МОСКОВСКИЙ АВТОМОБИЛЬНО‑ДОРОЖНЫЙ  
ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ  
УНИВЕРСИТЕТ (МАДИ)

Кафедра «Автоматизированные системы управления»

ОТЧЕТ  
О ВЫПОЛНЕНИИ ЗАДАНИЙ ARDUINO  
по дисциплине «**Микропроцессорные информационно-управляющие системы на транспорте**»

Выполнил: Михайлин И. Ю.

Группа: 2бИТС3

Преподаватель: Максимычев О. И.

Москва  
2024

1. **Подключение светодиода и кнопки к плате Arduino UNO**

**1.1 Электрическая схема**

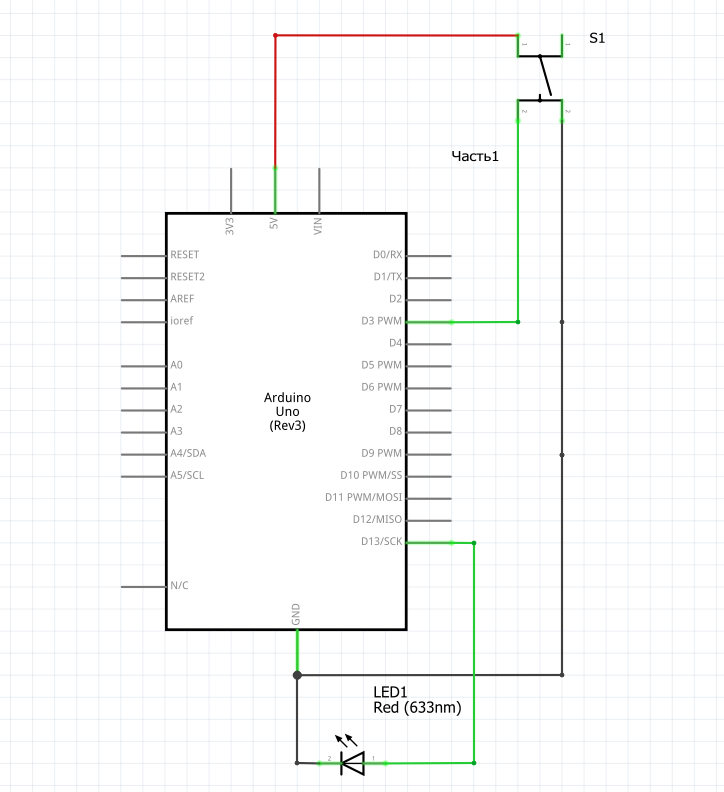


Рисунок 1.1 – Электрическая (принципиальная) схема подключения светодиода и кнопки к плате Arduino UNO

**1.2 Монтажная схема**

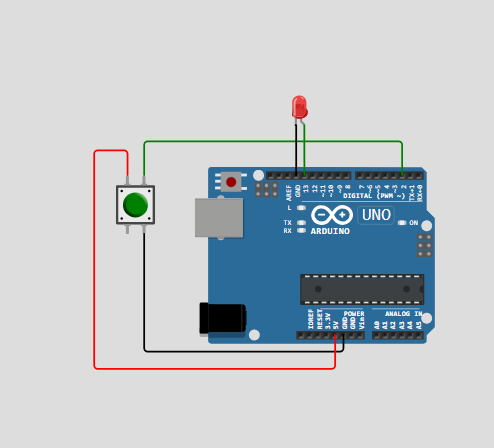


Рисунок 1.2 – Монтажная схема подключения светодиода и кнопки к плате Arduino UNO

**1.3 Код подключения**

// Контакт 13 для подключения светодиода

int LED=13;

// Контакт 2 для подключения кнопки

int BUTTON=2;

// переменная статуса кнопки buttonState

boolean buttonState;

void setup() {

// определяем вывод LED (светодиод) как выход

pinMode(LED, OUTPUT);

// определяем вывод BUTTON (кнопка) как вход

pinMode(BUTTON, INPUT\_PULLUP);

}

void loop() {

// считываем состояние BUTTON входа и записываем в buttonState

buttonState = digitalRead(BUTTON);

// инверсия переменной buttonState

// для схемы с подтягивающим резистором

//buttonState =! buttonState;

// записываем состояние из buttonState на выход LED (светодиод)

digitalWrite(LED, buttonState);

}

**1.2 Подключение светодиодов и кнопок к плате Arduino UNO**

**1.1 Электрическая схема**

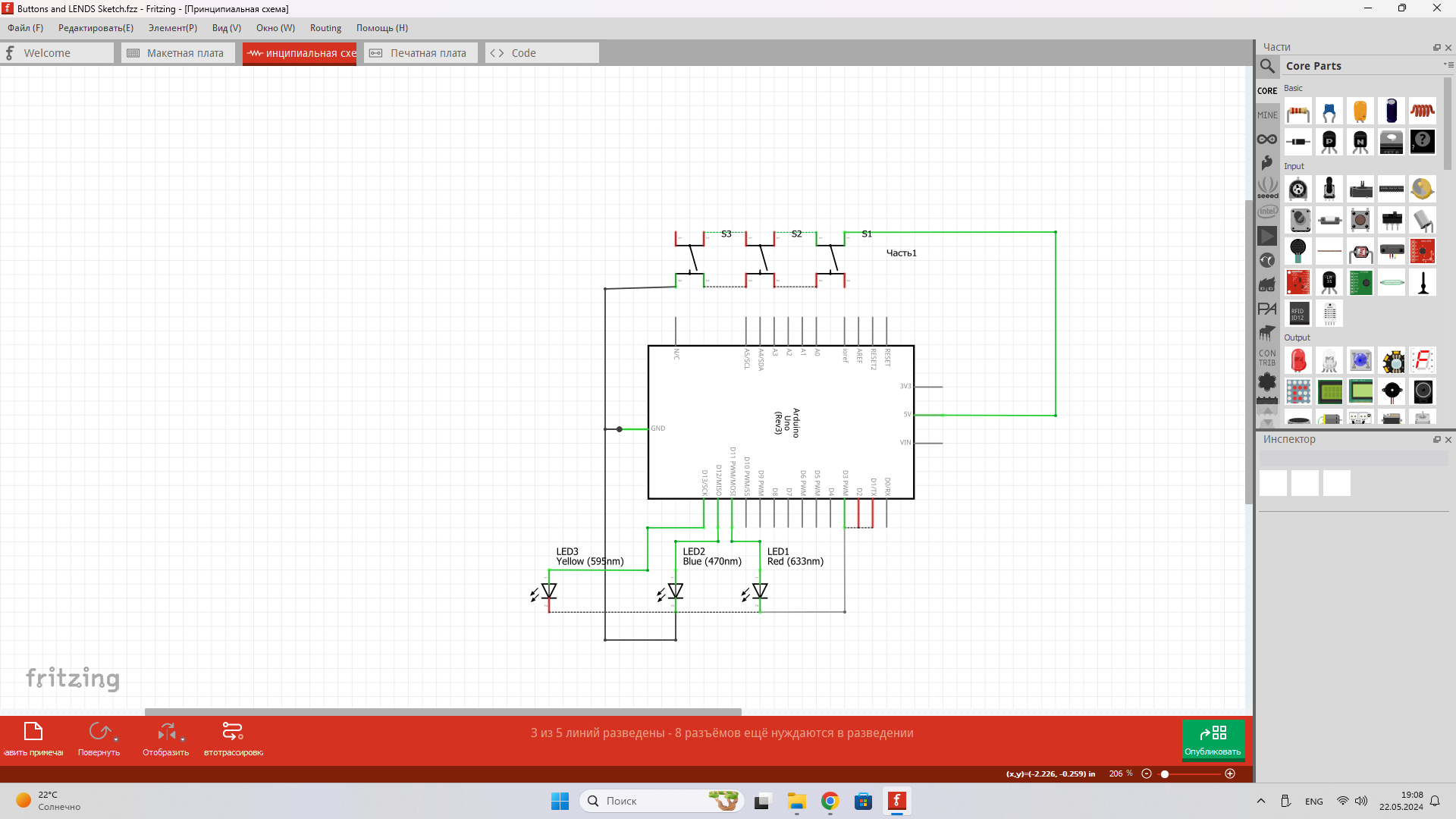


Рисунок 1.2.1 – Электрическая (принципиальная) схема подключения светодиодов и кнопок к плате Arduino UNO

**1.2 Монтажная схема**

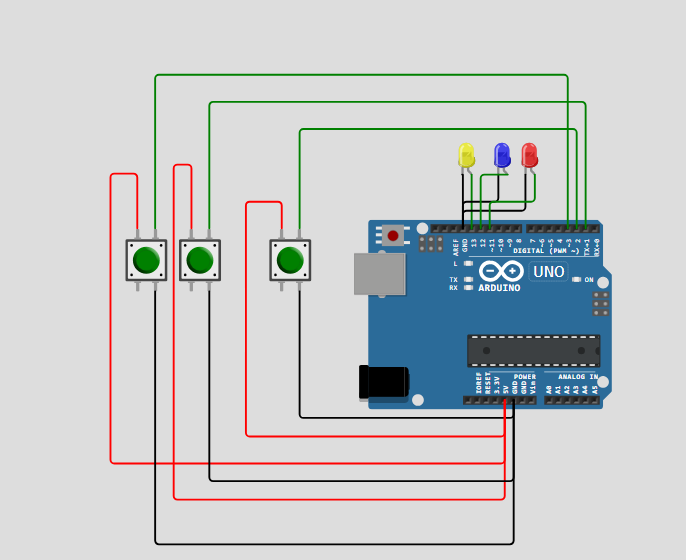


Рисунок 1.2.2 – Монтажная схема подключения светодиодов и кнопок к плате Arduino UNO

**1.3 Код подключения**

// Контакт 13 для подключения светодиода 0

int LED=13;

// Контакт 13 для подключения светодиода 1

int LED1=12;

// Контакт 13 для подключения светодиода 2

int LED2 =11;

// Контакт 2 для подключения кнопки 0

int BUTTON=2;

// Контакт 2 для подключения кнопки 1

int BUTTON1=3;

// Контакт 2 для подключения кнопки 2

int BUTTON2=4;

// переменная статуса кнопки buttonState

boolean buttonState;

boolean buttonState1;

boolean buttonState2;

void setup() {

// определяем вывод LED (светодиод) как выход

pinMode(LED, OUTPUT);

pinMode(LED1, OUTPUT);

pinMode(LED2, OUTPUT);

// определяем вывод BUTTON (кнопка) как вход

pinMode(BUTTON, INPUT\_PULLUP);

pinMode(BUTTON1, INPUT\_PULLUP);

pinMode(BUTTON2, INPUT\_PULLUP);

}

void loop() {

// считываем состояние BUTTON входа и записываем в buttonState

buttonState = digitalRead(BUTTON);

buttonState1 = digitalRead(BUTTON1);

buttonState2 = digitalRead(BUTTON2);

// инверсия переменной buttonState

// для схемы с подтягивающим резистором

//buttonState = ! buttonState;

// записываем состояние из buttonState на выход LED (светодиод)

digitalWrite(LED, buttonState);

digitalWrite(LED1, buttonState1);

digitalWrite(LED2, buttonState2);

}

1. **Подключение потенциометра и дисплея к плате Arduino UNO**

**1.1 Электрическая схема**

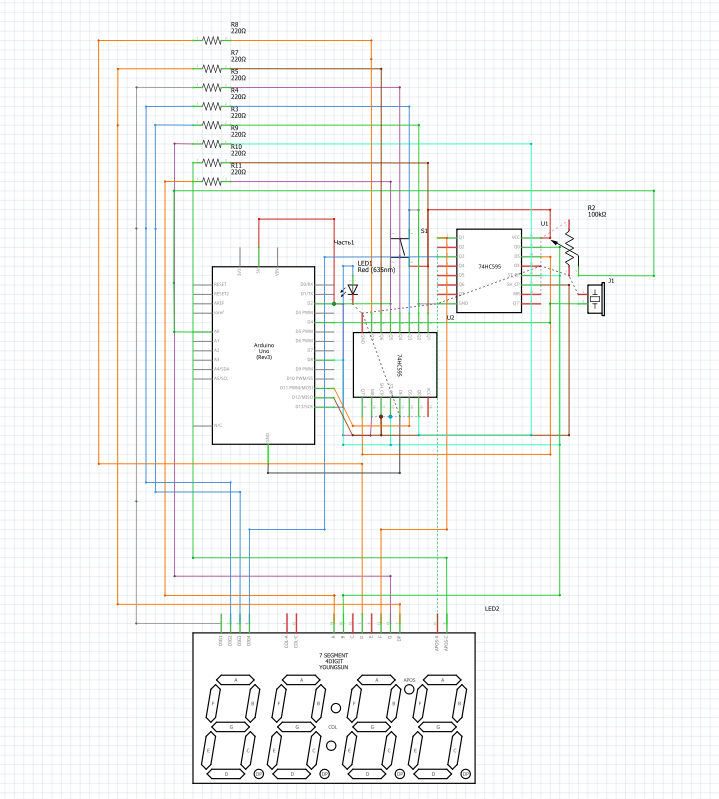


Рисунок 2.1 – Электрическая (принципиальная) схема подключения светодиодов и кнопок к плате Arduino UNO

**1.2 Монтажная схема**

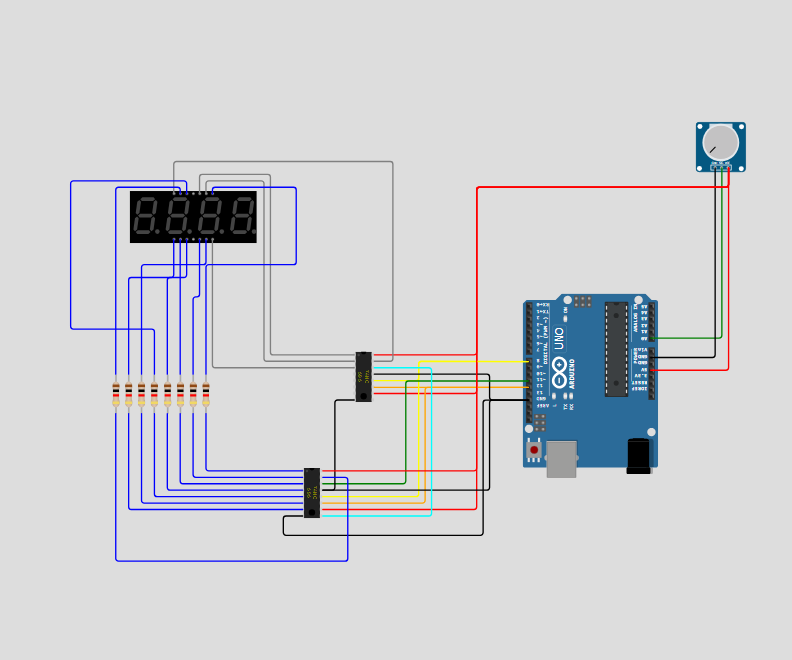


Рисунок 2.2 – Монтажная схема подключения светодиодов и кнопок к плате Arduino UNO

**1.3 Код подключения**

int latchPin = 8; //Пин "защелки" первого регистра подключен к ST\_CP входу первого регистра отвечающего за сегменты

int clockPin = 12; //Пин подключен к SH\_CP входу 74HC595

int dataPin = 11; //Пин подключен к DS входу 74HC595

int TimeLight = 5; //время для разогрева сегментов

int sensorPin=A0;

byte SegDisplay; //переменная для вывода символов на индикаторе

byte RazrDisplay; //переменная для включения разрядов

// Настройка комбинации для отображения каждого номера на дисплее.

// Комбинация выбрана на основе таблицы отображаемого знака данным порта

// Соответственно { 0 , 1 , 2 , 3 , 4 , 5 , 6 , 7 , 8 , 9 , dp, - }

byte g\_digits[12]={192,249,164,176,153,146,130,248,128,144,127,255}; //массив цифр, генерирующий из сегментов цифры

byte g\_registerArray[4]={1,2,4,8}; //массив цифр, указывающий разряды

void setup() {

// обозначаем все пины как выходы

pinMode(latchPin, OUTPUT);

pinMode(clockPin, OUTPUT);

pinMode(dataPin, OUTPUT);

}

void loop() {

int disp = 0; //создаем переменную для вывода на экран

int analogueValue = analogRead(sensorPin); // читаем аналоговый пин A0

analogueValue = map(analogueValue, 0, 1023, 0, 8888); //преобразуем диапазон с А0 (0-1023) в нужный нам (0-8888)

disp = analogueValue; // записываем получившуюся после преобразования цифру в переменну для вывода на экран

// Разбиваем цифру по разрядам индикатора

if (disp < 10) // если наша цифра меньше 10, то

{

Indicate(0, 11); // пишем в первый разряд пусто

Indicate(1, 11); // пишем во второй разряд пусто

Indicate(2, 11); // пишем в третий разряд пусто

Indicate(3, disp); // пишем в четвертый разряд нашу цифру

}

else if (disp < 100) // если наша цифра меньше 100, то

{

Indicate(0, 11); // пишем в первый разряд пусто

Indicate(1, 11); // пишем во второй разряд пусто

Indicate(2, disp / 10); // пишем в третий разряд - цифру делёную на 10

Indicate(3, disp % 10); // пишем в четвертый разряд цифру оставшуюся от деления на 10

/\*

Допустим наша цифра 25.

Если мы ее поделим на 10, то у нас получится 2,5. Цифры после запятой, в данном случае, у нас не остаются. Таким образом мы имеем в третем разряде цифру 2.

В чевертый разряд мы, как раз и записываем цифру-остаток полученную в результате деления.

В нашем случае это и есть та самая 5.

Аналогичным образом разбивается наша цифра и далее.

\*/

}

else if (disp < 1000)

{

Indicate(0, 11);

Indicate(1, disp / 100);

Indicate(2, (disp % 100) / 10);

Indicate(3, disp % 10);

}

else

{

Indicate(0, disp / 1000);

Indicate(1, (disp % 1000) / 100);

Indicate(2, (disp % 100) / 10);

Indicate(3, disp % 10);

}

}

void Indicate(int r,int x)

{

SegDisplay=g\_digits[x]; // получаем цифру и выводим символ, из массива цифр, соответствующий этой цифре.

RazrDisplay=g\_registerArray[r]; // получаем цифру и выводим номер регистра, из массива цифр, соответствующий этой цифре.

digitalWrite(latchPin, LOW); // устанавливаем синхронизацию "защелки" на LOW

shiftOut(dataPin, clockPin, MSBFIRST, RazrDisplay); // Записываем информацию для второго регистра (Номер разряда)

shiftOut(dataPin, clockPin, MSBFIRST, SegDisplay); // Записываем информацию для первого регистра (Номер символа)

digitalWrite(latchPin, HIGH); //"защелкиваем" регистр, тем самым устанавливая значения на выходах

delay(TimeLight); // пауза, чтобы сегменты "разгорелись"

}

**2.2 Подключение потенциометра, динамика, дисплея и светодиода к плате Arduino UNO**

**1.1 Электрическая схема**

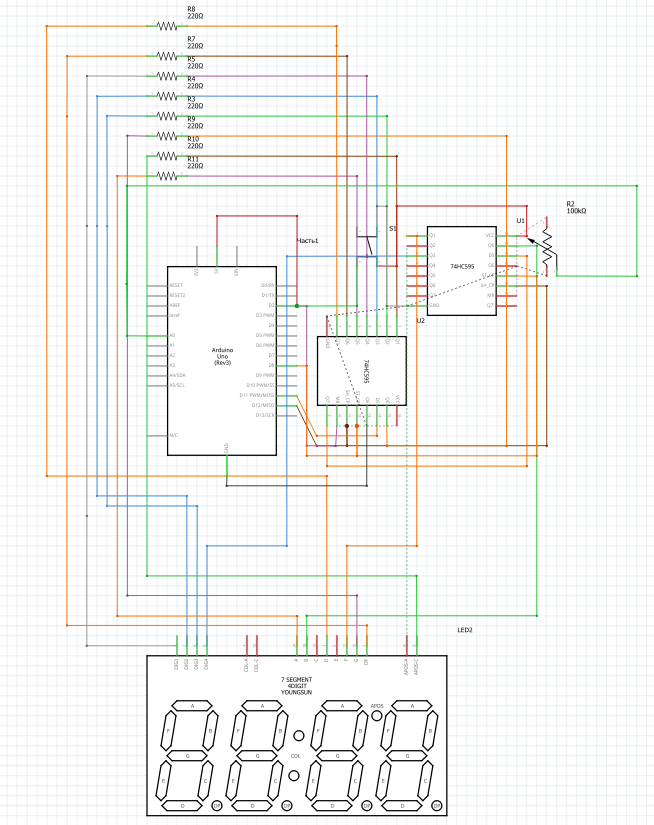


Рисунок 2.2.1 – Электрическая (принципиальная) схема подключения потенциометра, динамика, дисплея и светодиода к плате Arduino UNO

**1.2 Монтажная схема**

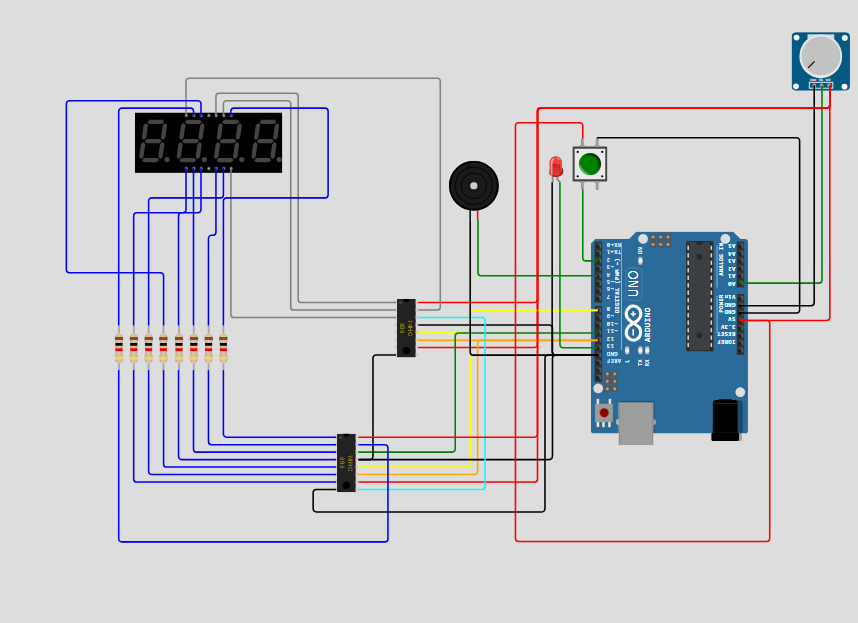


Рисунок 2.2.2 – Монтажная схема подключения потенциометра, динамика, дисплея и светодиода к плате Arduino UNO

**1.3 Код подключения**

int latchPin = 8; //Пин "защелки" первого регистра подключен к ST\_CP входу первого регистра отвечающего за сегменты

int clockPin = 12; //Пин подключен к SH\_CP входу 74HC595

int dataPin = 11; //Пин подключен к DS входу 74HC595

int TimeLight = 5; //время для разогрева сегментов

int sensorPin=A0;

byte SegDisplay; // переменная для вывода символов на индикаторе

byte RazrDisplay; // переменная для включения разрядов

// Настройка комбинации для отображения каждого номера на дисплее.

// Комбинация выбрана на основе таблицы отображаемого знака данным порта

// Соответственно {0 , 1 , 2 , 3 , 4 , 5 , 6 , 7 , 8 , 9 , dp, - }

byte g\_digits[12]={192,249,164,176,153,146,130,248,128,144,127,255}; //массив цифр, генерирующий из сегментов цифры

byte g\_registerArray[4]={1,2,4,8}; //массив цифр, указывающий разряды

// Контакт 13 для подключения светодиода

int LED=13;

// Контакт 2 для подключения кнопки

int BUTTON=2;

// переменная статуса кнопки buttonState

boolean buttonState;

// длительность воспроизведения ноты

unsigned long duration;

// max и min значение длительности

#define MAX\_DURATION 3000

#define MIN\_DURATION 300

// пин подключения динамика

const int pinSpeaker=4;

// массив частот для нот первой октавы

// {до, ре, ми, фа, соль, ля, си}

int octave1[]={261,293,329,349,392,440,494};

// пин подключения потенциометра

const int pinPot=A0;

void setup() {

// обозначаем все пины как выходы

pinMode(latchPin, OUTPUT);

pinMode(clockPin, OUTPUT);

pinMode(dataPin, OUTPUT);

// определяем вывод LED (светодиод) как выход

pinMode(LED, OUTPUT);

// определяем вывод BUTTON (кнопка) как вход

pinMode(BUTTON, INPUT\_PULLUP);

// сконфигурировать контакт как выход

pinMode(pinSpeaker,OUTPUT);

}

void loop() {

int disp = 0; //создаем переменную для вывода на экран

int analogueValue = analogRead(sensorPin); // читаем аналоговый пин A0

analogueValue = map(analogueValue, 0, 1023, 0, 8888); //преобразуем диапазон с А0 (0-1023) в нужный нам (0-8888)

disp = analogueValue; // записываем получившуюся после преобразования цифру в переменну для вывода на экран

// Разбиваем цифру по разрядам индикатора

if (disp < 10) // если наша цифра меньше 10, то

{

Indicate(0, 11); // пишем в первый разряд пусто

Indicate(1, 11); // пишем во второй разряд пусто

Indicate(2, 11); // пишем в третий разряд пусто

Indicate(3, disp); // пишем в четвертый разряд нашу цифру

}

else if (disp < 100) // если наша цифра меньше 100, то

{

Indicate(0, 11); // пишем в первый разряд пусто

Indicate(1, 11); // пишем во второй разряд пусто

Indicate(2, disp / 10); // пишем в третий разряд - цифру делёную на 10

Indicate(3, disp % 10); // пишем в четвертый разряд цифру оставшуюся от деления на 10

/\*

Допустим наша цифра 25.

Если мы ее поделим на 10, то у нас получится 2,5. Цифры после запятой, в данном случае, у нас не остаются. Таким образом мы имеем в третем разряде цифру 2.

В чевертый разряд мы, как раз и записываем цифру-остаток полученную в результате деления.

В нашем случае это и есть та самая 5.

Аналогичным образом разбивается наша цифра и далее.

\*/

}

else if (disp < 1000)

{

Indicate(0, 11);

Indicate(1, disp / 100);

Indicate(2, (disp % 100) / 10);

Indicate(3, disp % 10);

}

else

{

Indicate(0, disp / 1000);

Indicate(1, (disp % 1000) / 100);

Indicate(2, (disp % 100) / 10);

Indicate(3, disp % 10);

}

// считываем состояние BUTTON входа и записываем в buttonState

buttonState = digitalRead(BUTTON);

// инверсия переменной buttonState

// для схемы с подтягивающим резистором

buttonState = ! buttonState;

// записываем состояние из buttonState на выход LED (светодиод)

digitalWrite(LED, buttonState);

// последовательное воспроизведение звуков

for(int i=0;i<7;i++) {

// вычисляем скорость воспроизведения

// (длительность ноты)

int val=analogRead(pinPot);

duration=map(val,0,1023,MIN\_DURATION,MAX\_DURATION);

// воспроизведение ноты

tone(pinSpeaker,octave1[i],duration);

delay(duration);

delay(duration/2);

}

// пауза перед следующим воспроизведением звукоряда

delay(duration\*5);

}

void Indicate(int r,int x)

{

SegDisplay=g\_digits[x]; // получаем цифру и выводим символ, из массива цифр, соответствующий этой цифре.

RazrDisplay=g\_registerArray[r]; // получаем цифру и выводим номер регистра, из массива цифр, соответствующий этой цифре.

digitalWrite(latchPin, LOW); // устанавливаем синхронизацию "защелки" на LOW

shiftOut(dataPin, clockPin, MSBFIRST, RazrDisplay); // Записываем информацию для второго регистра (Номер разряда)

shiftOut(dataPin, clockPin, MSBFIRST, SegDisplay); // Записываем информацию для первого регистра (Номер символа)

digitalWrite(latchPin, HIGH); //"защелкиваем" регистр, тем самым устанавливая значения на выходах

delay(TimeLight); // пауза, чтобы сегменты "разгорелись"

}

1. **Подключение сервопривода, датчика ультразвука к плате Arduino UNO**

**1.1 Электрическая схема**

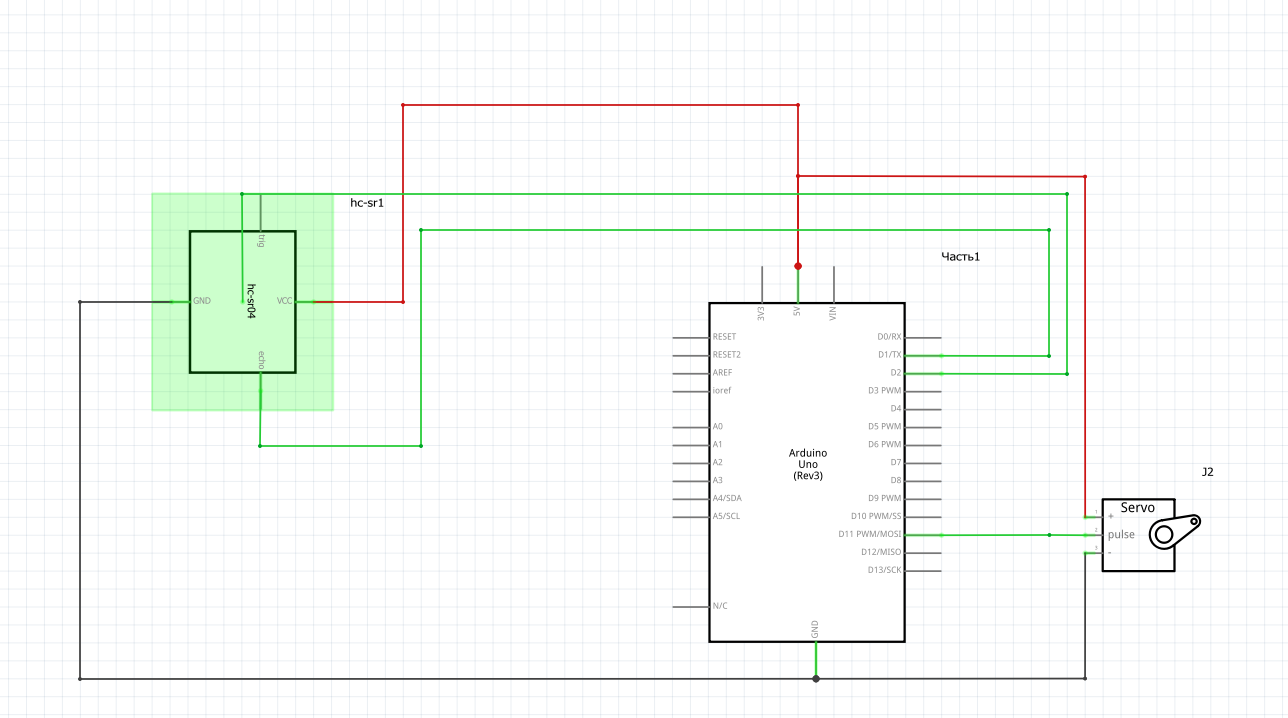


Рисунок 3.1 – Электрическая (принципиальная) схема подключения сервопривода, датчика ультразвука к плате Arduino UNO

**1.2 Монтажная схема**

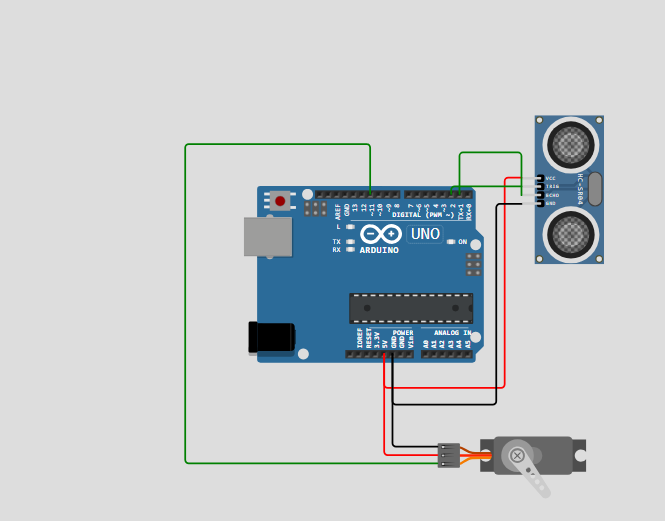


Рисунок 3.2 – Монтажная схема подключения сервопривода и ультразвука к плате Arduino UNO

**1.3 Код подключения**

#include <Servo.h>

#define TRIGGER\_PIN 2

#define ECHO\_PIN 1

#define SERVO\_PIN 11

#define STOP\_DISTANCE 19 // расстояние в сантиметрах, при котором останавливаем движение сервопривода

Servo myservo;

long duration;

int distance;

int angle = 0;

bool point = false;

void setup() {

Serial.begin(9600);

myservo.attach(SERVO\_PIN);

pinMode(TRIGGER\_PIN, OUTPUT);

pinMode(ECHO\_PIN, INPUT);

}

void loop() {

delay(15);// Отправляем ультразвуковой импульс

digitalWrite(TRIGGER\_PIN, LOW);

delayMicroseconds(2);

digitalWrite(TRIGGER\_PIN, HIGH);

delayMicroseconds(10);

digitalWrite(TRIGGER\_PIN, LOW);

// Получаем время прохождения звукового импульса

duration = pulseIn(ECHO\_PIN, HIGH);

// Преобразуем время в расстояние

distance= duration\*0.034/2;

if (distance > STOP\_DISTANCE) {

if (point == false)

{

angle = angle - 1;

}

else

{

angle = angle + 1;

}

if (angle >= 180)

{

point = false;

}

if (angle <= 0) {

point = true;

}

//myservo.write(angle);

}

myservo.write(angle);

// Задержка перед следующим измерением

//point = myservo.angle;

}

1. **Подключение датчика ультразвука и дисплея к плате Arduino**

**1.1 Электрическая схема**

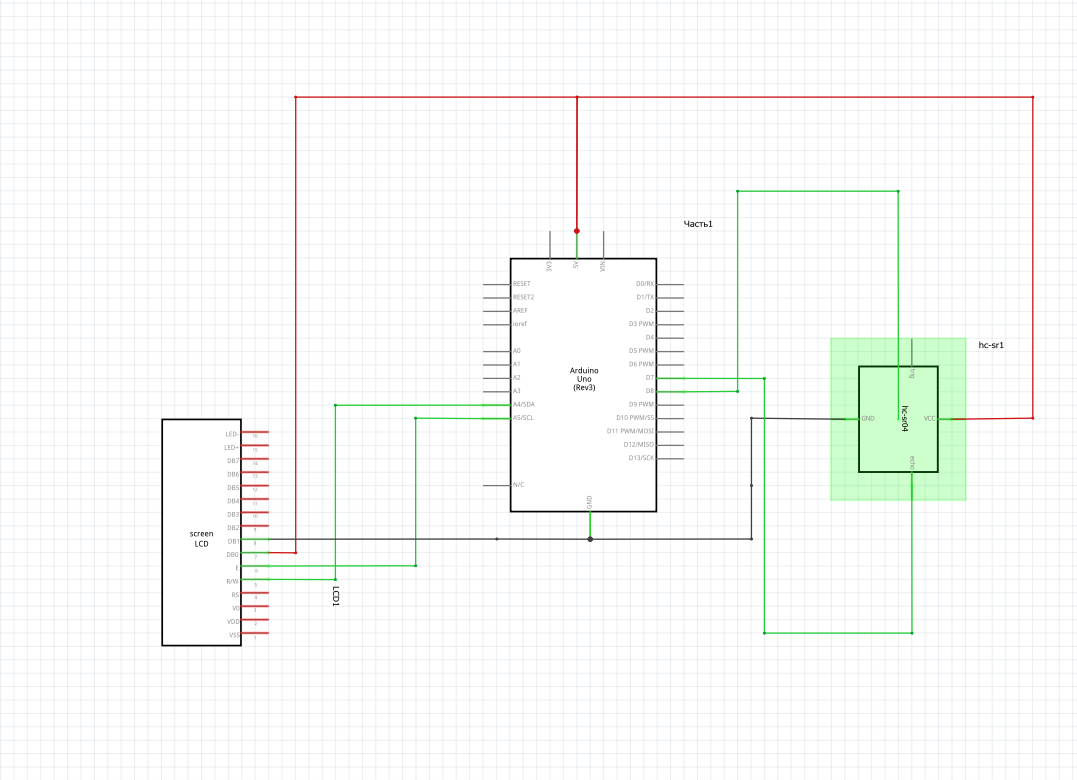


Рисунок 4.1 – Электрическая (принципиальная) схема подключения датчика ультразвука и дисплея к плате Arduino UNO

**1.2 Монтажная схема**

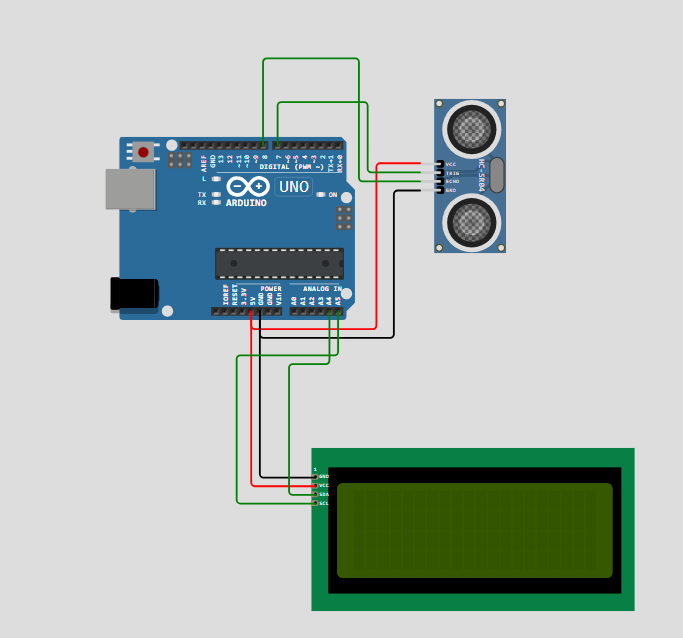


Рисунок 4.2 – Схема подключения датчика ультразвука и дисплея

**1.3 Код подключения**

#include <LiquidCrystal\_I2C.h>

int duration, cm; // назначение меры длины в см

int echoPin = 8; // назначение цифрового вывода для Echo

int trigPin = 7; // назначение цифрового вывода для Trig

LiquidCrystal\_I2C lcd(0x27,20,4); /\* Задаем адрес и размерность дисплея.

//При использовании LCD I2C модуля с дисплеем 20х04 ничего в коде изменять не требуется, cледует только задать правильную размерность \*/

void setup() {

Serial.begin (9600); // подключение монитора порта для выведения данных

pinMode(trigPin, OUTPUT); // генерируется импульс для измерения расстояния

pinMode(echoPin, INPUT); // завершение импульса, считывание дистанции

lcd.init(); // Инициализация lcd

lcd.backlight(); // Включаем подсветку

}

void loop() {

digitalWrite(trigPin, LOW); // подача низкого сигнала на пин trigPin

delayMicroseconds(2); // пауза 2 микросекунды

digitalWrite(trigPin, HIGH); // подача высокого сигнала на пин trigPin

delayMicroseconds(10); // пауза 10 микросекунд

digitalWrite(trigPin, LOW); // подача низкого сигнала на пин trigPin

duration = pulseIn(echoPin, HIGH); // pulseIn - длина положительного импульса на пине echoPin в микросекундах

cm = (duration / 2) / 29.1; // формула для выведения корректных данных в см

double distance = cm;

lcd.setCursor(3, 0);

lcd.print("Hello, world!");

lcd.setCursor(0, 3);

lcd.print("Distance is" + String(distance, 2) + "cm");

}