Laboratorium z przedmiotu Systemy wbudowane (SW)

Karta projektu – zadanie 7

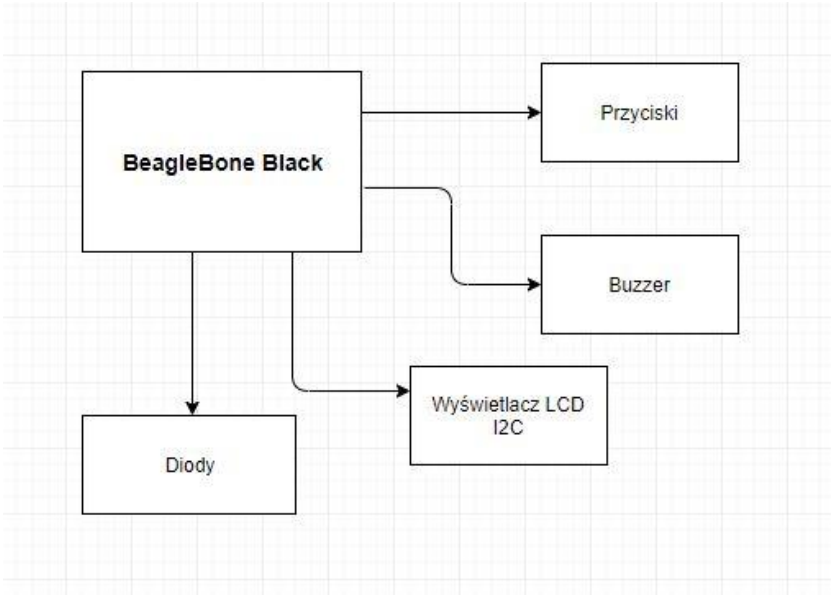
Nazwa projektu:

BeagleBone Black – maszyna grająca

Prowadzący:	*Autorzy (tylko nr indeksu):*	*Grupa dziekańska:*	*I4.2*
Mgr. Inż. Ariel Antonowicz	136774 136785	***Ocena:***	

Cel projektu: Celem naszego projektu jest zbudowanie i zaprogramowanie gry polegającej na odtworzeniu przedstawionej sekwencji wyświetlanej za pomocą diod LED. Po każdej poprawnie powtórzonej sekwencji do obecnie istniejącego ciągu zostaje dołożony kolejny element do odtworzenia. Odtwarzanie sekwencji będzie odbywało się za pomocą przycisków znajdujących się pod odpowiednimi diodami. Przy naciśnięciu przycisku będzie odtwarzany sygnał dźwiękowy. Na wyświetlaczu LCD będzie wyświetlany aktualny wynik podczas gry oraz aktualny najwyższy wynik ustanowiony podczas gry odczytywany z pliku.

Schemat:



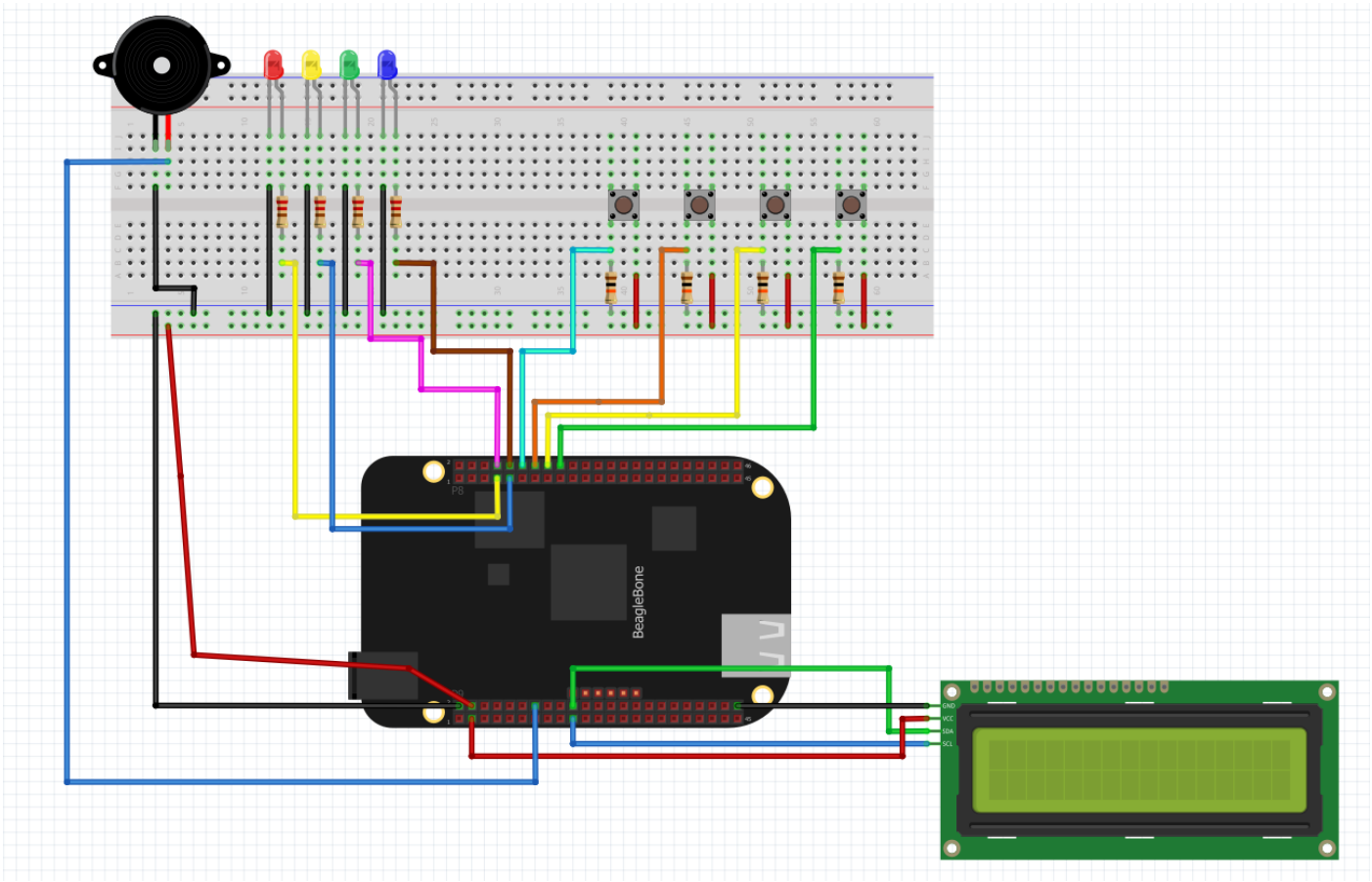
Wykorzystana platforma sprzętowa, czujniki pomiarowe, elementy wykonawcze:

- *BeagleBone Black*
- *Rezystory*
- *Diody LED*
- *Przyciski*
- *Buzzer*
- *Wyświetlacz LCD I2C*

2. Cel i zakres projektu

Celem naszego projektu było zbudowanie i zaprogramowanie gry polegającej na odtworzeniu przedstawionej sekwencji wyświetlanej za pomocą diod LED. Po każdej poprawnie powtórzonej sekwencji do obecnie istniejącego ciągu zostaje dołożony kolejny element do odtworzenia. Odtwarzanie sekwencji odbywa się za pomocą przycisków znajdujących się pod odpowiednimi diodami. Zarówno przy generacji kolejnej sekwencji jak i przy naciśnięciu przycisku odtwarzany jest sygnał dźwiękowy za pomocą buzzera. Na wyświetlaczu LCD jest wyświetlany aktualny wynik podczas gry oraz aktualny najwyższy wynik ustanowiony podczas gry odczytywany z pliku (highscore). Po zrobieniu niewłaściwego ruchu lub przekroczeniu aktualnego czasu przeznaczonego na daną rundę zostaje wyświetlona na wyświetlaczu informacja o zakończeniu gry i wyświetleniu wyniku końcowego gracza. W zakres naszej pracy wchodziło zbudowanie całego układu na platformie sprzętowej BeagleBone Black, całej gry typu memory za pomocą przycisków, diod LED, buzzera i wyświetlacza LCD.

3. Schemat połączeniowy wszystkich komponentów projektu wykonany w programie Fritzing



4. Projekt a realizacja

Do realizacji całego projektu podeszliśmy bardzo skrupulatnie i szczegółowo. Przed pierwszymi zajęciami związanymi z realizacją projektów przygotowaliśmy cały schemat połączeniowy wszystkich komponentów oraz napisaliśmy wstępny kod gry w języku Python z wykorzystaniem diod LED i przycisków, co ostatecznie na pierwszych zajęciach bardzo nam ułatwiło pracę i pozwoliło osiągnąć nasz pierwszy mały wyznaczony cel, którym była działająca gra w wersji podstawowej z diodami i przyciskami. Następnie na drugich zajęciach udało nam się najpierw poradzić z obsługą buzzera na platformie sprzętowej BeagleBone Black i następnie powtórzyliśmy tę sztukę z obsługą wyświetlacza LCD I2C, co pozwoliło na połączenie całego układu w jedną całość i zademonstrowanie jego działania na końcu drugich zajęć związanych z realizacją. Naszym założeniem od samego początku było przygotowanie się w domu, a realizację połączenia i debugowanie programu na rzeczywistym schemacie podczas zajęć. Cel uważamy za osiągnięty, gdyż znaczna część projektu została przygotowana w domu, a rzeczywiste odwzorowanie wykonaliśmy podczas zajęć. Naszym zdaniem projekt został w pełni zrealizowany z założeniami. W przyszłości można rozwinąć projekt o ładne zabudowanie całości. Rozumiemy przez to poprawienie wizualne naszego układu grającego. Jedną z dodatkowych opcji o jaką można by rozwinąć nasz układ jest przechowywanie większej ilości najlepszych wyników i wyświetlanie ich na wyświetlaczu podczas gry. Opcjonalnie można by było również dodać kolejne przyciski i diody do układu w celu podniesienia trudności rozgrywki.

5. Najważniejsze fragmenty kodu z komentarzem

```
button0 = "P8_12"
button1 = "P8_14"
button2 = "P8_16"
button3 = "P8_18"
led0 = "P8_7"
led1 = "P8_9"
led2 = "P8_8"
led3 = "P8_10"
BUZZER = "P9_14"
NOTES = [100, 400, 700, 999]
lcd = lcddriver.lcd()

game_array = []

current_round = 1
score = 0

GPIO.setup(button0, GPIO.IN)
GPIO.setup(button1, GPIO.IN)
GPIO.setup(button2, GPIO.IN)
GPIO.setup(button3, GPIO.IN)
GPIO.setup(led0, GPIO.OUT)
GPIO.setup(led1, GPIO.OUT)
GPIO.setup(led2, GPIO.OUT)
GPIO.setup(led3, GPIO.OUT)
```

Na wstępie programu inicjujemy zmienne wartościami początkowymi przypisanymi do odpowiednich portów na platformie sprzętowej. Za pomocą polecenia GPIO.setup ustawiamy przyciski jako wejście, oraz diody jako wyjście.

```
if __name__ == '__main__':
    file = open('Wyniki.txt', "r+")
    data = file.read().split()
    today = datetime.datetime.now()

    lcd.lcd_display_string("Highscore: " + str(data[0]), 3)  # 2 LINIA 3 zamienione z 2
    lcd.lcd_display_string("Current Score: 0", 1)  # 2 LINIA 3 zamienione z 2

    result = game()
    print("Your result: " + str(result))
    lcd.lcd_display_string("                ", 1)
    lcd.lcd_display_string("                ", 3)
    lcd.lcd_display_string("END GAME", 1)  # 2 LINIA 3 zamienione z 2
    lcd.lcd_display_string("Your result " + str(result), 3)  # 2 LINIA 3 zamienione z 2
    if result > int(data[0]):
        file.seek(0)
        file.write(str(result) + " " + str(today.strftime("%x")))
    file.close()
```

Następnie program rozpoczyna swoje właściwe działanie. Z pliku tekstowego zostaje odczytany aktualny najwyższy wynik i zostaje wypisany na wyświetlaczu LCD razem z aktualnym wynikiem gracza. Do zmiennej result zostaje przypisany ostateczny wynik z jakim gracz kończy grę, zwracany z funkcji game. Jeśli wynik gracza jest wyższy od aktualnego najwyższego, zostaje ustawiony jako nowy rekord i zapisujemy do pliku.

```
def game():
    while True:
        if GPIO.input(button0) or GPIO.input(button1) or GPIO.input(button2) or GPIO.input(button3):
            print("Let's start the game!")
            return game_loop()
```

Kolejno inicjowana jest rozgrywka po naciśnięciu dowolnego z czterech przycisków.

```
def game_loop():
    global current_round
    while True:
        generate_game_round()

        if not (player_turn()):
            return score
        else:
            current_round += 1
```

W pętli gry na początku danej rundy generowana jest sekwencja do powtórzenia za pomocą funkcji generate_game_round() oraz po wykonaniu ruchu gracza w zależności od poprawności zwracamy końcowy wynik lub przechodzimy do następnej rundy.

```
def generate_game_round():
    game_array.append(random.randint(0, 3))
    time.sleep(1.5)
    for x in range(0, current_round):
        blink(game_array[x], 0.5)
```

Do dotychczas istniejącej sekwencji zostaje dodany losowy ruch do powtórzenia. Pętla for odpowiedzialna jest za zapalenie odpowiedniej diody i generowanie sygnału dźwiękowego o określonej częstotliwości.

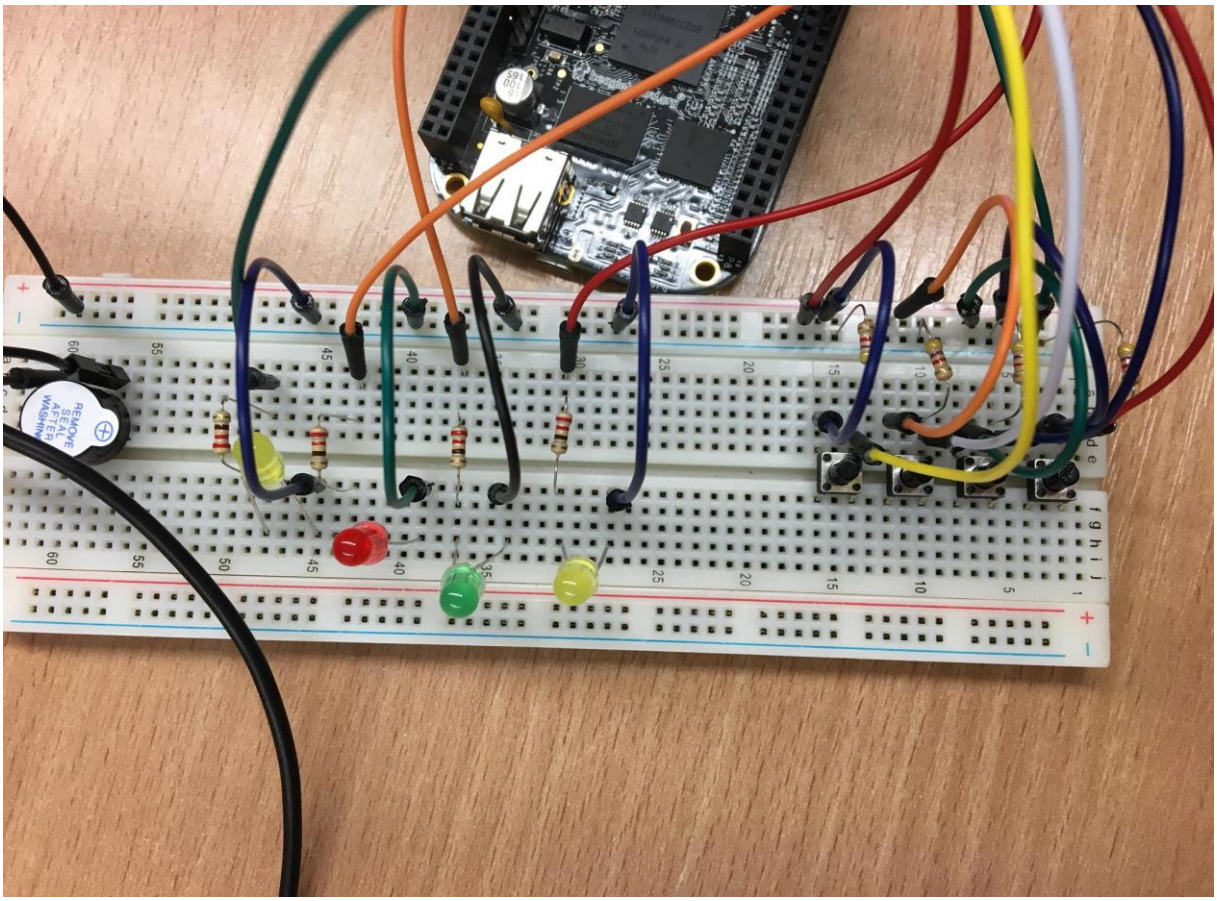
```
def blink(led, timer):
    if led == 0:
        GPIO.output(led0, GPIO.HIGH)
        PWM.start(BUZZER, 50, NOTES[0])
        time.sleep(timer)
        PWM.stop(BUZZER)
        GPIO.output(led0, GPIO.LOW)
    elif led == 1:
        GPIO.output(led1, GPIO.HIGH)
        PWM.start(BUZZER, 50, NOTES[1])
        time.sleep(timer)
        PWM.stop(BUZZER)
        GPIO.output(led1, GPIO.LOW)
    elif led == 2:
        GPIO.output(led2, GPIO.HIGH)
        PWM.start(BUZZER, 50, NOTES[2])
        time.sleep(timer)
        PWM.stop(BUZZER)
        GPIO.output(led2, GPIO.LOW)
    else:
        GPIO.output(led3, GPIO.HIGH)
        PWM.start(BUZZER, 50, NOTES[3])
        time.sleep(timer)
        PWM.stop(BUZZER)
        GPIO.output(led3, GPIO.LOW)
    time.sleep(timer)
```

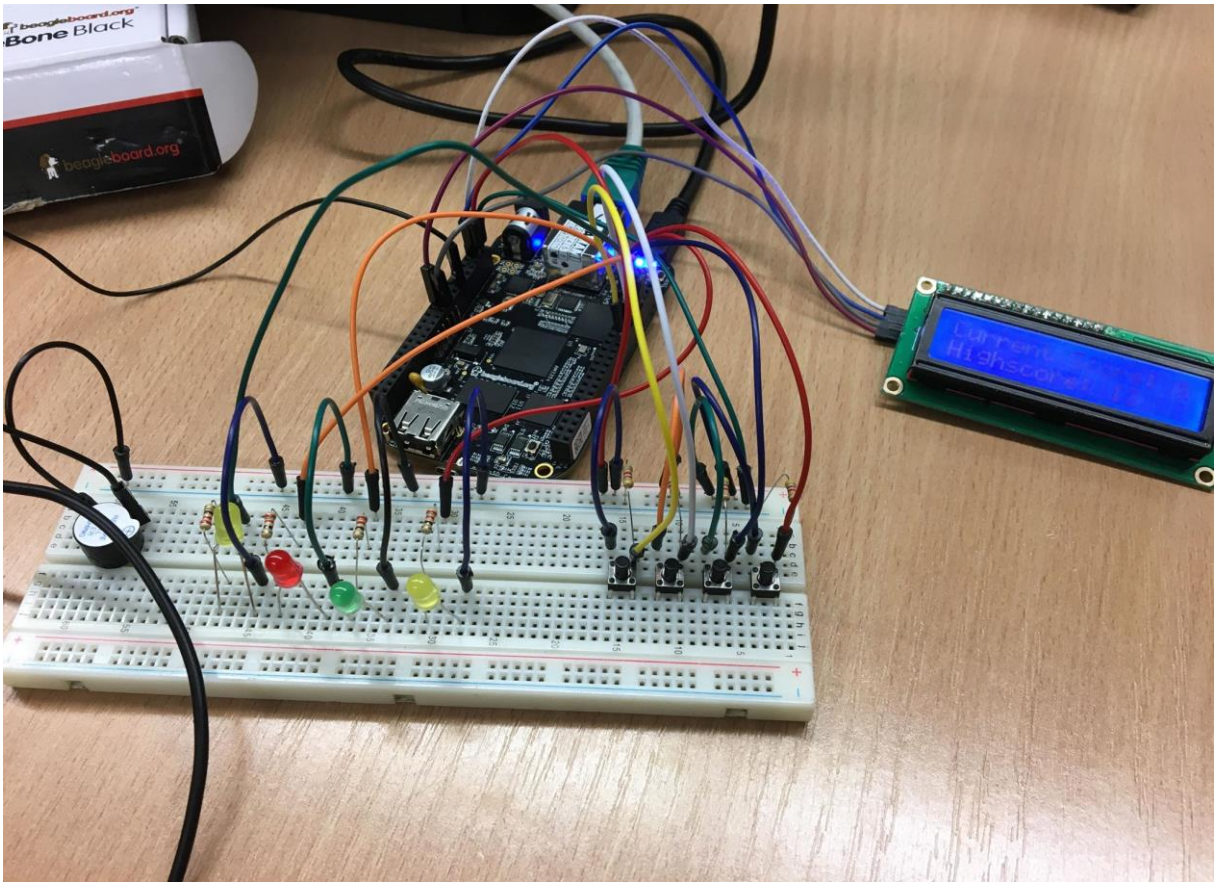
W zależności od diody, która ma zostać zapalona i sygnału, który ma zostać zagrany, wybieramy odpowiedni blok i wykonujemy określone działanie.

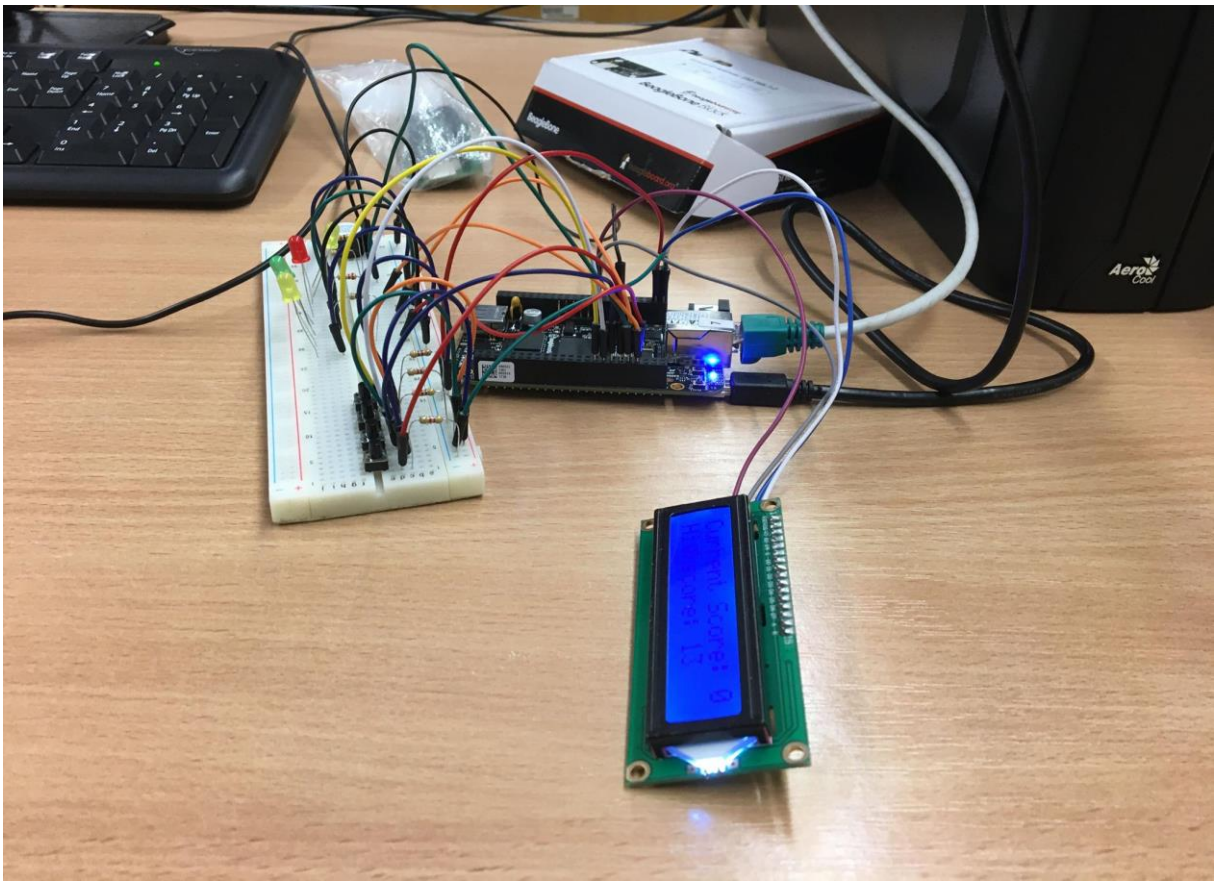
```
def player_turn():
    global score
    time_start = time.time()
    time_end = time.time()
    bad_move = False
    good_move = 0
    index = 0
    while ((time_end - time_start) < current_round + 3) and (not bad_move) and good_move < current_round:

        if GPIO.input(button0):
            if game_array[index] != 0:
                bad_move = True
            else:
                good_move += 1
            index += 1
            PWM.start(BUZZER, 50, NOTES[0])
            time.sleep(0.5)
            PWM.stop(BUZZER)

        if GPIO.input(button1):
            if game_array[index] != 1:
                bad_move = True
            else:
                good_move += 1
            index += 1
            PWM.start(BUZZER, 50, NOTES[1])
            time.sleep(0.5)
            PWM.stop(BUZZER)

        if GPIO.input(button2):
            if game_array[index] != 2:
                bad_move = True
            else:
                good_move += 1
            index += 1
            PWM.start(BUZZER, 50, NOTES[2])
            time.sleep(0.5)
            PWM.stop(BUZZER)

        if GPIO.input(button3):
            if game_array[index] != 3:
                bad_move = True
            else:
                good_move += 1
            index += 1
            PWM.start(BUZZER, 50, NOTES[3])
            time.sleep(0.5)
            PWM.stop(BUZZER)

        time_end = time.time()
    if good_move == current_round:
        score += 1
        lcd.lcd_display_string("Current Score: " + str(score), 1)
        return True
    else:
        return False
```

Powyższa funkcja odpowiada za sprawdzenie poprawności ruchów gracza. Sprawdza ona czy gracz nie zastanawia się zbyt długo (czas przewidziany na daną rundę to wartość aktualnej rundy powiększona o 3, czyli current_round + 3 sekund), jeśli tak, kończy grę i zwraca odpowiednią wartość. Nie jest to jednak jedyna możliwość zakończenia gry. Gracz, który próbuje odtworzyć sekwencje, jest sprawdzany przez warunki i w przypadku wystąpienia pomyłki gra jest kończona i zwracany zostaje odpowiednia wartość świadcząca o pomyłce. W przypadku poprawnego zagrania, zwracamy wartość sygnalizującą przejście do generowania kolejnej rundy. Podczas gry, gracz używający przycisku usłyszy dźwięk generowany przez buzzer.

6. Zdjęcia fizycznego urządzenia/połączeń







7. Podsumowanie i wnioski

Reasumując projekt uznajemy za bardzo rozwijający. Wszystkie wstępne założenia, jakie sobie ustaliliśmy zostały zrealizowane. Nie doświadczyliśmy większych problemów z realizacją układu dzięki wcześniejszemu przygotowaniu z wyjątkiem użytkowania wyświetlacza LCD. Byliśmy przekonani, że jego podłączenie i obsługa będzie równe łatwe co na platformie Arduino. Jednak po dłuższym zastanowieniu i z tym udało się uporać. Uważamy, że zdobyte doświadczenie przy realizacji tego projektu jest godne odnotowania i na pewno zaprocentuje to w przyszłości przy podobnych projektach.