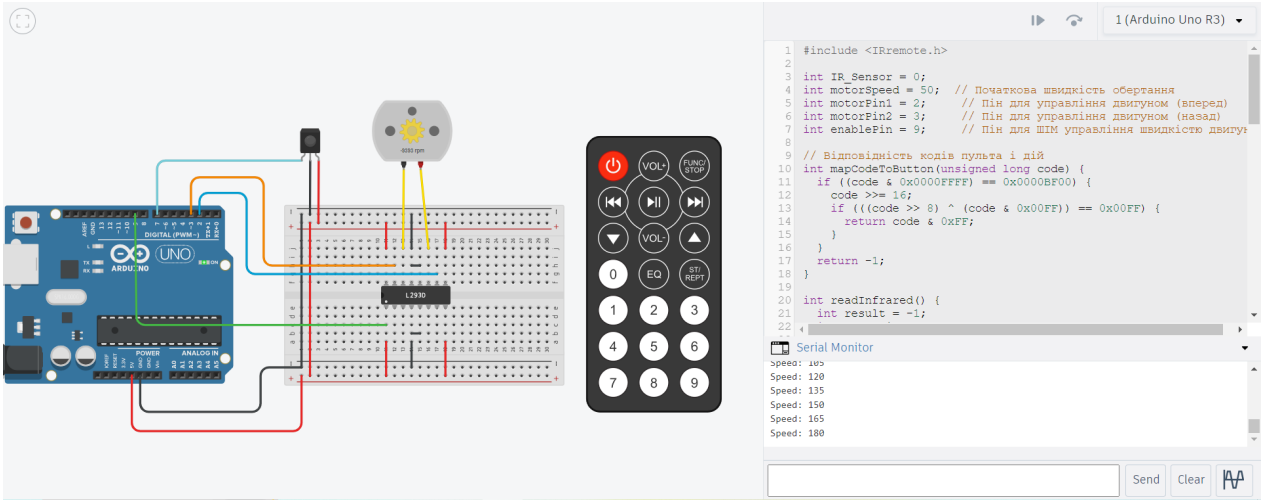
Як бачимо, усе працює, отже можна переходити до створення другого проекту з кроковим двигуном.

Результати роботи наведено на знімку екрану нижче, повний код програми можна знайти у додатках, також там знаходиться і посилання на проект.




```arduino
#include <Stepper.h>

const int stepsPerRevolution = 200;  // це залежить від вашого крокового двигуна
Stepper myStepper(stepsPerRevolution, 8, 9, 10, 11);

const int buttonPin = 2;     // пін для кнопки
const int potPin = A2;       // пін для потенціометра

int buttonState = 0;         // поточний стан кнопки
int lastButtonState = 0;     // попередній стан кнопки
bool clockwise = false;       // напрямок обертання
int speed = 0;               // швидкість обертання
bool shouldRotate = false;   // чи потрібно виконати обертання

void setup() {
  pinMode(buttonPin, INPUT_PULLUP);
  Serial.begin(9600);
  Serial.println("Натисніть кнопку, щоб повернути двигун на 230 градусів.");
}

void loop() {
  // Зчитування стану кнопки
  buttonState = digitalRead(buttonPin);

  // Перевірка зміни стану кнопки
  if (buttonState != lastButtonState) {
    if (buttonState == LOW) {
      // Кнопка натиснута
      shouldRotate = true;
      clockwise = !clockwise;  // Зміна напрямку при кожному натисканні
      Serial.println(clockwise ? "Підготовка до повороту за годинниковою стрілкою" : "Під
    }
    delay(5);  // Затримка для дебаунсингу
```

Як бачимо на екрані, усе працює, отже можна переходити до створення третього проекту з двигуном постійного струму.




```arduino
#include <IRremote.h>

int IR_Sensor = 0;
int motorSpeed = 50;  // Початкова швидкість обертання
int motorPin1 = 2;      // Пін для управління двигуном (вперед)
int motorPin2 = 3;      // Пін для управління двигуном (назад)
int enablePin = 9;      // Пін для ШІМ управління швидкістю двигуна

// Відповідність кодів пульта і дій
int mapCodeToButton(unsigned long code) {
  if ((code & 0x0000FFFF) == 0x0000BF00) {
    code >>= 16;
    if ((code >> 8) ^ (code & 0x00FF)) == 0x00FF) {
      return code & 0xFF;
    }
  }
  return -1;
}

int readInfrared() {
  int result = -1;
```

3. Контрольні питання

1. Що таке сервопривід?

Сервопривід – це електромеханічний пристрій, призначений для точного керування кутом обертання валу.

2. Яка бібліотека дозволяє керувати сервоприводом?

Бібліотека, що дозволяє керувати сервоприводом в Arduino, – це Servo.

3. Як підключити бібліотеку, що є серед бібліотек вашого Arduino, до поточного скетчу?

Щоб підключити бібліотеку в Arduino, потрібно використати команду #include <Servo.h> на початку скетчу.

4. Як заставити серводвигун повернутися на потрібний кут?

Щоб змусити серводвигун повернутися на потрібний кут, потрібно використати функцію servo.write(кут);, де кут – це значення від 0 до 180.

5. Що таке крокoвий двигун?

Крокoвий двигун – це електродвигун, що обертається на визначені кути (кроки), що дозволяє здійснювати точне позиціонування.

6. Принцип роботи крокового двигуна.

Принцип роботи крокового двигуна полягає в активації котушок в певній послідовності, що дозволяє йому обертатися на заданий кут (крок).

7. Що означають терміни 8 – керуюча послідовність, 64 передаточне число редуктора, 512 кроків для здійснення повного оберту валу при роботі з крокoвим двигуном?

Керуюча послідовність – це послідовність активування котушок; передаточне число редуктора – співвідношення обертів двигуна до обертів виходу; кроки для повного оберту – кількість кроків, необхідних для 360° обертання.

8. Що таке ULN2003 і як працювати з нею?

ULN2003 – це інтерфейсний чіп для керування індуктивними навантаженнями (такі як крокові двигуни), що дозволяє підключати їх до мікроконтролерів, використовуючи низькі керуючі напруги.

9. На який мінімальний кут можна повернути крокoвий двигун?

Мінімальний кут, на який можна повернути крокoвий двигун, залежить від його конструкції, але зазвичай складає 1.8° за крок, отже, 0.9° з використанням половинного кроку.

10. Як заставити крокoвий двигун повертатися в ту чи іншу сторону?

Щоб змусити крокoвий двигун повертатися в ту чи іншу сторону, потрібно активувати його котушки в потрібній послідовності (за годинниковою стрілкою або проти).

4. Висновок

У даній лабораторній роботі я навчилась роботі з серво та крововим двигунами, за допомогою мікроконтролера Arduino. В процесі виконнання я створила 3 проекти з використанням серводвигуна, крокового двигуна та двигуна постійного струму. Усі результати наведені на знімках екрану вище, код програми та посилання на сам проект

5. Додатки

Посилання на перший проект: https://wokwi.com/projects/409576948987139073

Код проекту:

```
#include <Servo.h>

Servo myServo;  // Об'єкт серводвигуна
int potPin = A2;   // Пін потенціометра
int servoPin = 9;  // Пін серводвигуна

int targetAngle = 230;  // Цільовий кут повороту
int delayTime = 700;     // Затримка для повільного повернення

void setup() {
  Serial.begin(9600);       // Ініціалізація серійного зв'язку для монітора порту
  myServo.attach(servoPin); // Підключаємо серводвигун до зазначеного піна
  myServo.write(0);         // Початкове положення серводвигуна на 0 градусів
}

void loop() {
  // Поворот серводвигуна на кут, введений через монітор порту
  if (Serial.available() > 0) {
    targetAngle = Serial.parseInt();  // Читання значення кута з монітора порту
    if (targetAngle >= 0 && targetAngle <= 230) {
      Serial.println("Швидкий поворот на: " + String(targetAngle) + " градусів");
      myServo.write(targetAngle);  // Швидкий поворот на введений кут
      delay(1000);                 // Затримка перед поверненням
      slowReturnToStart();         // Повільне повернення до 0 градусів
```

```
    } else {
      Serial.println("Будь ласка, введіть кут від 0 до 230.");
    }
  }

  // Зчитуємо кут з потенціометра
  int potValue = analogRead(potPin);
  targetAngle = map(potValue, 0, 1023, 0, 230); // Мапуємо значення
потенціометра до кута 0-230
  myServo.write(targetAngle);                  // Поворот на заданий кут з
потенціометра
  delay(1000);                              // Затримка перед повільним поверненням
  slowReturnToStart();                        // Повільне повернення
}

void slowReturnToStart() {
  // Повільне повернення серводвигуна до 0 градусів
  for (int pos = targetAngle; pos >= 0; pos--) {
    myServo.write(pos);
    delay(delayTime);  // Затримка між змінами положення для повільного руху
  }
  Serial.println("Повернувся в початкове положення.");
}
```

Посилання на другий проект :
Код проекту : https://wokwi.com/projects/409577111079176193

```
#include <Stepper.h>

const int stepsPerRevolution = 200;  // це залежить від вашого крокового
двигуна
Stepper myStepper(stepsPerRevolution, 8, 9, 10, 11);

const int buttonPin = 2;     // пін для кнопки
const int potPin = A2;       // пін для потенціометра
```

```
int buttonState = 0;         // поточний стан кнопки
int lastButtonState = 0;     // попередній стан кнопки
bool clockwise = false;       // напрямок обертання
int speed = 0;               // швидкість обертання
bool shouldRotate = false;   // чи потрібно виконати обертання

void setup() {
  pinMode(buttonPin, INPUT_PULLUP);
  Serial.begin(9600);
  Serial.println("Натисніть кнопку, щоб повернути двигун на 230 градусів.");
}

void loop() {
  // Зчитування стану кнопки
  buttonState = digitalRead(buttonPin);

  // Перевірка зміни стану кнопки
  if (buttonState != lastButtonState) {
    if (buttonState == LOW) {
      // Кнопка натиснута
      shouldRotate = true;
      clockwise = !clockwise;  // Зміна напрямку при кожному натисканні
      Serial.println(clockwise ? "Підготовка до повороту за годинниковою
стрілкою" : "Підготовка до повороту проти годинникової стрілки");
    }
    delay(5);  // Затримка для дебаунсингу
  }
  lastButtonState = buttonState;

  // Зчитування значення з потенціометра та встановлення швидкості
  speed = map(analogRead(potPin), 0, 1023, 1, 100);
  myStepper.setSpeed(speed);

  if (shouldRotate) {
```

```
    // Обертання двигуна
    int stepsToMove = (230.0 / 360.0) * stepsPerRevolution;
    if (clockwise) {
      myStepper.step(stepsToMove);
    } else {
      myStepper.step(-stepsToMove);
    }

    Serial.print("Виконано поворот на 230 градусів. Швидкість: ");
    Serial.println(speed);

    shouldRotate = false;  // Скидаємо прапорець, щоб запобігти повторному
обертанню
  }
}
```

Посилання на третій проект :
Код проекту : https://www.tinkercad.com/things/6JdxWWJ9Dos-lab43

```
#include <IRremote.h>

int IR_Sensor = 0;
int motorSpeed = 50;  // Початкова швидкість обертання
int motorPin1 = 2;     // Пін для управління двигуном (вперед)
int motorPin2 = 3;     // Пін для управління двигуном (назад)
int enablePin = 9;     // Пін для ШІМ управління швидкістю двигуна

// Відповідність кодів пульта і дій
int mapCodeToButton(unsigned long code) {
  if ((code & 0x0000FFFF) == 0x0000BF00) {
    code >>= 16;
    if (((code >> 8) ^ (code & 0x00FF)) == 0x00FF) {
      return code & 0xFF;
    }
```

```
  }
  return -1;
}

int readInfrared() {
  int result = -1;
  if (IrReceiver.decode()) {
    unsigned long code = IrReceiver.decodedIRData.decodedRawData;
    result = mapCodeToButton(code);
    IrReceiver.resume();
  }
  return result;
}

void setup() {
  IrReceiver.begin(7);  // ІЧ датчик підключено до піна 7
  Serial.begin(9600);
  pinMode(motorPin1, OUTPUT);
  pinMode(motorPin2, OUTPUT);
  pinMode(enablePin, OUTPUT);  // ШІМ пін для управління швидкістю
}

void loop() {
  IR_Sensor = readInfrared();
  Serial.println(IR_Sensor);

  // Обертання за годинниковою стрілкою
  if (IR_Sensor == 16) {
    digitalWrite(motorPin1, HIGH);
    digitalWrite(motorPin2, LOW);
    analogWrite(enablePin, motorSpeed);  // Встановлення швидкості
  }

  // Обертання проти годинникової стрілки
  if (IR_Sensor == 17) {
```

```
  digitalWrite(motorPin1, LOW);
  digitalWrite(motorPin2, HIGH);
  analogWrite(enablePin, motorSpeed);
 }

 //Збільшення швидкості
 if (IR_Sensor == 18) {
  for(int i = 0; i<= 255; i+=15){
        analogWrite(enablePin, i);
        Serial.print("Speed: ");
        Serial.println(i);
        delay(1000);
  }
 }

 //Зменшення швидкості
 if (IR_Sensor == 20) {
  for(int i = 255; i>= 0; i-=15){
        analogWrite(enablePin, i);
        Serial.print("Speed: ");
        Serial.println(i);
        delay(1000);
  }
 }

 delay(10);  // Затримка для оптимізації симуляції
}
```