**Министерство науки и высшего образования Российской Федерации**

**Федеральное государственное автономное образовательное**

**учреждение высшего образования**

**«КАЗАНСКИЙ (ПРИВОЛЖСКИЙ) ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»**

#### ИНСТИТУТ ФИЗИКИ

#### КАФЕДРА РАДИОФИЗИКИ

Направление подготовки: 10.03.01 – Информационная безопасность

Профиль: Информационная безопасность телекоммуникационных систем

Лабораторная работа

«Программное обеспечение для микроконтроллера»

Обучающийся 1 курса 06-451 группы

очной формы обучения Ахметсафин Р.Н.

Научный руководитель Сарычев Д.В.

#### **Казань – 2024**

## Цель работы

Разработать программу на языке C для микроконтроллера Arduino, реализующую управление сервоприводом на основе данных ультразвукового дальномера с периодическим опросом, обработкой данных и плавным изменением положения сервопривода.

**Схема на Arduino**

Изображение выглядит как электроника, Электронная техника, снимок экрана, диаграмма

Контент, сгенерированный ИИ, может содержать ошибки.

## Код программы

#include <Servo.h>

// Константы для настройки системы

#define TRIG\_PIN 9 // Пин для управления дальномером (Trig)

#define ECHO\_PIN 10 // Пин для чтения данных дальномера (Echo)

#define SERVO\_PIN 6 // Пин управления сервоприводом

#define LED\_PIN 13 // Пин светодиода для индикации

#define MEASURE\_INTERVAL 100 // Интервал измерений (мс)

#define MIN\_DISTANCE 10 // Минимальное расстояние (см)

#define MAX\_DISTANCE 200 // Максимальное расстояние (см)

#define MIN\_ANGLE 0 // Минимальный угол сервопривода

#define MAX\_ANGLE 180 // Максимальный угол сервопривода

// Глобальные переменные

Servo myServo; // Объект сервопривода

unsigned long lastMeasureTime = 0; // Время последнего измерения

int currentAngle = 90; // Текущий угол сервопривода

int targetAngle = 90; // Целевой угол сервопривода

bool ledState = false; // Состояние светодиода

void setup() {

// Инициализация пинов

pinMode(TRIG\_PIN, OUTPUT);

pinMode(ECHO\_PIN, INPUT);

pinMode(LED\_PIN, OUTPUT);

// Инициализация сервопривода

myServo.attach(SERVO\_PIN);

myServo.write(currentAngle);

// Инициализация последовательного порта для отладки

Serial.begin(9600);

// Мигание светодиодом при старте

for (int i = 0; i < 3; i++) {

digitalWrite(LED\_PIN, HIGH);

delay(200);

digitalWrite(LED\_PIN, LOW);

delay(200);

}

}

void loop() {

unsigned long currentTime = millis();

// Периодический опрос дальномера

if (currentTime - lastMeasureTime >= MEASURE\_INTERVAL) {

lastMeasureTime = currentTime;

// Измерение расстояния

float distance = measureDistance();

// Обработка данных и расчет угла

if (distance > 0) {

targetAngle = mapDistanceToAngle(distance);

// Индикация работы

ledState = !ledState;

digitalWrite(LED\_PIN, ledState);

// Вывод отладочной информации

Serial.print("Distance: ");

Serial.print(distance);

Serial.print(" cm, Target angle: ");

Serial.println(targetAngle);

}

}

// Плавное изменение положения сервопривода

smoothServoMovement();

}

// Функция измерения расстояния

float measureDistance() {

// Генерация импульса для измерения

digitalWrite(TRIG\_PIN, LOW);

delayMicroseconds(2);

digitalWrite(TRIG\_PIN, HIGH);

delayMicroseconds(10);

digitalWrite(TRIG\_PIN, LOW);

// Измерение длительности импульса

long duration = pulseIn(ECHO\_PIN, HIGH);

// Расчет расстояния (см)

float distance = duration \* 0.034 / 2;

// Проверка на допустимый диапазон

if (distance < MIN\_DISTANCE || distance > MAX\_DISTANCE) {

return -1; // Некорректное расстояние

}

return distance;

}

// Функция преобразования расстояния в угол сервопривода

int mapDistanceToAngle(float distance) {

// Ограничение расстояния в допустимом диапазоне

distance = constrain(distance, MIN\_DISTANCE, MAX\_DISTANCE);

// Линейное преобразование расстояния в угол

int angle = map(distance, MIN\_DISTANCE, MAX\_DISTANCE, MAX\_ANGLE, MIN\_ANGLE);

// Дополнительное сглаживание (по желанию)

angle = constrain(angle, MIN\_ANGLE, MAX\_ANGLE);

return angle;

}

// Функция плавного перемещения сервопривода

void smoothServoMovement() {

if (currentAngle != targetAngle) {

// Медленное изменение угла (1 градус за цикл)

if (currentAngle < targetAngle) {

currentAngle++;

} else {

currentAngle--;

}

// Установка нового положения

myServo.write(currentAngle);

// Небольшая задержка для плавности

delay(15);

}

}

## Результат работы программы

Программа успешно реализует:

1. Периодический опрос ультразвукового дальномера
2. Преобразование расстояния до объекта в угол поворота сервопривода
3. Плавное изменение положения сервопривода при изменении расстояния
4. Светодиодную индикацию работы системы

## Заключение

Разработана программа для микроконтроллера Arduino, которая позволяет автоматически управлять положением сервопривода на основе данных, получаемых с ультразвукового дальномера. Программа включает функции периодического опроса датчика, обработки данных, преобразования расстояния в угол поворота и плавного изменения положения сервопривода. Система также обеспечивает визуальную индикацию работы через светодиод.