

# Raport z projektu

Temat projektu: Testy gry na refleks

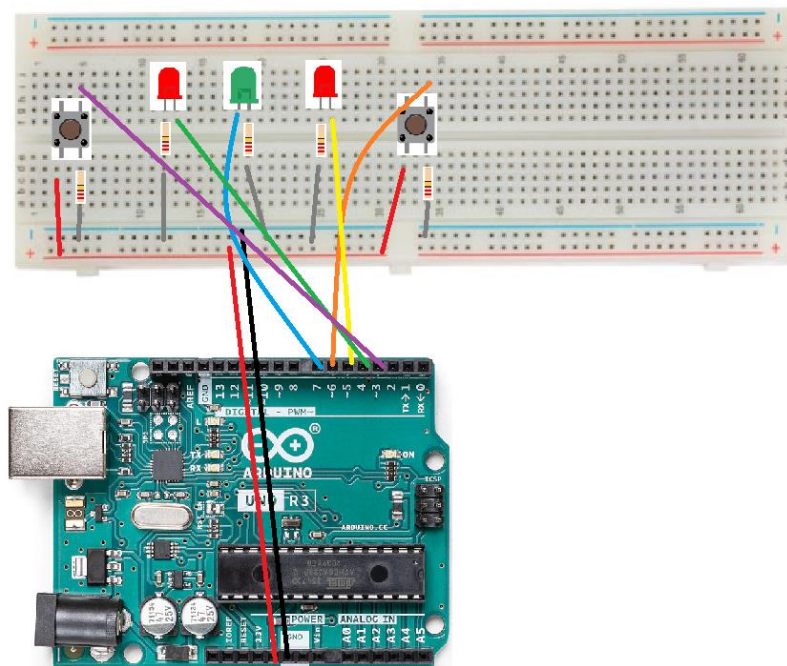
Przedmiot: Testowanie i Niezawodność

Wykonały: Agata Krześniak

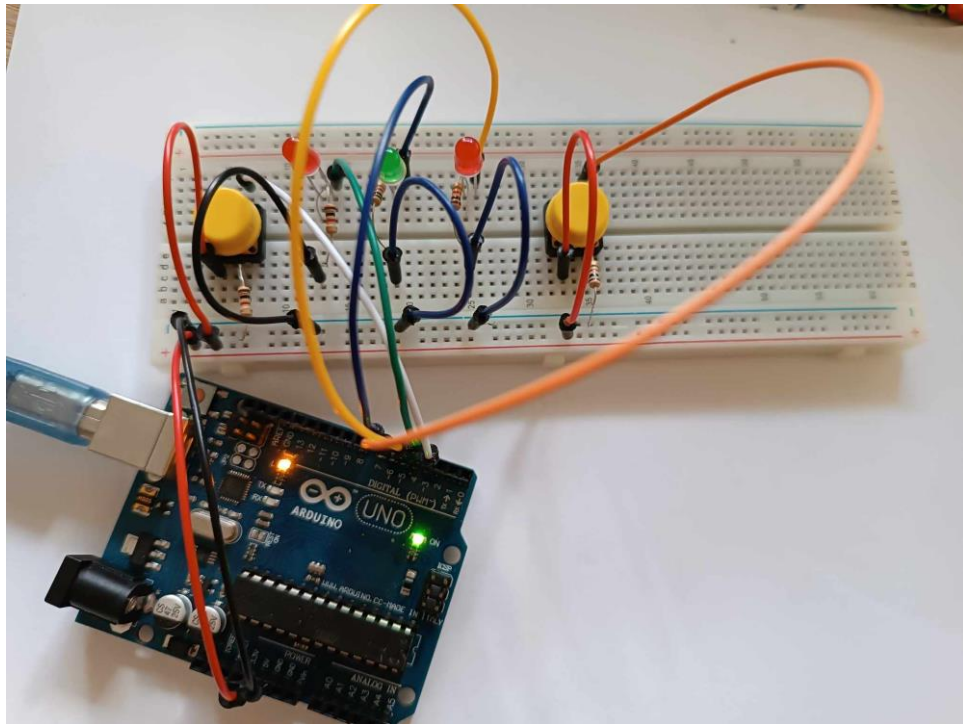
Klaudia Litwin

## 1. Opis projektu

Jako projekt, który będzie testowany stworzyliśmy mini grę na refleks. Do stworzenia tej gry wykorzystaliśmy arduino uno, do którego zostały podłączone dwa przyciski i trzy diody. W celu zapewnienia prawidłowego działania przycisków i diod wykorzystano również trzy rezystory o rezystancji 220Ohm. Schemat połączeń został zaprezentowany na Rys. 1. Z kolei na Rys.2 zostało pokazane zdjęcie z rzeczywistym wyglądem naszego schematu.



Rys.1 Schemat połączenia.



Rys.2 Zdjęcie zrealizowanego schematu.

Płytkę arduino została zaprogramowana w taki sposób, by po wgraniu na nią kodu zapalała się zielona dioda. Następnie wybierana jest losowa liczba z zakresu 2 do 7 sekund i po takim czasie zielona dioda gaśnie. W czasie po zgaśnięciu oczekujemy na kliknięcie jednego z przycisków. Przy tym, który zostanie wciśnięty pierwszy zostanie zapalona czerwona dioda. Gdy czerwona dioda gaśnie ponownie zapala się dioda zielona i tym samym sekwencja zaczyna się od nowa.

## 2. Zasady gry

Poniżej przedstawione zostały zasady gry.

1. Rozpoczęcie rozgrywki:
  - a. Gra rozpoczyna się po zapaleniu zielonej diody.
  - b. Dioda zielona będzie świecić przez losowy czas, w przedziale od 2 do 7 sekund.
  - c. Gracze muszą być gotowi do reakcji, ponieważ rozgrywka rozpocznie się po zgaśnięciu zielonej diody.
2. Rywalizacja o szybkość reakcji:
  - a. Po zgaśnięciu diody zielonej każdy z graczy naciska na swój przycisk.
  - b. Rozgrywkę wygrywa ten z graczy, który okazał się zręczniejszy i przycisnął przycisk jako pierwszy.

- c. Wygrana gracza sygnalizowana jest poprzez zaświecenie się czerwonej diody znajdującej się przy jego przycisku. Dodatkowo wiadomość o tym który gracz wygrał wysyłana jest poprzez port szeregowy.

Uwaga! Gracz nie może przytrzymywać wciśniętego przycisku.

### 3. Testy

Testy do naszego projektu zostały przeprowadzone na kilka sposobów.

- a. Jednym ze sposobów było dodanie do kodu w języku arduino możliwości zasymulowania wciśnięcia przycisku poprzez wpisanie odpowiedniej litery z poziomu Monitora portu szeregowego tak jak zostało to zaprezentowane to na Rys. 3.

```
// Odczytaj komendę z portu szeregowego
while (Serial.available()) {
    char command = Serial.read(); // Odczytaj pojedynczy znak z portu szeregowego

    // Jeżeli otrzymano komendę "L" (zmiana stanu przycisku L na HIGH)
    if (command == 'L' && stanGreen == LOW) {
        stanLeft = !stanLeft;
    }
    // Jeżeli otrzymano komendę "R" (zmiana stanu przycisku R na HIGH)
    else if (command == 'R' && stanGreen == LOW) {
        stanRight = !stanRight;
    }
}
```

Rys. 3 Kod umożliwiający zasymulowanie wciśnięcia przycisku.

Dodanie takiej możliwości pozwoliło na lepszą obserwację, co się dzieje, gdy przycisk zostaje wciśnięty i przytrzymany. W takim przypadku zaobserwowaliśmy, że o ile w kodzie nie zostaje dodane wymuszenie zmiany stanu przycisku to pozostaje on cały czas w tym samym stanie. Z poziomu monitora widzimy, że działanie programu w tym przypadku jest następujące. Gdy przycisk zostaje wciśnięty w trakcie, gdy dioda zielona jest zaświecona, dioda czerwona nie zaświeci się, dopóki zielona nie zgaśnie. Jednak, gdy zielona gaśnie czerwona dioda zapala się i pojawia się informacja o wciśnięciu przycisku i zapaleniu się tej diody, mimo że pierwsze wywołanie stanu nastąpiło przed zgaśnięciem zielonej diody. W przypadku zasymulowania wciśnięcia i przytrzymania obu przycisków w trakcie świecenia się zielonej diody po zgaśnięciu przyjmowane jest, że wygrał ten, który wciśnięty został jako pierwszy. Powyższy przypadek został też przetestowany w sposób manualny na fizycznym modelu. W takim przypadku przyciśnięcie i przytrzymanie przycisku powodują problemy w działaniu projektu.

- b. Przetestowanie połączenia z poziomym terminalem Python, w celu umożliwienia napisania dalszych testów.

W celu przetestowania działania gry, w pierwszej kolejności został napisany skrypt w języku Python, który odczytuje oraz wysyła dane przez port szeregowy do płytki. W celu osiągnięcia tej komunikacji wykorzystana została biblioteka serial. Należało zainicjalizować połączenie szeregowo – ustawić odpowiedni port oraz szybkość transmisji – 9600. Następnie zostały napisane funkcje, które umożliwiają odczyt danych oraz ich wysyłanie. Na Rys. 4 znajdują się dane które zostały odebrane przez napisany skrypt w pythonie. Skrypt ten pozwolił przede wszystkim sprawdzić poprawność połączenia z portem.

```
Odczytano dane: Zaswieca sie zielona
Odczytano dane: Gasnie zielona
Odczytano dane: Wygral gracz 2
Odczytano dane: Zaswieca sie czerwona 2
Odczytano dane: Gasnie czerwona 2
Odczytano dane: Zaswieca sie zielona
Odczytano dane: Gasnie zielona
```

Rys.4 Wyniki testu połączenia z poziomu terminala Python

c. Testy przy użyciu biblioteki pytest:

Pierwszym wykonanym w ten sposób testem był test weryfikujący czy dioda gaśnie w ciągu maksymalnie 7 sekund od zaświecenia się. Test ten został zawarty w pliku “test\_gaszenia\_diody.py” i weryfikuje na podstawie komunikatów przesyłanych wraz z zaświeceniem się i zgaśnięciem diody jaki czas minął do zgaszenia diody. Jeżeli czas ten wyniesie mniej niż 7 sekund test przejdzie prawidłowo, tak jak zostało to zaprezentowane na Rys.5.

```
===== test session starts =====
platform win32 -- python: 3.9.7, pytest-7.4.0, pluggy-1.2.0
rootdir: c:\Users\kmalia\libwin\desktop\studa\magisterskie\TDA\projekt
collected 1 item

test_gaszenia_diody.py . (100%)

1 passed in 0.44s
```

Rys.5 Prawidłowe przejście testu sprawdzającego czas zgaszenia diody

Aby zweryfikować prawidłowość testu zmieniliśmy wartości, z których losowany jest czas oczekiwania na zgaśnięcie diody tak by losowana wartość przekraczała 7 sekund. W tym przypadku, zgodnie z oczekiwaniami pojawił się błąd, który został zaprezentowany na Rys.6.

```
collected 1 item
test.py F [100%]

===== FAILURES =====
test_arduino

arduino_serial = Serial(id=0x2229b4052b0, open=True)(port='COM3', baudrate=9600, bytesize=8, parity='N', stopbits=1, timeout=None, xonxoff=False, rtscts=False, dsrdtr=False)

def test_arduino(arduino_serial):
    # Wysłanie komunikatu "Zaswieca sie zielona" do Arduino
    arduino_serial.write(b'Zaswieca sie zielona\n')

    # Oczekiwanie na odpowiedź od Arduino przez 7 sekund
    start_time = time.time()
    response = ""
    while time.time() - start_time < 7:
        if arduino_serial.in_waiting > 0:
            response += arduino_serial.readline().decode().strip()
            if "gasnie zielona" in response:
                break

    # Sprawdzenie wyniku testu
    > assert "gasnie zielona" in response, "Błąd - nie otrzymano oczekiwanych danych"
E   AssertionError: Błąd - nie otrzymano oczekiwanych danych
E   assert 'gasnie zielona' in 'Zaswieca sie zielona'

test.py:22: AssertionError

===== short test summary info =====
FAILED test.py::test_arduino - AssertionError: Błąd - nie otrzymano oczekiwanych danych
1 failed in 0.20s
```

Rys.6 Nieprawidłowe przejście testu sprawdzającego czas zgaszenia diody

Wykonany został też analogiczny test weryfikujący czy dioda nie zgasła w czasie krótszym niż dwie sekundy. Test ten został przestany w pliku o nazwie test\_gaszenie\_ponizej\_2s.py. Rezultat testu przy poprawnym działaniu został zamieszczony na Rys.7.

```
platform win32 -- python 3.9.7, pytest-7.4.0, pluggy-1.2.0
rootdir: C:\Users\Klaudia\lib\win\desktop\studies\magisterskie\TIN\projekt
collected 1 item

test_gaszenie_ponizej_2s.py . [100%]

1 passed in 2.97s
```

Rys.7 Prawidłowe przejście testu sprawdzającego czy czas zgaszenia diody jest dłuższy niż 2 sekundy.

W tym przypadku również sprawdzono reakcje na błędne działanie programu i zgodnie z oczekiwaniami pojawił się komunikat o błędzie jak na Rys. 8.

```
collected 1 item
test.py F [100%]

===== FAILURES =====
test_arduino

arduino_serial = Serial(id=0x2c7fb1853d0, open=True)(port='COM3', baudrate=9600, bytesize=8, parity='N', stopbits=1, timeout=2, xonxoff=False, rtscts=False, dsrdtr=False)

def test_arduino(arduino_serial):
    # Wysłanie komunikatu "Zaswieca sie zielona" do Arduino
    arduino_serial.write(b'Zaswieca sie zielona\n')

    # Oczekiwanie na odpowiedź od Arduino przez 2 sekundy
    start_time = time.time()
    response = ""
    while time.time() - start_time < 2:
        if arduino_serial.in_waiting > 0:
            response += arduino_serial.readline().decode().strip()

    # Sprawdzenie wyniku testu
    > assert "gasnie zielona" not in response, "Błąd - otrzymano informacje o gasnieciu zielonej"
E   AssertionError: Błąd - otrzymano informacje o gasnieciu zielonej
E   assert 'gasnie zielona' not in 'Zaswieca si...nie zielona'
E   'gasnie zielona' is contained here:
E   Zaswieca sie zielonagasnie zielona

test.py:24: AssertionError

===== short test summary info =====
FAILED test.py::test_arduino - AssertionError: Błąd - otrzymano informacje o gasnieciu zielonej
1 failed in 2.17s
```

Rys.8 Nieprawidłowe przejście testu sprawdzającego czy czas zgaszenia diody jest dłuższy niż 2 sekundy.

Kolejny test miał za zadanie weryfikacji, czy po zasymulowaniu kliknięcia przycisku lewego, rozgrywkę wygra gracz 1. Test znajduje się w pliku test\_przycisk\_lewy.py. Zasymulowanie działania przycisku polega na wpisaniu komendy "L" w port szeregowy, dzięki temu stan przycisku lewego zmienia stan z niskiego na wysoki. Oczekiwany rezultat takiego działania jest otrzymanie kolejno danych: " wygrał

gracz 1 ", "zaswieca sie czerwona 1", "gasnie czerwona 1". Otrzymano oczekiwany rezultat, wynik został przedstawiony poniżej, na Rys.9.

```
PS C:\Users\Klaudia.Litwin\Desktop\studia\magisterskie\TIN\projekt> pytest test_przycisk_lewy.py
===== test session starts =====
platform win32 -- Python 3.9.7, pytest-7.4.0, pluggy-1.2.0
rootdir: C:\Users\Klaudia.Litwin\Desktop\studia\magisterskie\TIN\projekt
collected 1 item

test_przycisk_lewy.py . [100%]

===== 1 passed in 0.40s =====
```

Rys.9 Test przycisku lewego – otrzymano oczekiwany rezultat

Aby zweryfikować poprawność działania testu w kodzie arduino dla wygranej gracza 1 zmieniono zapalanie się diody 1 na diodę 2. Na Rys.10 widać informacje – operacja się nie powiodła otrzymano komunikat " Bład – nie otrzymano oczekiwanych danych – zaswieca się czerwona 1".

```
collected 1 item

test3.py F [100%]

===== FAILURES =====
test_game_results

ser = Serial(id=0cdf5752150a0, open=True)(port='COM3', baudrate=9600, bytesize=8, parity='N', stopbits=1, timeout=None, xonoff=False, rtscts=False, dsrdtr=False)

def test_game_results(ser):
    expected_data = "wygral gracz 1"
    expected_data1 = "zaswieca sie czerwona 1"
    expected_data2 = "gasnie czerwona 1"

    received_data = ''
    for _ in range(5):
        data = read_data(ser)
        received_data = received_data + data

    assert expected_data in received_data, "Bład - nie otrzymano oczekiwanych danych - wygral gracz 1"
> assert expected_data1 in received_data, "Bład - nie otrzymano oczekiwanych danych - zaswieca sie czerwona 1"
E AssertionError: Bład - nie otrzymano oczekiwanych danych - zaswieca sie czerwona 1
E assert "zaswieca sie czerwona 1" in "zaswieca sie zielonawygral gracz 1zaswieca sie czerwona 2gasnie czerwona 1"

test3.py:43: AssertionError
----- Captured stdout call -----
zaswieca sie zielona
gasnie zielona
wygral gracz 1
zaswieca sie czerwona 2
gasnie czerwona 1

===== short test summary info =====
FAILED test3.py::test_game_results - AssertionError: Bład - nie otrzymano oczekiwanych danych - zaswieca sie czerwona 1
===== 1 failed in 0.19s =====
```

Rys.10 Test przycisku lewego – otrzymano błędne dane

Dla przetestowania działania przycisku prawego powstał analogiczny test, gdzie na port szeregowy wpisywana jest komenda "R". Test zawiera się w pliku test\_przycisk\_prawy.py. Oczekiwany rezultat takiego działania jest otrzymanie kolejno danych: "wygral gracz 2", "zaswieca sie czerwona 2", "gasnie czerwona 2".

```
platform win32 -- Python 3.9.7, pytest-7.4.0, pluggy-1.2.0
rootdir: C:\Users\Klaudia.Litwin\Desktop\studia\magisterskie\TIN\projekt
collected 1 item

test_przycisk_prawy.py . [100%]

===== 1 passed in 0.07s =====
```

Rys.11 Test przycisku prawego – otrzymano oczekiwany rezultat

#### 4. Znalezione błędy i ich możliwe rozwiązania

Znaleziony błąd	Propozycje rozwiązania
Wciśnięcie i przytrzymanie przycisku	<ul style="list-style-type: none"><li>• Komunikat o błędzie i zatrzymanie gry, gdy przycisk jest zbyt długo w stanie wysokim</li><li>• Wymuszenie zmiany stanu przycisku</li><li>• Automatyczna wygrana drugiego gracza, gdy zarejestrowane zostało wciśnięcie przycisku w trakcie świecenia zielonej diody</li></ul>

#### 5. Podsumowanie

Powyżej wykonane testy dotyczyły głównie sprawdzenia odpowiednich reakcji zapalania się diod. Powyższe testy można również rozwinąć o inne przypadki testowe.