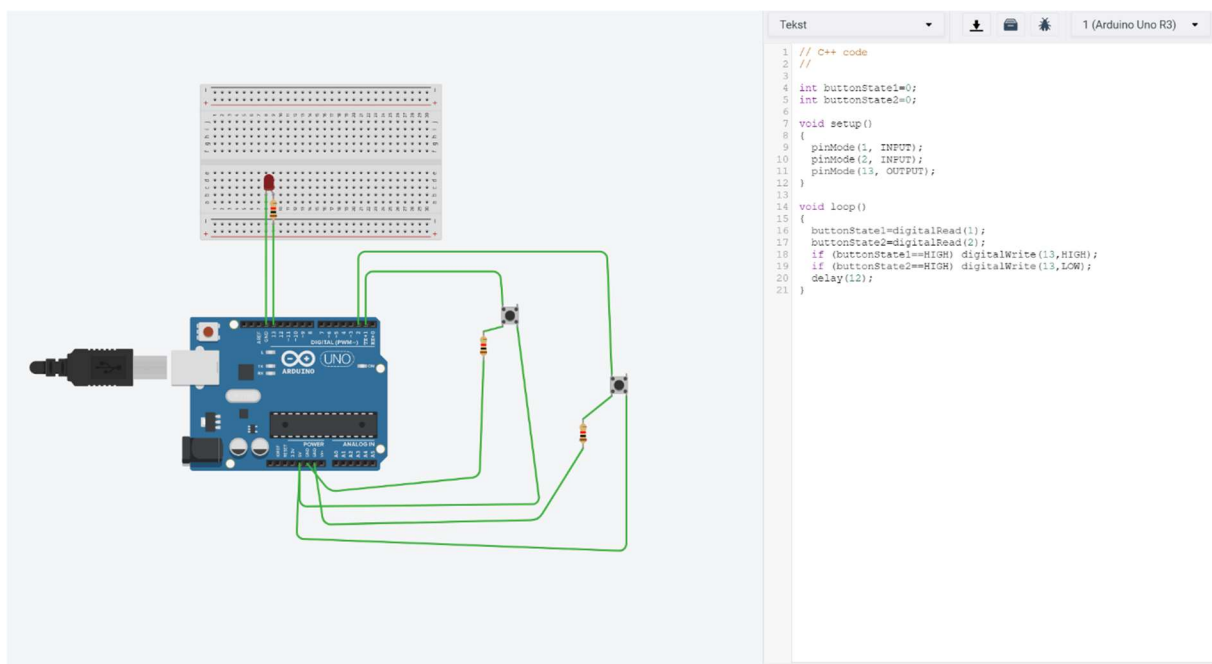
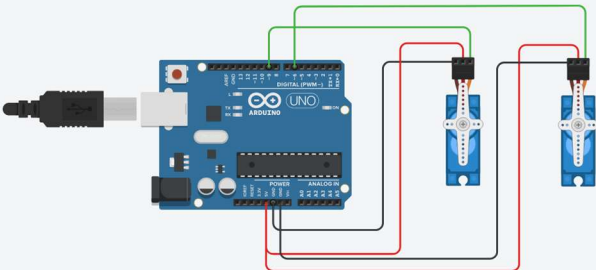


Link do githuba: repozytorium: <https://github.com/kuba14082001/mechatronika>

1. git push – wysyła commity do repozytorium zdalnego,
2. git commit – zbiera zmiany ze stage'a i tworzy z nich commit,
3. git pull – zachowuje zmiany w schowku,
4. git stash – zapamiętuje zmiany, które nie zostały zatwierdzone,
5. git fetch – pobiera dane z repozytorium zdalnego do repozytorium lokalnego,
6. git checkout – przechodzi między danymi gałęziami projektu oraz na tworzenie nowej gałęzi.

Symulacja z przyciskiem on/off

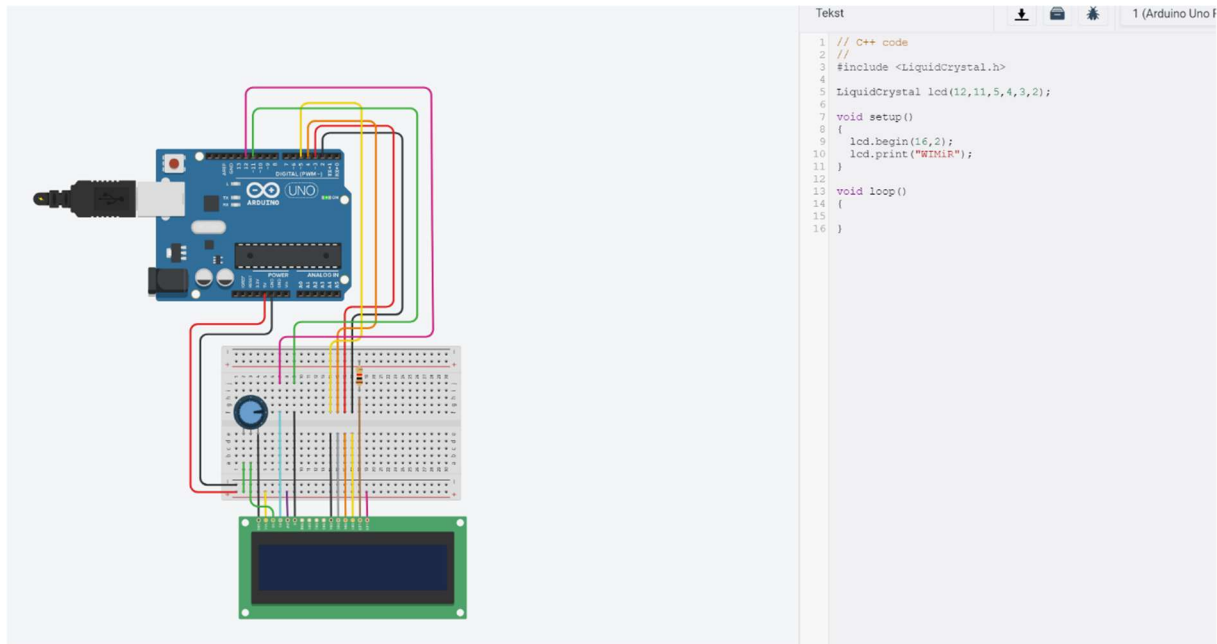




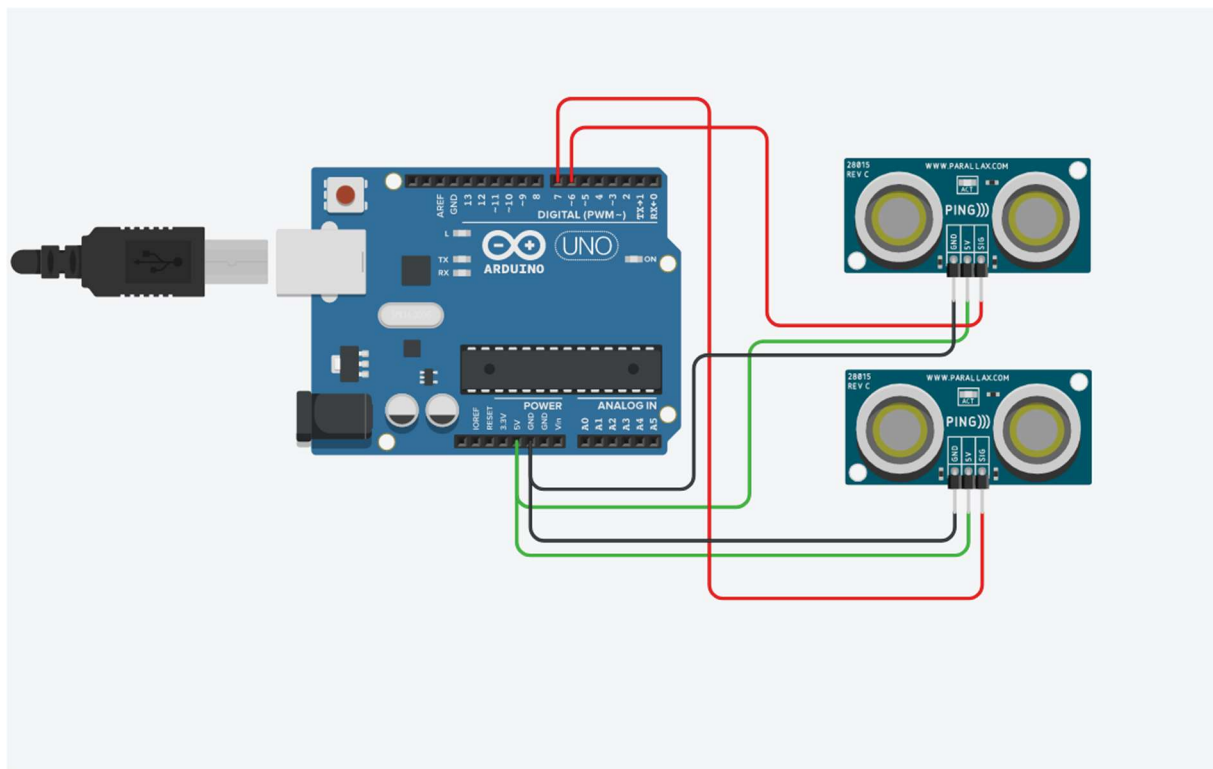
Tekst

1 (Arduino Uno R3)

```
1 // C++ code
2 //
3 #include <Servo.h>
4
5 int x = 0;
6 int y = 0;
7 Servo servo6;
8 Servo servo9;
9
10 void setup()
11 {
12   servo6.attach(6, 500, 2500);
13   servo9.attach(9, 500, 2500);
14 }
15
16 void loop()
17 {
18   for (x = 0; x <= 180; x += 1)
19   {
20     servo6.write(x);
21     delay(15);
22   }
23
24   for (y = 180; y >= 0; y -= 1)
25   {
26     servo9.write(y);
27     delay(15);
28   }
29 }
30
```



```
1  int inch=0;
2  int cm=0;
3
4  long readUD(int trigger, int echo)
5  {
6    pinMode(trigger, OUTPUT);
7    digitalWrite(trigger, LOW);
8    delayMicroseconds(2);
9
10    digitalWrite(trigger, HIGH);
11    delayMicroseconds(2);
12    digitalWrite(trigger, LOW);
13    pinMode(echo, INPUT);
14
15    return pulseIn(echo, HIGH);
16  }
17
18  void setup()
19  {
20    Serial.begin(9600);
21  }
22
23  void loop()
24  {
25    cm=0.01723*readUD(7,7);
26    cm=0.01723*readUD(6,6);
27    inch=cm/2.54;
28    Serial.println("inches: ");
29    Serial.println(inch);
30    Serial.println("cm: ");
31    Serial.println(cm);
32    delay(100);
33  }
```



## 6 Zadanie 6

### 6.1 Obsługa UART

Uniwersalny asynchroniczny nadajnik-odbiornik, czyli układ scalony służący do asynchronicznego przekazywania i

odbierania informacji poprzez port szeregowy.

Działanie UART opiera się na szeregowym wysłaniu ciągu bitów, które następnie składane są w pojedynczą informację.

Transmisja rozpoczyna się od bitu startu. Zawsze jest to bit będący logicznym zerem. Następnie, zależnie od konfiguracji,

następuje po sobie 7, 8 lub 9 bitów danych, które są wysyłaną informacją. Bit stopu to bit będący logiczną jedynką i mówi

on o końcu transmisji.

W Arduino interesują nas dwa piny:

1. Tx do wysyłania danych (pin 1 w Arduino),
2. Rx do odbierania danych (pin 0 w Arduino).

### 6.2 I2C

I

2C to szeregową, dwukierunkową magistralę służącą do przesyłania danych w urządzeniach elektronicznych.

Na samym początku wysyłany jest bit startu, który inicjuje transmisję. Charakteryzuje się on tym, że linia SDA

przechodzi z ze stanu wysokiego na niski, kiedy linia SCL jest dalej w stanie wysokim. Następnie wysyłanych jest 7 bitów

zawierających adres urządzenia oraz jeden bit R/W, który informuje, czy urządzenie typu master chce dokonać odczytu

(1) czy zapisu (0).

Następnie urządzenie slave powinno zwrócić nam 9. bit, którym jest Ack. Jeżeli wysłaliśmy poprawnie adres urządzenia

docelowego, to to urządzenie powinno automatycznie na linię SDA, podać stan niski co będzie oznaczało poprawny odbiór

adresu. Natomiast, jeżeli na linii SDA ciągle będzie stan wysoki to znaczy, że urządzenie slave nie rozpoznało adresu bądź

jest jakiś inny błąd związany z komunikacją pomiędzy urządzeniami i w tym momencie następuje przerwanie procedury.

Jeżeli nasz adres został poprawnie rozpoznany to jest podawany kolejny adres, tzw. adres wewnętrznego rejestru, który

nam zwraca daną wartość z układu.

W tym momencie jest podobna sytuacja, do tej z początku transmisji – czekamy na bit 9. Ack, czy urządzenie poprawnie

rozpoznało adres. Jeżeli podaliśmy zły adres to jest ta sama sytuacja co wcześniej, a jeżeli podaliśmy dobry adres to w

następnych 8 bitach, urządzenie slave wysyła do mastera dane, a następnie cały proces jest zamykany bitem Ack, oraz

bitem stopu, charakteryzującym się tym, że linia SDA przechodzi ze stanu niskiego w wysoki, kiedy linia SCL, jest w

stanie wysokim, czyli tak na prawdę jest to sytuacja odwrotna do bitu startu.

### 6.3 Mostek H

Mostek H jest układem elektrycznym umożliwiającym sterowanie kierunkiem działania prądu stałego. Składa się on

z czterech styków połączonych w kształt litery H. Odpowiednie zwieranie i rozwieranie styków pozwala zmienić kierunek

5

przepływu prądu przez silnik, a co za tym idzie zmienić kierunek obrotów silnika.

### 6.4 Czujniki analogowe a cyfrowe

Główną różnicą jest to, że czujnik cyfrowy generuje dyskretny (czyli nieciągły) sygnał wyjściowy, a czujnik analogowy

generuje sygnał ciągły.

Cyfrowe czujniki generują sygnały składające się z pojedynczych bitów. Gdy przetątnik jest ustawiony w pozycji

„włącz”, to domyka obwód, umożliwiając przepływ prądu, a gdy zostanie ustawiony w pozycji „wyłącz”, rozwiera obwód,

uniemożliwiając przepływ prądu. Sygnał ten może przyjąć jeden z dwóch stanów: 1 (włączony) albo 0 (wyłączony). Bity

mogą być łączone w ciągi, tworząc bajty składające się z  $n$  przekazywanych równolegle bitów.

Czujniki analogowe generują sygnał wyjściowy w postaci określonego napięcia. Nie mają wbudowanych układów

scalonych, więc konwersja sygnału musi przebiegać poza czujnikiem. Mają większą dokładność, gdyż nieprzekształcony

sygnał charakteryzuje się większą rozdzielczością. Analogowy sygnał jest jednak bardziej narażony na zakłócenia.