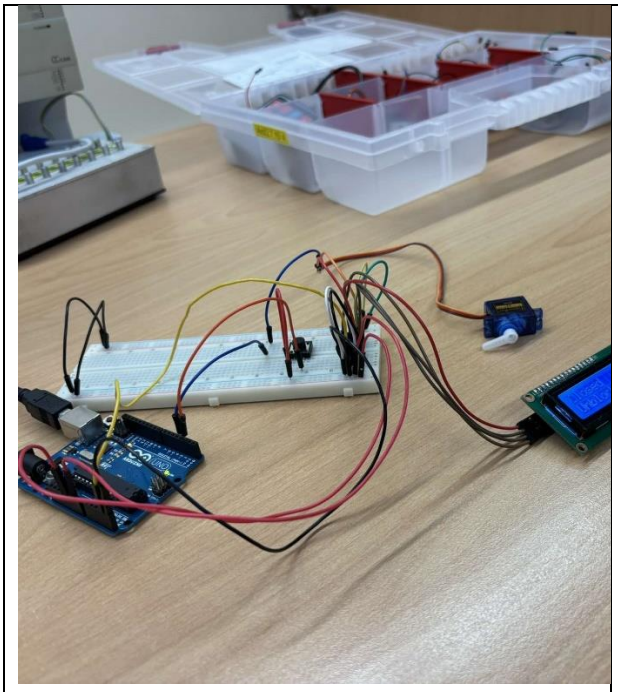
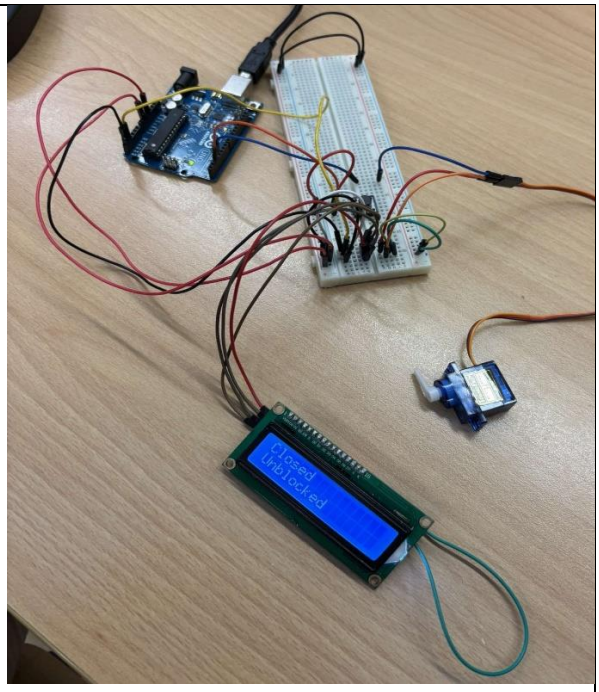
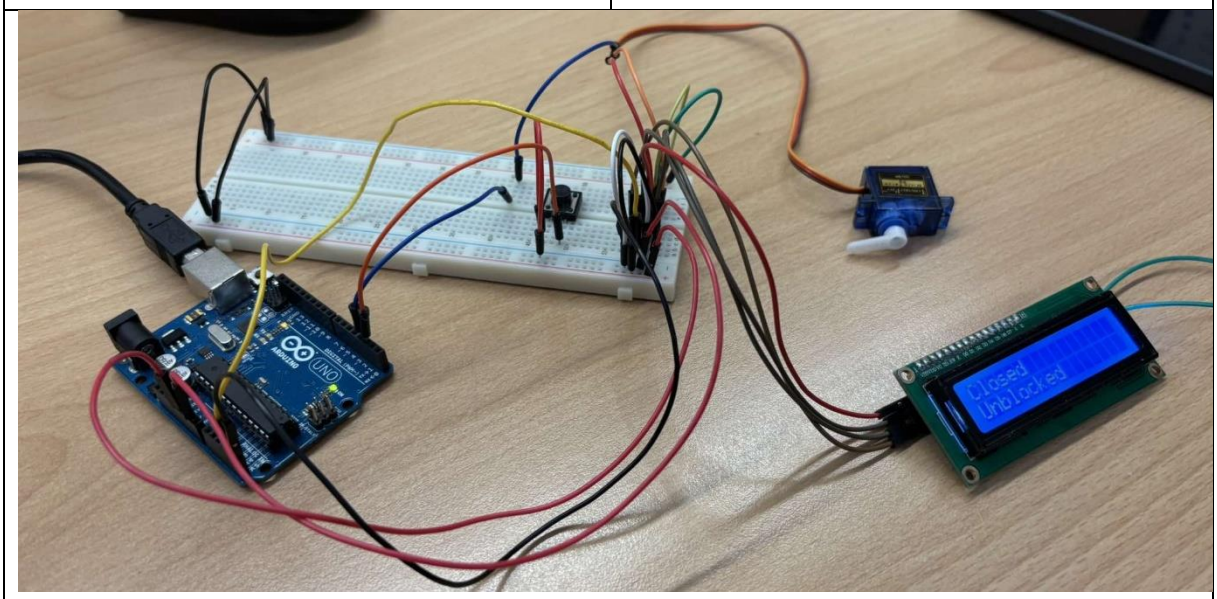
[Z3] Arduino UNO – elementy wykonawcze

Grupa laboratoryjna: L11 **Podgrupa: 4**	Paweł Kolec 155873	
	Adam Nowacki 155838	
	Prowadzący zajęcia:	**dr inż. Ariel Antonowicz**

ZDJĘCIA ZREALIZOWANEGO UKŁADU







KOD ZREALIZOWANEGO ALGORYTMU (wraz z komentarzami)

```
#include <OneWire.h>
#include <LiquidCrystal_I2C.h>
#include <Servo.h>

LiquidCrystal_I2C lcd(0x27, 16, 2); // Wyświetlacz LCD 16x2 na adresie 0x27
Servo doorServo;                    // Serwo do kontroli drzwi

String state = "off";               // Stan drzwi ("on" = otwarte, "off" = zamknięte)
int blockstate = 0;                 // Stan blokady (0 = zablokowane, 1 = odblokowane)
int doorPosition = 0;               // Pozycja drzwi (0 = zamknięte, 90 = otwarte)
bool doorOpeningRequested = false;  // Flaga otwierania drzwi

void setup() {
  pinMode(2, INPUT_PULLUP);
  lcd.init();                       // Inicjalizacja LCD
  lcd.backlight();                  // Włącza podświetlenie
  start();                          // Wyświetla stan początkowy
  delay(500);
  attachInterrupt(digitalPinToInterrupt(2), changeState, FALLING); // Przycisk zmiany stanu
  doorServo.attach(3);              // Serwo na pinie 3
  doorServo.write(0);               // Pozycja startowa serwa
  Serial.begin(9600);
}

// Wyświetla stan początkowy drzwi i blokady
void start() {
  lcd.setCursor(0, 0);
  lcd.print("Closed");
  lcd.setCursor(0, 1);
  lcd.print("Blocked");
}

// Zmienia stan blokady po naciśnięciu przycisku
void changeState() {
  if (state == "off") {             // Zmiana stanu na odblokowany
    blockstate = 1;
    state = "on";
  } else {                          // Zmiana stanu na zablokowany
    blockstate = 0;
    state = "off";
  }
}

void loop() {
  if (blockstate == 0) {            // Blokada aktywna
    lcd.setCursor(0, 1);
    lcd.print("Blocked  ");
    delay(1000);
  } else {                          // Blokada wyłączona
    lcd.setCursor(0, 1);
```

```
    lcd.print("Unblocked");
    delay(1000);

    while (Serial.available() > 0) {    // Odczytuje komendę przez Serial
      char cmd = Serial.read();
      if (blockstate == 0) {           // Ignoruje komendy w stanie blokady
        printErrorMessage("Blocked - Ignored");
        continue;
      }
      if (cmd == 'o' && doorPosition == 0 && !doorOpeningRequested) {
        doorOpeningRequested = true;    // Flaga otwierania
        openDoor();                     // Otwiera drzwi
      } else if (cmd == 'o' && doorPosition == 90 && !doorOpeningRequested) {
        printErrorMessage("Already open"); // Informacja: już otwarte
        lcd.clear();
        lcd.setCursor(0, 0);
        lcd.print("Open");
      } else if (cmd == 'c' && doorPosition == 90) {
        closeDoor();                    // Zamknięcie drzwi
      } else if (cmd == 'c' && doorPosition == 0) {
        printErrorMessage("Already closed"); // Informacja: już zamknięte
        lcd.clear();
        lcd.setCursor(0, 0);
        lcd.print("Closed");
      } else if (cmd != '\n' && cmd != '\r') {
        printErrorMessage("Wrong cmd"); // Komenda niepoprawna
      }
    }
  }
}

// Otwiera drzwi (serwo od 0 do 90 stopni)
void openDoor() {
  for (int pos = 0; pos <= 90; pos++) {
    doorServo.write(pos);
    delay(15);
  }
  lcd.clear();
  lcd.setCursor(0, 0);
  lcd.print("Open");
  doorPosition = 90;
  doorOpeningRequested = false;
}

// Zamyka drzwi (serwo od 90 do 0 stopni)
void closeDoor() {
  for (int pos = 90; pos >= 0; pos--) {
    doorServo.write(pos);
    delay(15);
  }
  lcd.clear();
  lcd.setCursor(0, 0);
```

```
  lcd.print("Closed");
  doorPosition = 0;
}

// Wyświetla komunikat błędu przez 3 sekundy
void printErrorMessage(const char* message) {
  lcd.clear();
  lcd.setCursor(0, 0);
  lcd.print(message);
  delay(3000);
}
```