

PRACOWNIA MIERNICTWA ELEKTRONICZNEGO/ INTERNETU RZECZY

ĆWICZENIE NR 6

Temat:

Budowa prostego dalmierza

Opracowanie wstępne:
Bezpieczeństwo Internetu Rzeczy

Komponenty:

Arduino
Tact switch THT

Czujnik odległości HC-SR04
Wyświetlacz LCD 2x16

Potencjometr THT
Rezystor 100 Ω



Czujnik odległości HC-SR04

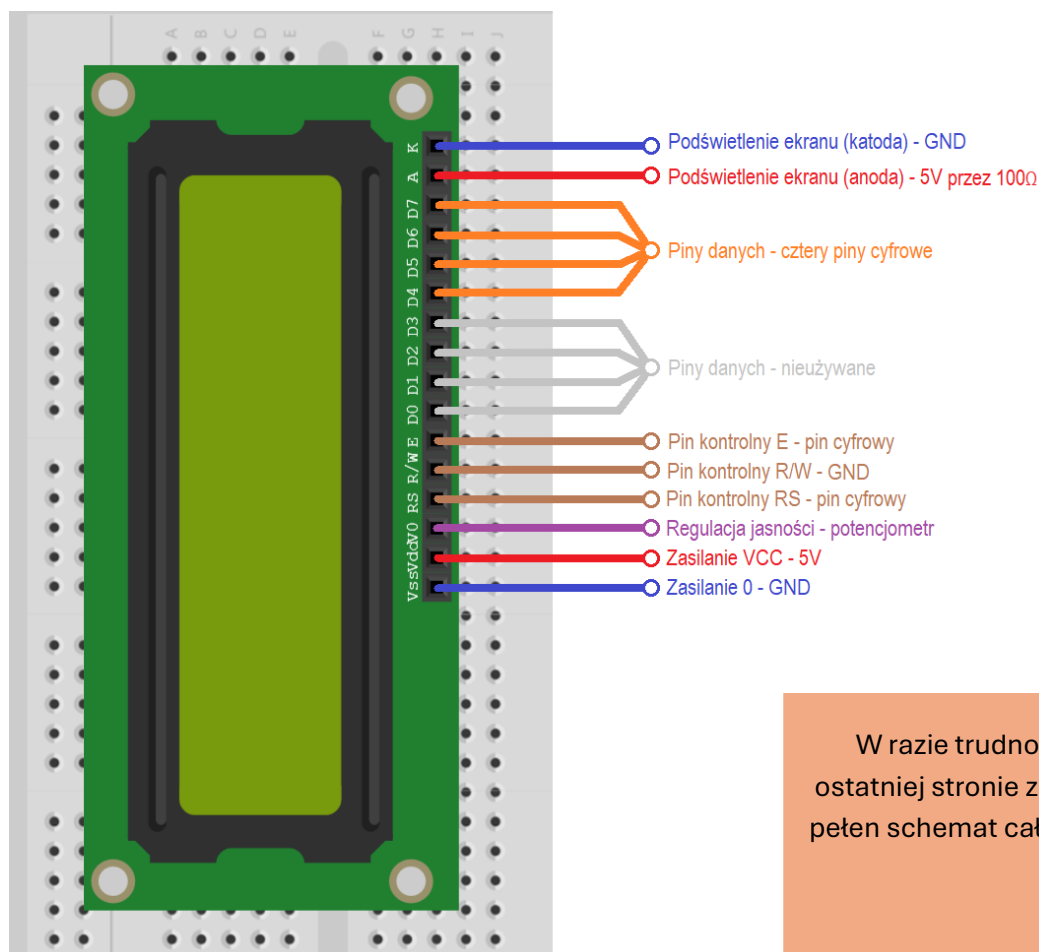
Cel ćwiczenia:

Złożenie i zaprogramowanie układu, który mierzy odległość do przedmiotu położonego przed czujnikiem odległości i wyświetla ją na ekranie LCD. Wyniki mają wyświetlać się w metrach i centymetrach. Wyświetlaną jednostkę można zmienić przy pomocy guzika.

INSTRUKCJA

Krok 1.

Należy podłączyć ekran LCD według poniższego opisu pinów i za pomocą odpowiedniego kodu wyświetlić napis:



```
#include <LiquidCrystal.h> // inicjowanie biblioteki LiquidCrystal
LiquidCrystal lcd(1, 3, 4, 5, 6, 7);
void setup() {
  lcd.begin(16, 2); // Specyfikacja ekranu (szerokość, wysokość)
}

void loop() {
  lcd.setCursor(0,0);
  lcd.print("Pomiar");
  lcd.setCursor(0,1);
  lcd.print("odleglosci");
}
```

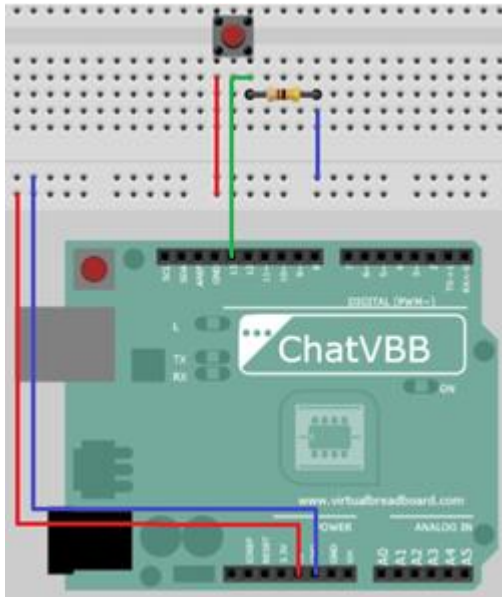
W powyższym kodzie korzystamy z biblioteki `LiquidCrystal.h`, która zawiera szereg obiektów i funkcji ułatwiających korzystanie z ekranu LCD (ale możliwe jest również bezpośrednie zarządzanie ekranem). Na początku tworzymy obiekt `lcd` – parametrami wejściowymi są tu kolejno: pin E, pin RS oraz cztery kolejne piny danych – D4-D7. W metodzie `lcd.init` podajemy rozmiar okna: 16 pól i 2 linie. Pin 2 ustawiamy na `LOW` – jest to pin

odczytu/zapisu (R/W), w naszym przypadku ustawiony na stałe na LOW (alternatywnie może zostać podłączony do GND). W bloku `loop` ustawiamy kursor przy pomocy `lcd.setCursor` i wyświetlamy napis przy pomocy `lcd.print`.

Krok 2.

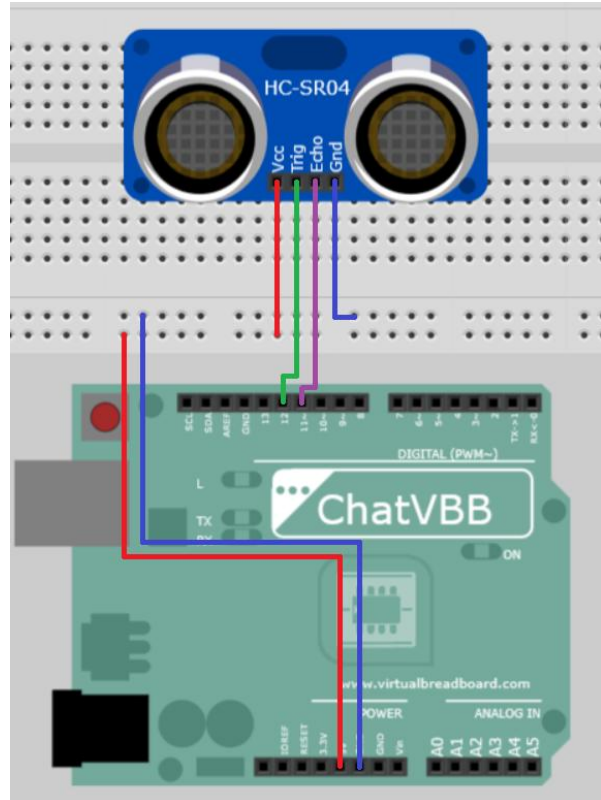
Układ należy uzupełnić o

Podłączenie przycisku:



UWAGA: Zaprezentowane schematy podłączenia są orientacyjne – numery pinów powinny być dostosowane do konkretnego układu i kodu

Podłączenie czujnika odległości:



```
#include <LiquidCrystal.h>
LiquidCrystal lcd(1, 2, 4, 5, 6, 7);
const int trigPin = 12;
const int echoPin = 11;
const int buttonPin = 9;
int unit = 0;
```

```
void setup() {
  lcd.begin(16, 2);
  pinMode(trigPin, OUTPUT);
  pinMode(echoPin, INPUT);
  pinMode(buttonPin, INPUT);
}

void display(double value) {
  lcd.print("Dystans: ");
  lcd.setCursor(0, 1);
  if (unit == 0) {
    lcd.print(value, 2);
    lcd.print("cm");
  }
  if (unit == 1) {
```

```

        lcd.print(value, 4);
        lcd.print("m");
    }
    delay(1000);
    lcd.noCursor();
    lcd.clear();
}

double distance() {
    double time = 0;
    double distance;
    digitalWrite(trigPin, LOW);
    delayMicroseconds(2);
    digitalWrite(trigPin, HIGH);
    delayMicroseconds(10);
    digitalWrite(trigPin, LOW);
    time = pulseIn(echoPin, HIGH);
    distance = distance + time / 5.9;
    distance = distance / 10;
    if (unit == 1) distance = distance / 100;
    return distance;
}

void loop() {
    int sensorVal = digitalRead(buttonPin);
    if (sensorVal == 1) {
        if (unit == 0) { unit = 1; }
        else { unit = 0; }
    }
    display(distance());
}

```

Opracowanie końcowe:

Proszę omówić kilka przykładów systemów Internetu Rzeczy, które opierają się na pośrednim pomiarze odległości

