Sprawozdanie z Zadania 5.
Zastosowanie czujnika odległości

Adrian Lesniak nr.indeksu: 154256

1. Cel Ćwiczenia **3**
2. Opis Elementów Wykorzystanych w Ćwiczeniu **3**
Załączona Dokumentacja Techniczna - Podstawowe parametry. 4
Budowa Układu 6
3. Schemat Połączeń **7**
4. Obliczenia Rezystancji dla Diod. **8**
5. Obliczenia odległości za pomocą czujnika ultradźwiękowego **9**
6. Kod Programu z Komentarzami **10**
7. Uwagi i Wnioski **14**
8. Wykonanie projektu na odpowiednich komponentach. **15**

1. Cel Ćwiczenia

Celem ćwiczenia było zaprojektowanie i zbudowanie układu, który umożliwia pomiar odległości przy użyciu czujnika ultradźwiękowego HC-SR04. Układ miał wyświetlać zmierzoną odległość na wyświetlaczu LCD oraz za pomocą diod LED sygnalizować zakresy odległości: bardzo blisko, blisko i daleko.

2. Opis Elementów Wykorzystanych w Ćwiczeniu

Projekt wykorzystuje płytkę Arduino UNO R3, charakteryzującą się prostotą i uniwersalnością, co czyni ją idealną do celów edukacyjnych oraz prototypowych. Płyta ta jest wyposażona w mikrokontroler ATmega328, który oferuje wystarczające zasoby i porty I/O do obsługi czujnika ultradźwiękowego HC-SR04, trzech diod LED oraz wyświetlacza LCD, przy stosunkowo niskim zużyciu energii. W projekcie wykorzystano także bibliotekę LiquidCrystal_I2C.h do obsługi wyświetlacza LCD z interfejsem I2C, co umożliwia wygodną komunikację między urządzeniami za pomocą minimalnej liczby przewodów.

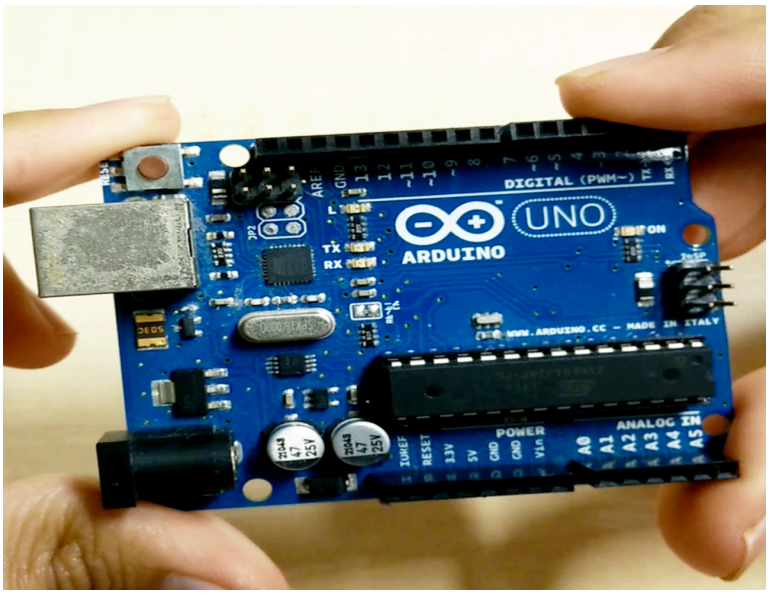
Rezystory o wartości 220Ω, które zostały użyte w projekcie, ograniczają prąd płynący przez diody LED, zapobiegając ich uszkodzeniu przez nadmierny prąd. Użycie jednolitych rezystorów upraszcza konstrukcję i pozwala na łatwe obliczenia oraz montaż układu. Czujnik ultradźwiękowy HC-SR04, użyty do pomiaru odległości, pozwala na dokładne wykrywanie obiektów w zakresie od 2 cm do 400 cm.

Wyświetlacz LCD 1602 z interfejsem I2C służy do wyświetlania pomiarów odległości oraz komunikatów o zakresie wykrycia ("Very close", "Close", "Far") w języku polskim, z zastosowaniem polskich znaków. Płytka stykowa (breadboard) umożliwia szybkie i łatwe prototypowanie układu bez konieczności lutowania, a przewody połączeniowe zapewniają elastyczność w konfiguracji układu.

- **Arduino UNO:** Platforma mikrokontrolerów używana do sterowania diodami i odczytu danych z czujnika ultradźwiękowego.
- **Dioda LED: Czerwona, żółta, zielona –** każda dioda podłączona przez rezystor 220Ω, które ograniczają prąd płynący przez diody.
- **Czujnik Ultrasoniczny HC-SR04:** Używany do pomiaru odległości od obiektów.

- **Wyświetlacz LCD 1602 z interfejsem I2C:** Służy do wyświetlania pomiarów odległości i statusu detekcji.
- **Rezystory 220Ω:** Ograniczają prąd diod LED, zapobiegając ich uszkodzeniu.
- **Płytka stykowa:** Używana do prototypowania układu.

Rys 1. Zdjęcie Arduino UNO R3



Załączona Dokumentacja Techniczna - **Podstawowe parametry.**

Tabela 2: Parametry zastosowanego wyświetlacza LCD z konwerterem I2C

Parametr	Wielkość
Rodzaj wyświetlacza	LCD 2×16 znaków
Kolor znaków	Biały
Podświetlenie	Niebieskie
Rozmiar modułu	80 × 36 mm
Wymiary znaku	2,45 × 5,00 mm
Zakres temperatury pracy	-20°C do +70°C

Tabela 1: Specyfikacja płytki Arduino UNO R3

Parametr	Wielkość
Napięcie zasilania	Od 7 V do 12 V
Mikrokontroler	ATmega328
Maksymalna częstotliwość zegara	16 MHz
Pamięć SRAM	2 kB
Pamięć Flash	32 kB
Pamięć EEPROM	1 kB
Porty I/O	14
Wyjścia PWM	6
Wejścia Analogowe	6
Interfejsy szeregowe	UART, SPI, I2C
Wymiary	68,6 × 53,4 mm

HC-SR04 Ultradźwiękowy czujnik odległości

Nazwy pinów

Nazwa	Opis
VCC	Napięcie zasilania (5V)
TRIG	Impuls do rozpoczęcia pomiaru
ECHO	Zmierz wysoką długość impulsu, aby uzyskać odległość
GND	Grunt

Budowa Układu

Układ zawiera Arduino UNO połączone z czujnikiem ultradźwiękowym HC-SR04, który mierzy odległość do obiektów. Arduino steruje trzema diodami LED (czerwoną, żółtą i zieloną) poprzez rezystory 220Ω. Każda dioda jest aktywowana w zależności od zmierzonej odległości: czerwona dioda dla bardzo małej odległości, żółta dla średniej i zielona dla dużej odległości. Wyświetlacz LCD jest podłączony do Arduino za pomocą interfejsu I2C, gdzie pin SDA jest podłączony do A4, a SCL do A5.

Połączenia Arduino UNO R3 z wyświetlaczem LCD za pomocą magistrali I2C dokonano w następujący sposób:

- VCC wyświetlacza do 5V Arduino,
- GND wyświetlacza do GND Arduino,
- SDA wyświetlacza do pinu A4 Arduino,
- SCL wyświetlacza do pinu A5 Arduino.

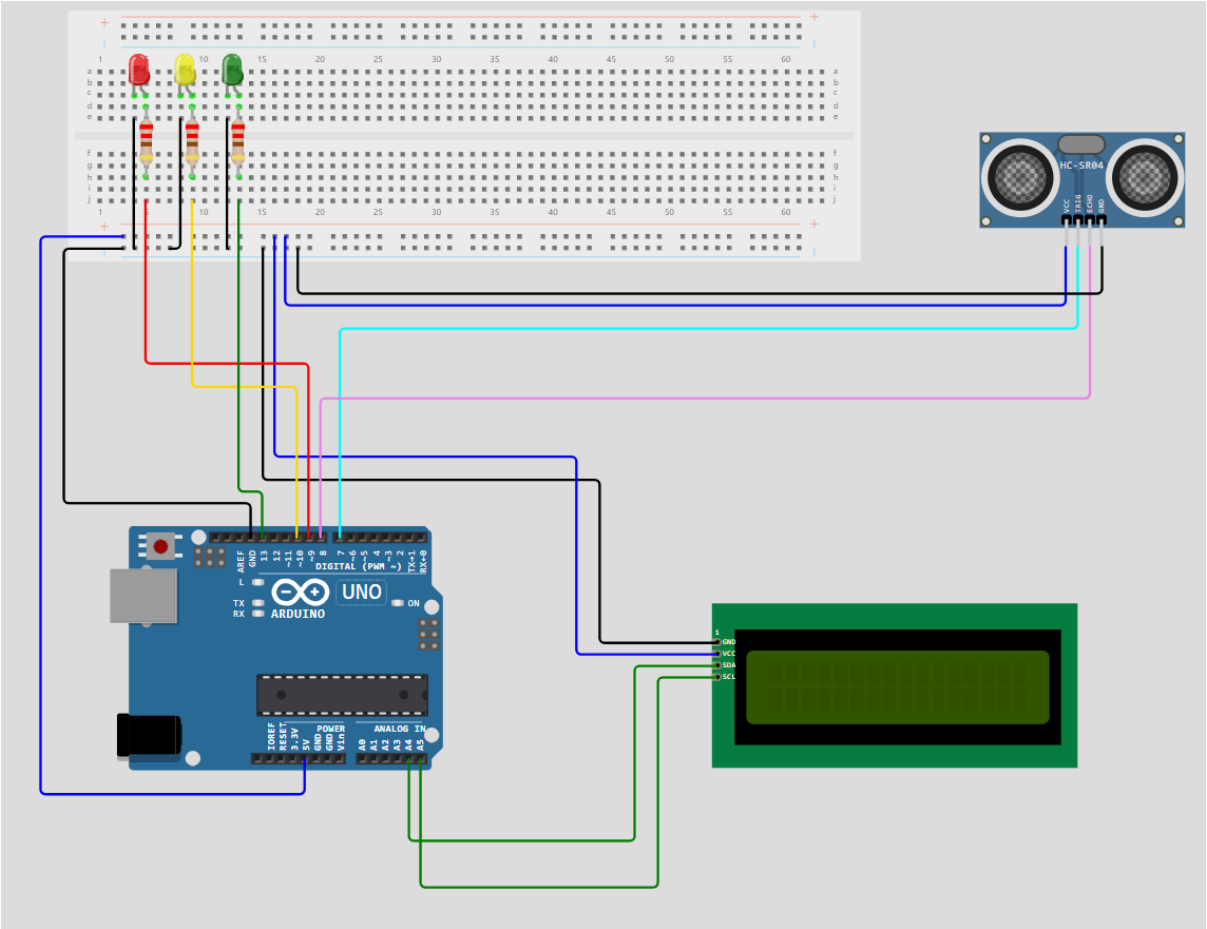
Dzięki wykorzystaniu szyny I2C, minimalizowana jest liczba potrzebnych przewodów, co upraszcza konstrukcję układu. Schemat połączeń ilustruje wszystkie połączenia, które są niezbędne do prawidłowego działania układu. Czujnik ultradźwiękowy, będący kluczowym elementem projektu, umożliwia precyzyjne pomiary odległości, co znacząco zwiększa funkcjonalność układu.

Tabela 3: Opis wyprowadzeń magistrali I2C dla wyświetlacza LCD

Nazwa	Opis	Podłączenie do Arduino
GND	Masa układu	GND
VCC	Zasilanie +5 V	5V
SDA	Linia danych magistrali I2C	A4
SCL	Linia zegarowa magistrali I2C	A5

3. Schemat Połączeń

Układ zawiera Arduino UNO połączone z czujnikiem ultradźwiękowym HC-SR04, który jest odpowiedzialny za pomiar odległości. Trzy diody LED są sterowane przez Arduino w zależności od zmierzonej odległości, sygnalizując odpowiedni zakres. Wyświetlacz LCD podłączony jest przez interfejs I2C, co minimalizuje liczbę używanych przewodów.



Rys 2. Na schemacie widoczne jest Arduino UNO, które jest połączone z czujnikiem ultradźwiękowym, trzema diodami LED (czerwoną, żółtą i zieloną) przez rezystory, oraz z wyświetlaczem LCD za pomocą interfejsu I2C.

4. Obliczenia Rezystancji dla Diod.

W zadaniu 4 użyto dwóch diod LED: czerwonej i zielonej. Zastosowano rezystory o wartości 220Ω, które ograniczają prąd płynący przez diody. Standardowe napięcie przewodzenia dla czerwonej diody wynosi około 2V, a dla zielonej 3V. Wykorzystanie tych rezystorów umożliwia bezpieczne użytkowanie diod przy zalecanym prądzie około 20 mA.

Dla każdej diody LED obliczenia wartości rezystancji dokonano z wykorzystaniem prawa Ohma:

$$R = \frac{V_s - V_f}{I}$$

gdzie:

- ***Vs*** to napięcie źródła (5V),
- ***Vf*** to spadek napięcia na diodzie (2V dla czerwonej, 3V dla zielonej),
- ***I*** to prąd diody.

Czerwona dioda:

Przy założeniu, że spadek napięcia na czerwonej diodzie wynosi 2V, a zalecany prąd diody wynosi 20 mA, obliczenia dają:

$$R = \frac{5V - 2V}{20mA} = 150\Omega$$

Jednakże, użyto rezystora 220Ω, co zwiększa bezpieczeństwo pracy diody.

Zielona dioda:

Przy spadku napięcia 3V i prądzie 20 mA, obliczenia dają:

$$R = \frac{5V - 3V}{20mA} = 100\Omega$$

Podobnie, zastosowanie rezystora 220Ω jest odpowiednie dla bezpiecznej pracy. Te obliczenia pokazują, że wartości rezystorów 220Ω są bezpieczne i odpowiednie dla obu diod, nawet jeśli nie są optymalne z matematycznego punktu widzenia. Pozwala to na uproszczenie układu i minimalizację ryzyka przepalenia diod LED.

5. Obliczenia odległości za pomocą czujnika ultradźwiękowego

Czujnik ultradźwiękowy HC-SR04 pracuje na zasadzie echolokacji, podobnie jak używana przez nietoperze czy delfiny. Czujnik wysyła fale dźwiękowe o bardzo wysokiej częstotliwości (ultradźwięki), które są niesłyszalne dla ludzkiego ucha, a następnie oczekuje na ich powrót jako echo po odbiciu się od przeszkody.

Proces pomiaru odległości wygląda następująco:

1. **Inicjacja pomiaru:** Arduino wysyła sygnał do czujnika, aby wyemitował krótki impuls ultradźwiękowy. Ten impuls jest generowany przez wysłanie wysokiego sygnału (HIGH) na pinie TRIGGER czujnika na krótki czas (zazwyczaj 10 mikrosekund).
2. **Emisja ultradźwięków:** Po odebraniu sygnału trigger, czujnik emituje serię ultradźwiękowych fal dźwiękowych, które rozchodzą się w powietrzu.
3. **Odbiór echa:** Echo fal ultradźwiękowych odbitych od przeszkody jest odbierane przez czujnik na pinie ECHO. Czas, który upłynął od wysłania impulsu do odbioru echa, jest mierzony przez Arduino.
4. **Obliczenie odległości:** Dystans do przeszkody jest obliczany na podstawie czasu trwania echa, wykorzystując zależność, gdzie prędkość dźwięku w powietrzu wynosi około 343 m/s (w temperaturze około 20°C). Dlatego, aby obliczyć odległość w centymetrach, mierzony czas (w mikrosekundach) jest mnożony przez 0.034 i dzielony przez 2 (ponieważ dźwięk musi przebyć drogę do przeszkody i z powrotem).

Formuła:

$$\text{Odległość (cm)} = \frac{\text{Czas trwania echa (µs)} \times 0.034}{2}$$

Zastosowanie tej metody pozwala na bardzo precyzyjne i szybkie pomiary odległości, co jest kluczowe w wielu zastosowaniach inżynierskich i robotycznych. Dzięki skutecznemu wykorzystaniu echolokacji, układ jest w stanie nie tylko wykrywać obiekty, ale także określać ich odległość, co jest wyświetlane na LCD oraz sygnalizowane za pomocą diod LED w zależności od zakresu odległości.

6. Kod Programu z Komentarzami

Program na Arduino odpowiada za pomiar odległości przy użyciu czujnika ultradźwiękowego HC-SR04. W zależności od zmierzonej odległości, program steruje trzema diodami LED: czerwoną (dla bardzo małej odległości), żółtą (dla średniej odległości) i zieloną (dla dużej odległości), jednocześnie wyświetlając odpowiedni status na wyświetlaczu LCD. Kod zapewnia aktualizację stanu tylko wtedy, gdy zmierzona odległość różni się od poprzednio zarejestrowanej, co zmniejsza częstotliwość odświeżania i poprawia czytelność wyświetlanych informacji.

Kod programu jest zorganizowany i komentowany w sposób, który ułatwia zrozumienie jego działania:

1. **Inicjalizacja komponentów:** Ustawienie pinów dla czujnika, diod LED oraz inicjalizacja wyświetlacza LCD z interfejsem I2C.
2. **Główna pętla programu:** Cykliczne wykonywanie pomiaru odległości, wyświetlanie wyniku na LCD i sterowanie diodami LED zależnie od zmierzonej odległości.
3. **Logika wyświetlania i sygnalizacji:** Warunki decydujące o tym, która dioda LED ma być aktywowana oraz jakie informacje mają być prezentowane na wyświetlaczu w zależności od odległości.
4. **Optymalizacja wydajności:** Zastosowanie opóźnień i warunków, które ograniczają niepotrzebne odświeżania i aktualizacje, co przekłada się na bardziej stabilne działanie urządzenia.

Dzięki strukturze i komentarzom, kod jest łatwy do modyfikacji i rozszerzenia, co pozwala na dalsze eksperymentowanie i dostosowanie programu do własnych potrzeb.

Kod źródłowy został opracowany zgodnie z poniższym listingiem:

sketch.ino ● diagram.json libraries.txt Library Manager

```
#include <LiquidCrystal_I2C.h> // Dołączenie biblioteki do obsługi wyświetlacza LCD z interfejsem I2C

LiquidCrystal_I2C lcd(0x27, 16, 2); // Inicjalizacja obiektu LCD z podanym adresem I2C, ilością znaków w wierszu (16) oraz ilością wierszy (2)

byte customCharL[8] = {B01100, B00100, B00110, B00100, B01100, B00100, B01110, B00000}; // Definicja znaku specjalnego 'ł'
byte customChars[8] = {B00010,B00100,B01110,B10000,B01110,B00001,B11110,B00000}; // Definicja znaku specjalnego 'ś'
byte customCharc[8] = {B00010,B00100,B01110,B10001,B10000,B10001,B01110,B00000}; // Definicja znaku specjalnego 'ć'

const int TRIGGER_PIN = 7;// Pin wyzwalający sygnał ultradźwiękowy
const int ECHO_PIN = 8;// Pin odbierający sygnał ultradźwiękowy
const int redLED = 9;// Pin sterujący czerwoną diodą LED
const int yellowLED = 10;// Pin sterujący żółtą diodą LED
const int greenLED = 13;// Pin sterujący zieloną diodą LED

unsigned int previousDistance = 0; // Zmienna do przechowywania poprzedniej zmierzonej odległości

void setup() { // Funkcja setup, wywoływana raz podczas startu programu
  lcd.init();// Inicjalizacja wyświetlacza LCD
  lcd.backlight();// Włączenie podświetlenia LCD
  lcd.createChar(0, customCharL); // Tworzenie znaku specjalnego 'ł'
  lcd.createChar(1, customCharS); // Tworzenie znaku specjalnego 'ś'
  lcd.createChar(2, customCharC); // Tworzenie znaku specjalnego 'ć'

  Serial.begin(9600);// Rozpoczęcie komunikacji szeregowej z prędkością 9600 bps
  pinMode(TRIGGER_PIN, OUTPUT);// Ustawienie pinu wyzwalającego jako wyjście
  pinMode(ECHO_PIN, INPUT);// Ustawienie pinu echo jako wejście
  pinMode(redLED, OUTPUT);// Ustawienie pinu czerwonej diody LED jako wyjście
  pinMode(yellowLED, OUTPUT);// Ustawienie pinu żółtej diody LED jako wyjście
  pinMode(greenLED, OUTPUT);// Ustawienie pinu zielonej diody LED jako wyjście
}

void loop() {// Funkcja loop, wywoływana w pętli (ciągle powtarzana przez mikrokontroler)
  digitalWrite(TRIGGER_PIN, LOW);// Ustawienie niskiego poziomu na pinie wyzwalającym
  delayMicroseconds(2);// Krótkie opóźnienie (2 mikrosekundy)
  digitalWrite(TRIGGER_PIN, HIGH);// Ustawienie wysokiego poziomu na pinie wyzwalającym
  delayMicroseconds(10);// Krótkie opóźnienie (10 mikrosekund)
  digitalWrite(TRIGGER_PIN, LOW);// Powrót do niskiego poziomu

  unsigned long duration = pulseIn(ECHO_PIN, HIGH);// Odczyt czasu trwania impulsu zwrotnego
  unsigned int distance = duration * 0.034 / 2;// Obliczenie odległości w centymetrach

  if (distance != previousDistance) { // Jeśli zmierzona odległość różni się od poprzedniej
    lcd.clear();// Czyszczenie wyświetlacza
    lcd.print("Odleg");// Wyświetlenie tekstu "Odległość:"
    lcd.write(byte(0)); // 'ł'
    lcd.print("o");// Wyświetlenie tekstu "Odległość:"
    lcd.write(byte(1)); // Wstawienie znaku specjalnego "ś"
    lcd.write(byte(2)); // Wstawienie znaku specjalnego "ć"
    lcd.print(": ");// Wyświetlenie tekstu "Odległość:"
    lcd.print(distance);// Wyświetlenie zmierzonej odległości
    lcd.print("cm");// Dodanie jednostki cm

    Serial.print("Odległość: ");// Wysłanie tekstu "Odległość:" do monitora szeregowego
    Serial.print(distance);// Wysłanie zmierzonej odległości
    Serial.println("cm");// Wysłanie jednostki cm i nowej linii

    if (distance > 0 && distance < 10) {// Warunki do sterowania diodami LED w zależności od odległości
      lcd.setCursor(0, 1);// Ustawienie kursora na drugim wierszu wyświetlacza
      lcd.print("Bardzo blisko");// Wyświetlenie informacji "Bardzo blisko"
      digitalWrite(redLED, HIGH);// Zaświecenie czerwonej diody
      digitalWrite(yellowLED, LOW);// Wyłączenie żółtej diody
      digitalWrite(greenLED, LOW);// Wyłączenie zielonej diody
    } else if (distance >= 10 && distance < 50) {
      lcd.setCursor(0, 1);// Ustawienie kursora na drugim wierszu
      lcd.print("Blisko");// Wyświetlenie informacji "Blisko"
      digitalWrite(redLED, LOW);// Wyłączenie czerwonej diody
      digitalWrite(yellowLED, HIGH);// Zaświecenie żółtej diody
      digitalWrite(greenLED, LOW);// Wyłączenie zielonej diody
    } else {
      lcd.setCursor(0, 1);// Ustawienie kursora na drugim wierszu
      lcd.print("Daleko");// Wyświetlenie informacji "Daleko"
      digitalWrite(redLED, LOW);// Wyłączenie czerwonej diody
      digitalWrite(yellowLED, LOW);// Wyłączenie żółtej diody
      digitalWrite(greenLED, HIGH);// Zaświecenie zielonej diody
    }
    previousDistance = distance; // Zapisanie nowej wartości jako poprzednia odległość
  }

  delay(500); // Opóźnienie o 500 milisekund dla kontroli szybkości odświeżania
}

```

Szczegółowe omówienie dostarczonego kodu z uwzględnieniem komentarzy wyjaśniających funkcję każdej linii.

Projekt składa się z kilku kroków:

- **Inicjalizacja i konfiguracja płytki Arduino oraz wyświetlacza LCD:** Na początku wykorzystano bibliotekę LiquidCrystal_I2C do komunikacji z wyświetlaczem LCD poprzez magistralę I2C. Adres I2C wyświetlacza to 0x27, a jego rozmiar to 16 znaków w dwóch wierszach. Płytkę Arduino UNO skonfigurowano, aby sterować czujnikiem ultradźwiękowym HC-SR04 oraz trzema diodami LED (czerwoną, żółtą i zieloną) za pośrednictwem wyjść cyfrowych.
- **Konfiguracja pinów i początkowe ustawienia czujnika ultradźwiękowego oraz diod LED:** Pin cyfrowy dla czujnika ultradźwiękowego oraz dla diod LED został skonfigurowany w funkcji setup() w kodzie Arduino. Na początku działania programu wszystkie diody są wyłączone, a czujnik jest gotowy do pomiarów.
- **Debouncing i ochrona przed przetrzymaniem przycisku:** Implementacja debouncingu przycisku jest niepotrzebna, ponieważ projekt nie wykorzystuje przycisków do sterowania diodami. Zamiast tego, diody są sterowane dynamicznie na podstawie odczytanej odległości.
- **Zaprogramowanie logiki reagowania na zmierzoną odległość:** Program dynamicznie wyświetla odległość i steruje diodami LED w zależności od odczytanej odległości. Wykorzystano funkcje digitalWrite() do kontrolowania stanu diod, a aktualny pomiar odległości decyduje o załączeniu odpowiedniej diody.
- **Aktualizacja wyświetlacza LCD i komunikaty w monitorze szeregowym:** Po każdym pomiarze, wyświetlacz LCD jest aktualizowany, aby pokazać zmierzoną odległość oraz informację o stanie ("Bardzo blisko", "Blisko", "Daleko"). Równocześnie, odpowiedni komunikat jest wysyłany do monitora portu szeregowego, co umożliwia śledzenie działania układu także przez port szeregowy Arduino.
- **Monitorowanie pomiarów odległości:** Program cały czas monitoruje odległość za pomocą czujnika ultradźwiękowego i na tej podstawie decyduje o stanie diod, co zapewnia precyzyjne i niezawodne działanie układu bez blokowania wykonywania innych zadań.
- **Testowanie i debugowanie programu:** Po skonfigurowaniu układu i wgraniu programu, przeprowadzono szereg testów, aby upewnić się, że czujnik i diody działają prawidłowo oraz że informacje na LCD oraz w monitorze szeregowym są zgodne z oczekiwaniami.

W ramach projektu zaimplementowano systematyczne podejście do budowy i programowania układu, co pozwoliło na głębsze zrozumienie zarówno hardware'u, jak i software'u używanego w projekcie. Użycie funkcji opartych o czas zamiast blokującej funkcji delay() znacząco poprawia wydajność i reaktywność systemu.

7. Testowanie i Uruchomienie Układu.

Układ był testowany w środowisku Wokwi, gdzie można było wizualnie zweryfikować poprawność działania czujnika ultradźwiękowego, diod LED oraz wyświetlacza. Odczyty odległości generowane przez czujnik ultradźwiękowy wpływały na sterowanie diodami LED (czerwoną dla bardzo małej odległości, żółtą dla średniej odległości i zieloną dla dużej odległości) oraz na aktualizację wyświetlacza LCD, który pokazywał mierzoną odległość oraz odpowiedni komunikat stanu.

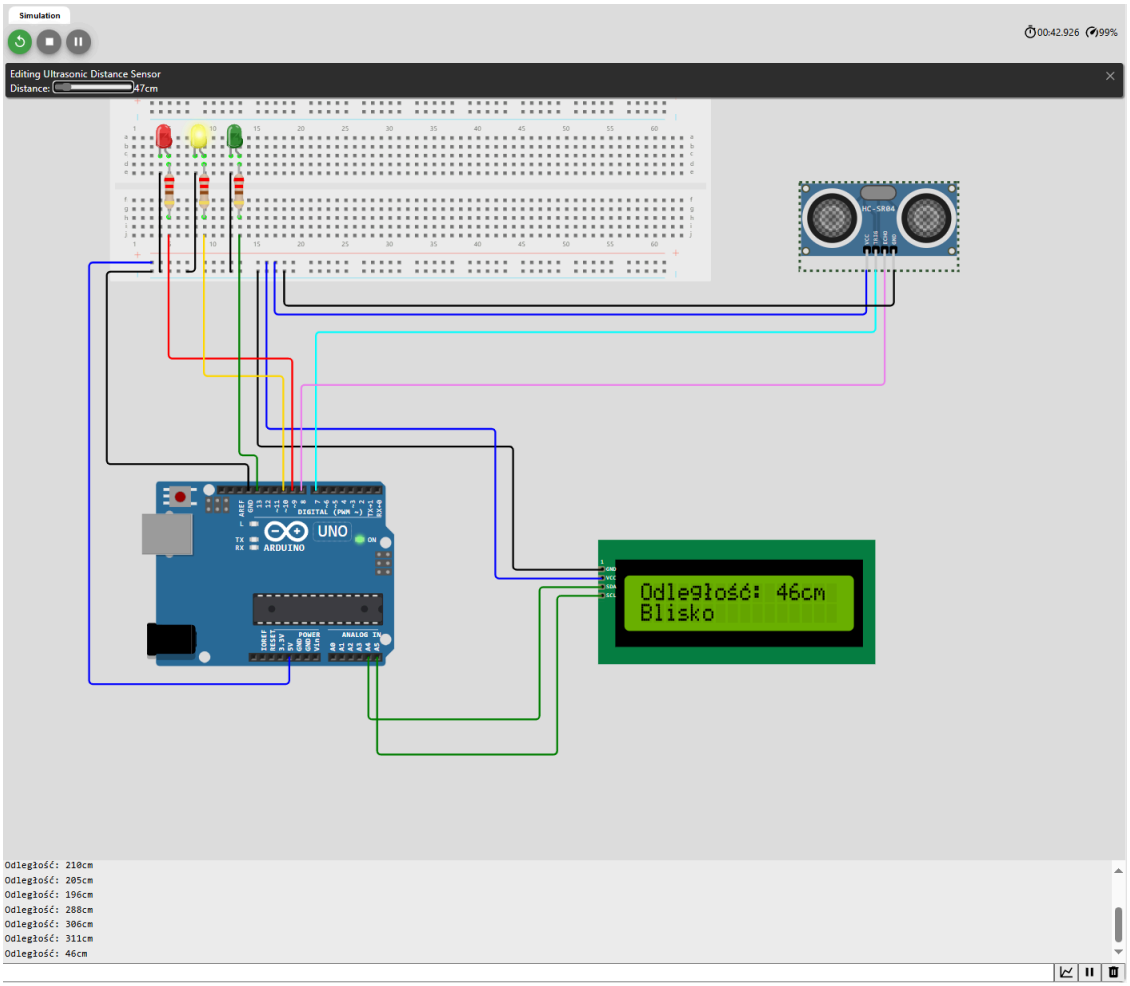
- Działanie programu dla zadania 5, które obejmuje pomiar odległości za pomocą czujnika ultradźwiękowego i Arduino, jest zorganizowane w sposób zapewniający jego funkcjonalność oraz niezawodność. Po uruchomieniu programu, rozpoczyna się od konfiguracji wyświetlacza LCD i portu szeregowego, które są niezbędne do komunikacji i wizualizacji stanu układu. Równocześnie, piny cyfrowe odpowiedzialne za sterowanie czujnikiem i diodami są ustawione jako wyjścia.

- Program korzysta z funkcji `pulseIn()` do odczytu czasu powrotu echa ultradźwięków, co umożliwia precyzyjne obliczenie odległości bez zatrzymywania działania kodu funkcją `delay()`. Zmiana odległości wykrywana przez czujnik jest rejestrowana przez program, który aktualizuje informacje na wyświetlaczu LCD i wysyła odpowiednie komunikaty do monitora portu szeregowego, informując o mierzonej odległości.

- Cały proces jest powtarzany w nieskończoność, dopóki układ jest zasilany. Program ciągle monitoruje zmiany odległości i odpowiednio reaguje, co pozwala na efektywną pracę układu bez niepotrzebnego obciążania procesora czekaniem.

- W trakcie rozwoju programu regularnie przeprowadzane były testy, aby upewnić się, że czujnik ultradźwiękowy, diody LED oraz wyświetlacz działają zgodnie z oczekiwaniami. Debugowanie było konieczne, gdy pojawiły się problemy z dokładnością pomiarów lub z komunikacją między Arduino a wyświetlaczem, co umożliwiło skuteczne rozwiązanie tych problemów i optymalizację działania układu.

Rys 3. Zdjęcie przedstawiające uruchomiony program z wyświetlaczem LCD, czujnikiem ultradźwiękowym, diodami i płytką stykową.



7. Uwagi i Wnioski

1. Projekt ten pozwolił na zrozumienie implementacji debouncingu przycisku oraz ochrony przed jego przetrzymaniem, co znacznie zwiększa niezawodność działania układu. Te techniki zapobiegają przypadkowemu załączaniu i wyłączaniu przekaźnika, co jest kluczowe dla stabilnego działania systemu.
2. Dzięki użyciu wyświetlacza LCD z interfejsem I2C, komunikacja z wyświetlaczem została uproszczona, co znacząco zmniejszyło liczbę wymaganych połączeń. To nie tylko upraszcza konstrukcję, ale również ogranicza potencjalne punkty awarii.

3. Programowanie mikrokontrolera do zarządzania stanem przekaźnika i diod LED bez użycia blokującej funkcji `delay()` umożliwiło lepszą responsywność i wydajność układu. Zastosowanie funkcji `millis()` do zarządzania czasem pozwala na płynne przełączanie stanów bez przerywania innych zadań wykonywanych przez mikrokontroler.
4. Projekt oferuje również potencjał dalszej rozbudowy. Można by dodać zdalne sterowanie przy użyciu Bluetooth lub WiFi, co umożliwiłoby kontrolę systemu z poziomu aplikacji mobilnej lub przez internet. Istnieje również możliwość rozbudowy o dodatkowe sensory, które mogłyby automatycznie kontrolować przekaźnik na podstawie określonych warunków, np. sensor temperatury decydujący o załączeniu wentylatora.

Ten projekt nie tylko zilustrował podstawowe koncepty elektroniki i programowania, ale także pokazał, jak można skalować i modyfikować proste układy do bardziej złożonych aplikacji, oferując solidne podstawy do dalszego rozwoju i eksperymentowania.

8. Wykonanie projektu na odpowiednich komponentach.

Wgranie programu (sketchu) do płytki Arduino UNO R3 za pomocą Arduino IDE obejmuje kilka kroków:

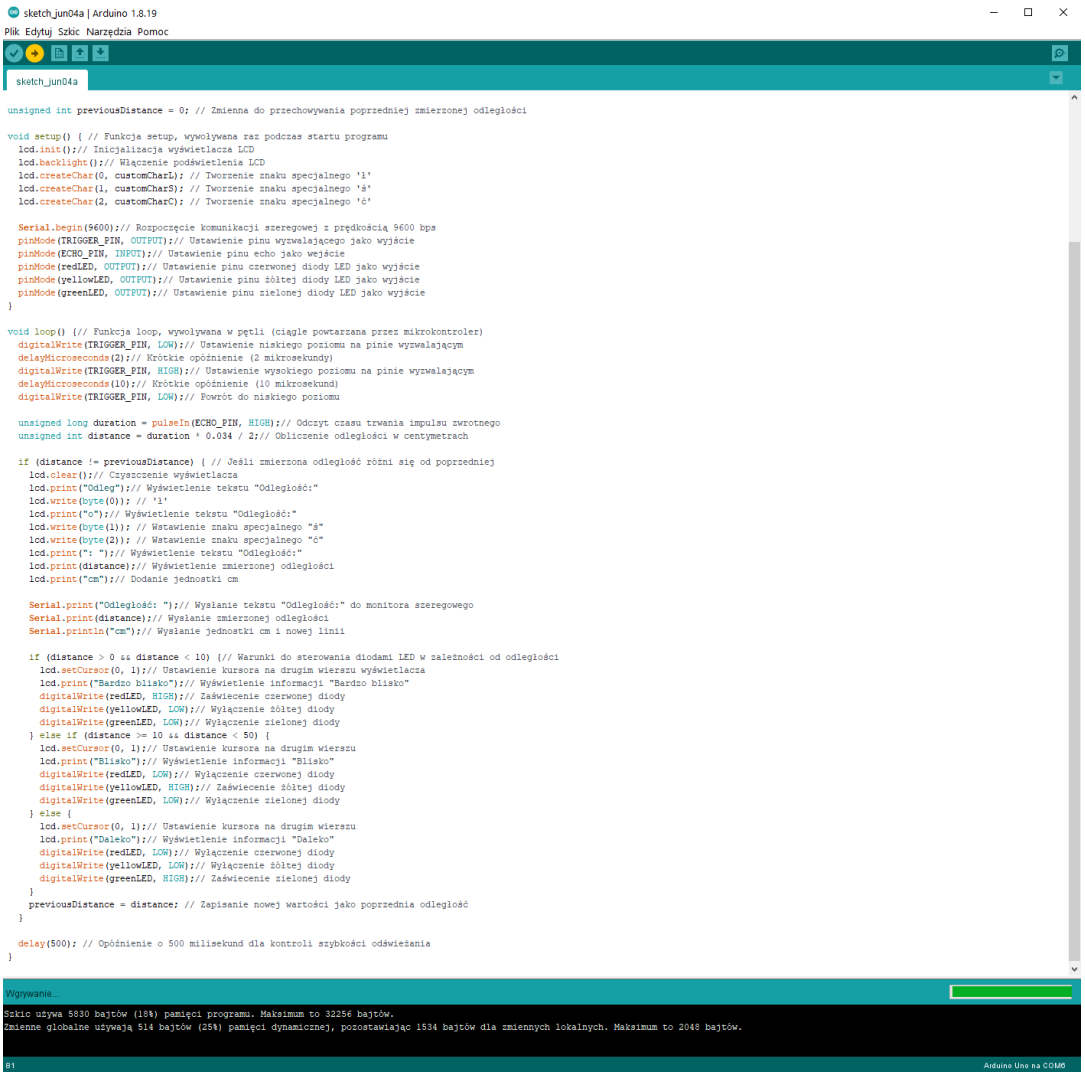
1. **Podłączenie Arduino do komputera:**
    - Użycie kabla USB do podłączenia Arduino UNO R3 do portu USB komputera zapewnia fizyczne połączenie potrzebne do programowania płytki.

2. **Otwarcie Arduino IDE:**
    - Uruchomienie Arduino IDE na komputerze pozwala na dostęp do narzędzi niezbędnych do skompilowania i wgrania kodu do mikrokontrolera.

3. **Wybranie Płytki:**
    - W narzędziach Arduino IDE wybranie opcji "Arduino/Genuino UNO" jako docelowej płytki, na której zostanie uruchomiony program.

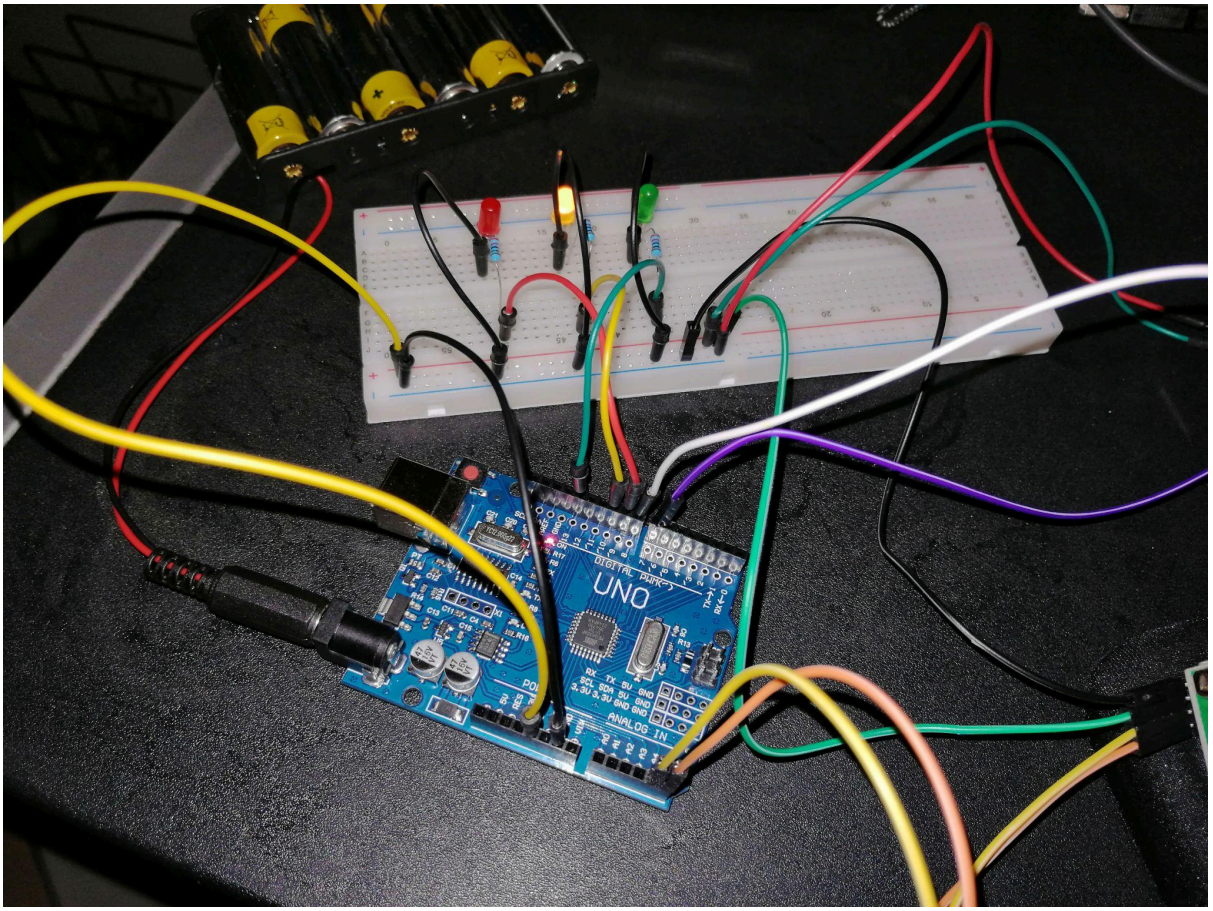
4. **Wybranie Portu:**
    - Wybranie portu, do którego podłączone jest Arduino (COM na Windows lub /dev/tty na macOS i Linux), jest kluczowe dla komunikacji między komputerem a mikrokontrolerem.

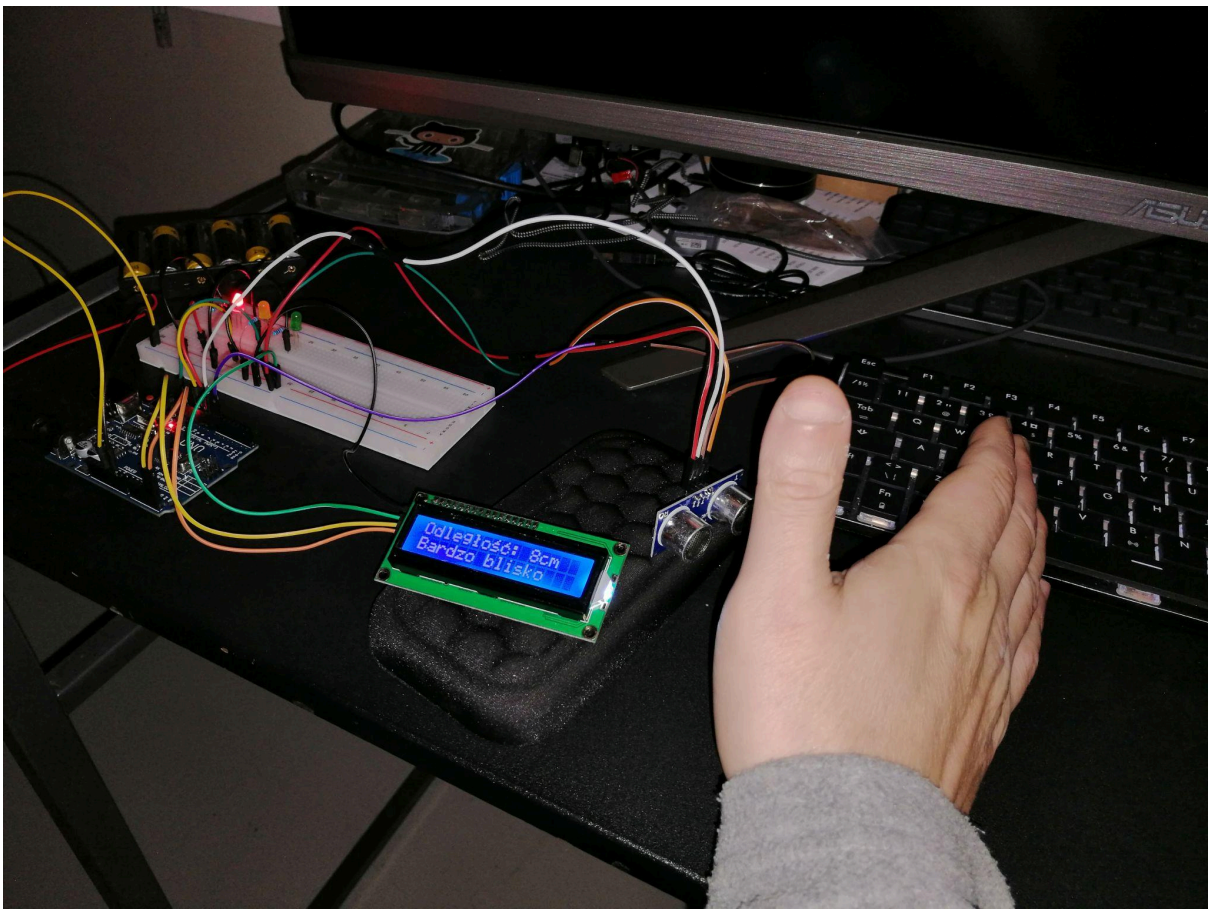
5. **Wgranie Programu:**
    - Skopiowanie programu (sketchu) do nowego okna w Arduino IDE.
    - Naciśnięcie przycisku "Weryfikuj" (ikona z symbolem kreski zakończonej znakiem "✓") w górnym lewym rogu pozwoli skompilować kod i sprawdzić, czy nie ma błędów.
    - Naciśnięcie przycisku "Wgraj" (ikona strzałki wskazującej w prawo), aby wgrać skompilowany program na płytkę Arduino.

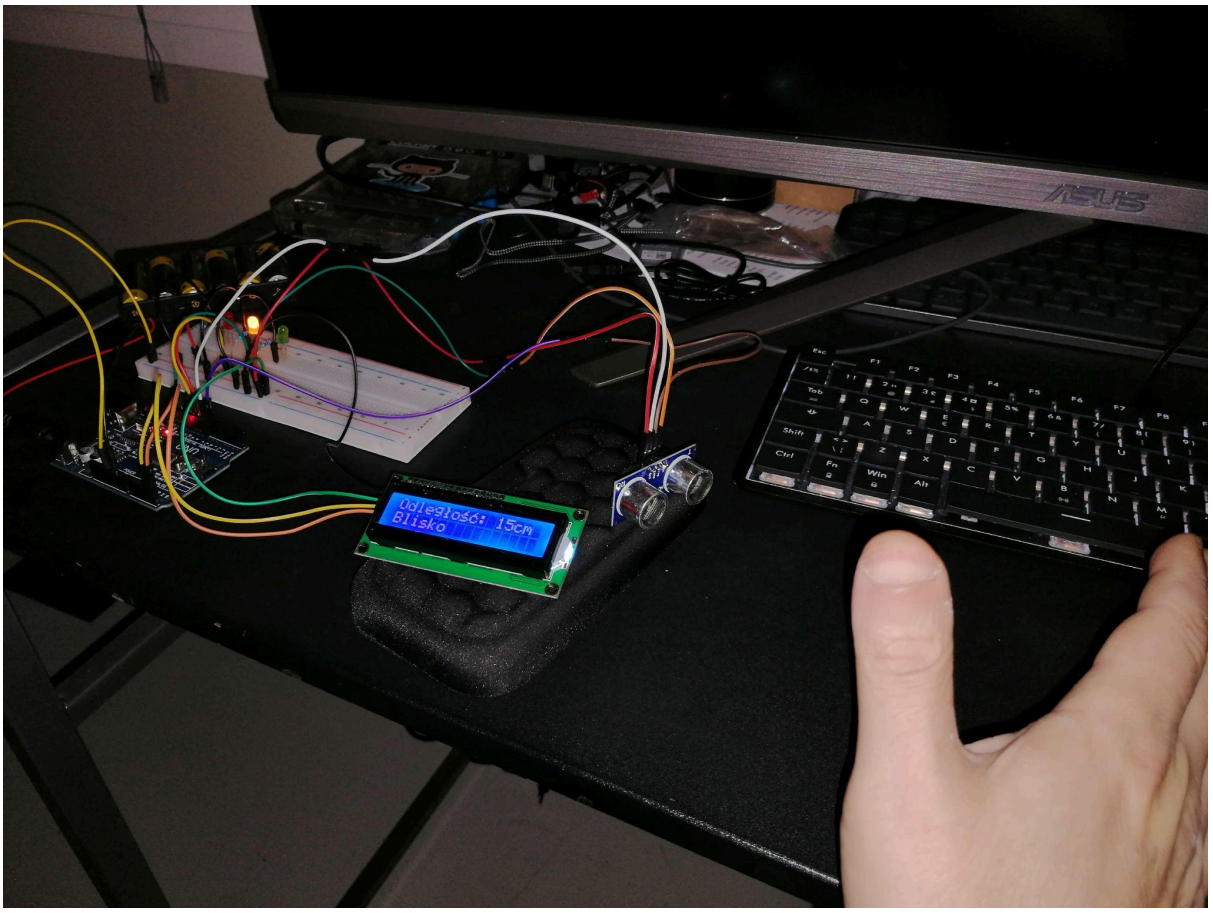


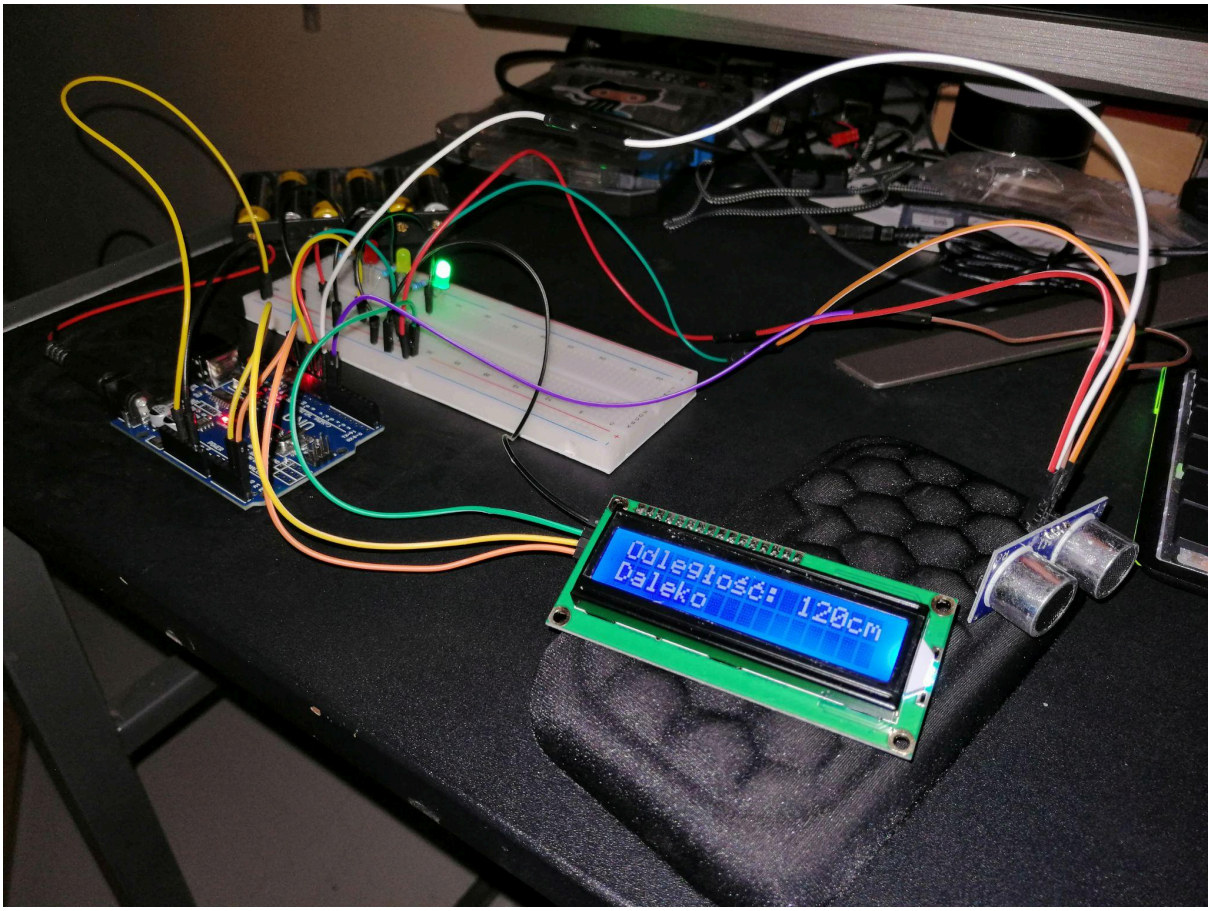
6. **Czekanie na Zakończenie**:
    - Arduino IDE pokaże postęp wgrywania na dole okna. Po zakończeniu powinien pojawić się komunikat "Wgrano".

7. **Testowanie**:
    - **Uruchomienie**: Po pomyślnym wgraniu programu, Arduino automatycznie zrestartuje się i zacznie wykonywać wgrany program.
    - **Weryfikacja działania**: Sprawdzenie, czy wyświetlacz LCD pokazuje odpowiedni tekst, odzwierciedlający zmierzoną odległość oraz czy diody LED odpowiednio reagują na zmiany odległości. Sprawdzanie poprawności działania czujnika ultradźwiękowego oraz aktualizacji wyświetlanych informacji.




Odległość: 8cm
Bardzo blisko


Odległość: 15cm
Blisko


Odległość: 120cm
Daleko