Raport z projektu

Temat projektu: Testy gry na refleks

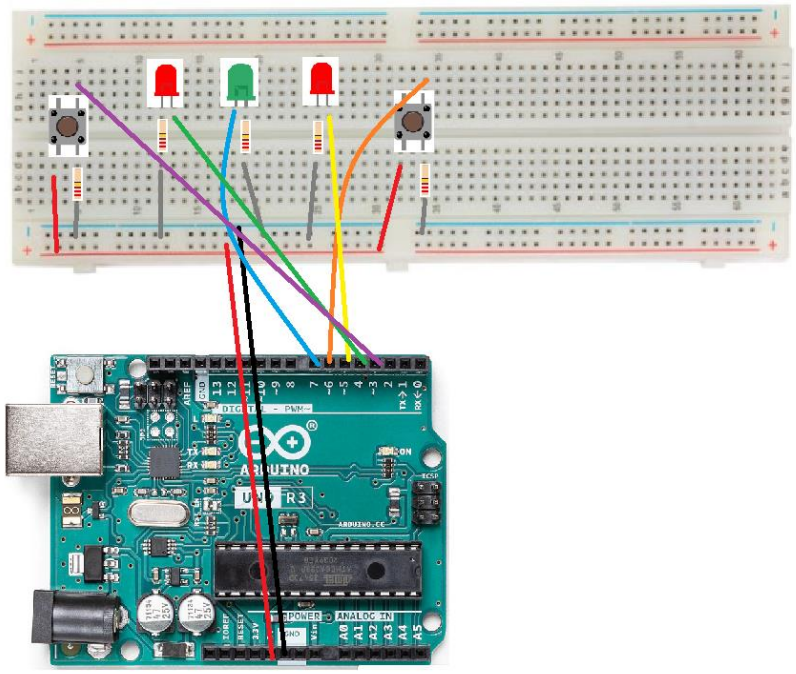
Przedmiot: Testowanie i Niezawodność

Wykonały: Agata Krześniak

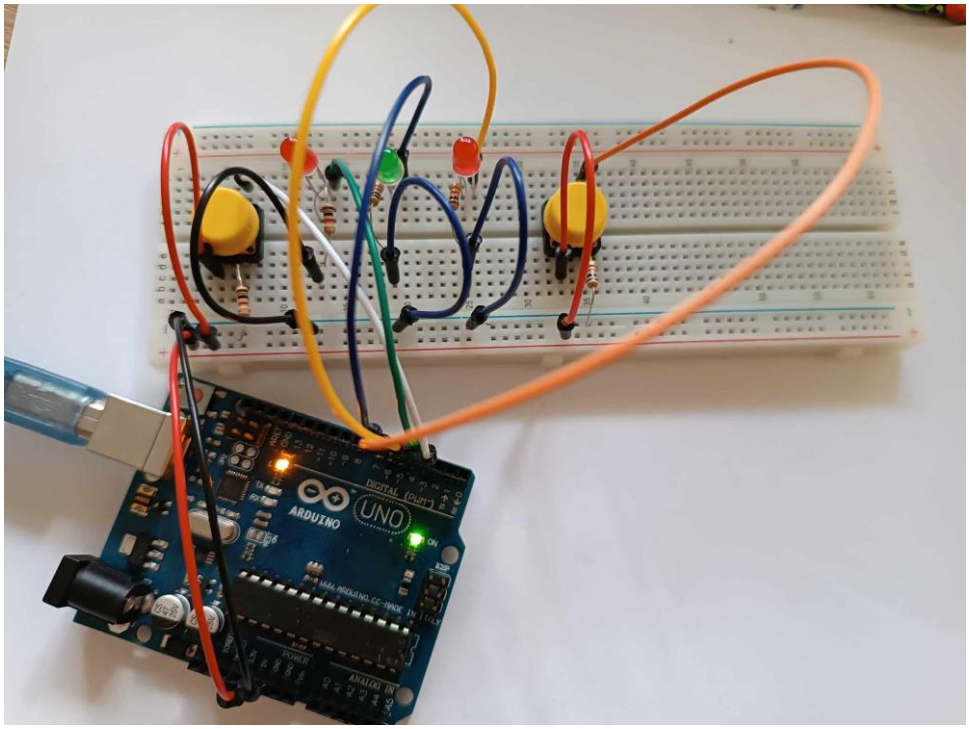
Klaudia Litwin

1. Opis projektu

Jako projekt, który będzie testowany stworzyłyśmy mini grę na refleks. Do stworzenia tej gry wykorzystałyśmy 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.

Płytka 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.

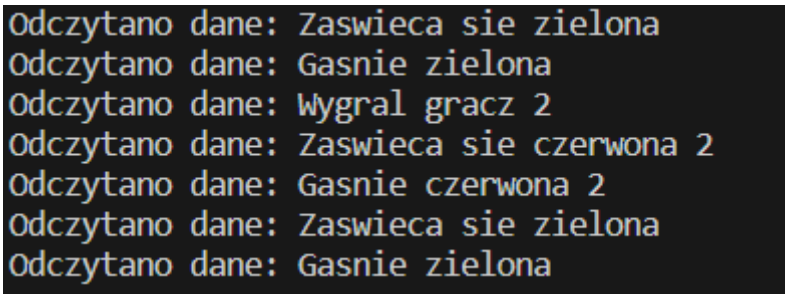
```
// Odczytaj komendę z portu szeregowego
  while (Serial.available()) {
    char command = Serial.read();  // Odczytaj pojedynczy znak z portu szeregowego

    // 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 zaobserwowałyśmy, 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 poziomu terminala Python, w celu umożliwienia napisania dalszych testów.

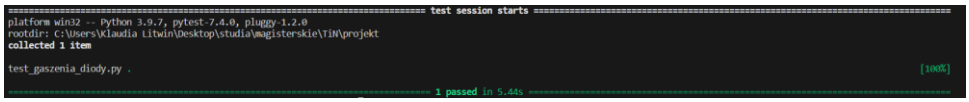
W celu przetestowania działania gry, w pierwszej kolejności został napisany skrypt w języku Python, który odczytuje oraz wysyła dane przez port szeregowy do płytki. W celu osiągnięcia tej komunikacji wykorzystana została biblioteka serial. Należało zainicjalizować połączenie szeregowe – ustawić odpowiedni 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.



Rys.4 Wyniki testu połączenia z poziomu terminala Python

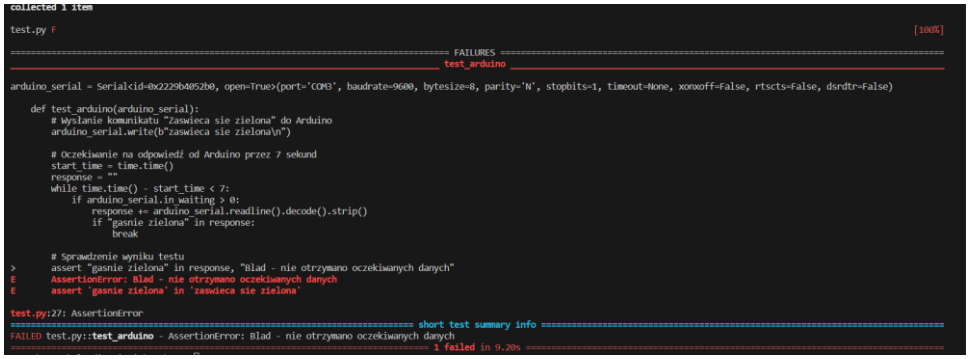
c. Testy przy użyciu biblioteki pytest:

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.



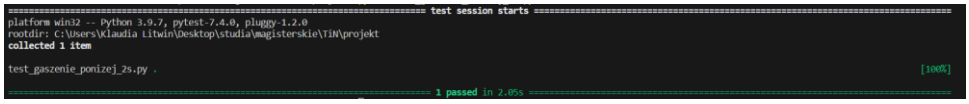
Rys.5 Prawidłowe przejście testu sprawdzającego czas zgaszenia diody

Aby zweryfikować prawidłowość testu zmieniłyśmy 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.



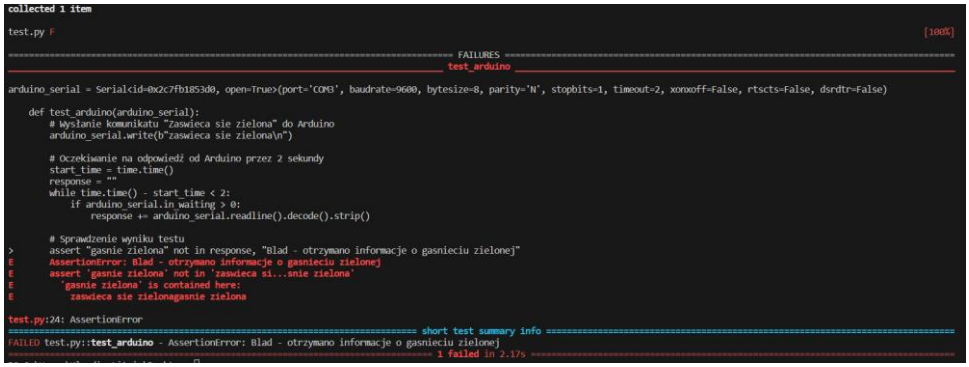
Rys.6 Nieprawidłowe przejście testu sprawdzającego czas zgaszenia diody

Wykonany został też analogiczny test weryfikujący czy dioda nie zgasła w czasie krótszym niż dwie sekundy. Test ten został przesłany w pliku o nazwie test_gaszenie_ponizej_2s.py. Rezultat testu przy poprawnym działaniu został zamieszczony na Rys.7.



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

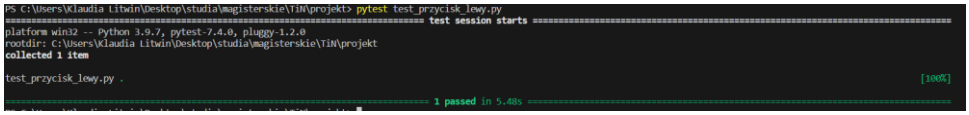
W tym przypadku również sprawdzono reakcje na błędne działanie programu i zgodnie z oczekiwaniami pojawił się komunikat o błędzie jak na Rys. 8.



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

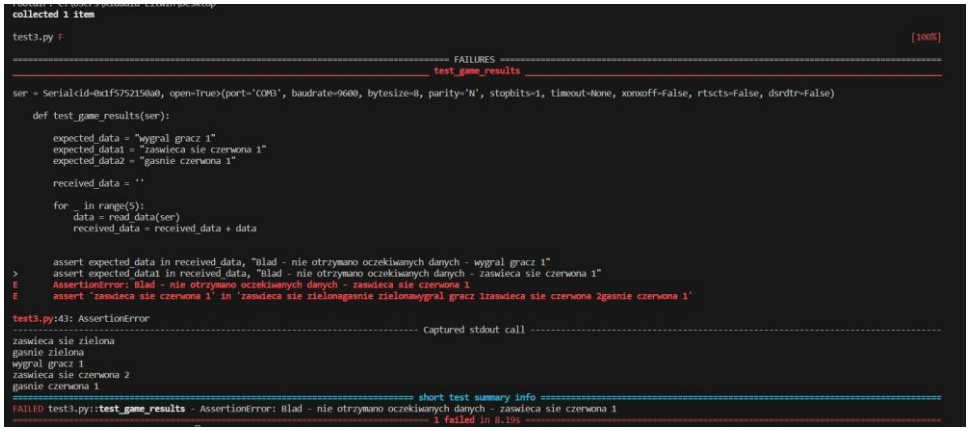
Kolejny test miał za zadanie weryfikacji, czy po zasymulowaniu kliknięcia przycisku lewego, rozgrywkę wygra gracz 1. Test znajduje się w pliku test_przycisk_lewy.py. Zasymulowanie działania przycisku polega na wpisaniu komendy “L” w port szeregowy, dzięki temu stan przycisku lewego zmienia stan z niskiego na wysoki. Oczekiwanym rezultatem takiego działania jest otrzymanie kolejno danych: “ wygral

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



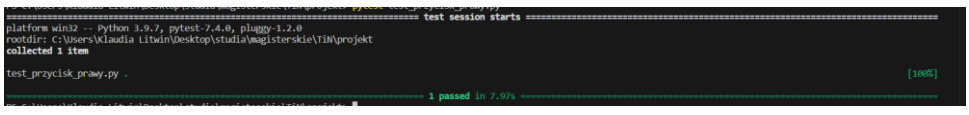
Rys.9 Test przycisku lewego – otrzymano oczekiwany rezultat

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 “ Blad – nie otrzymano oczekiwanych danych – zaswieca się czerwona 1”.



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. Oczekiwanym rezultatem takiego działania jest otrzymanie kolejno danych: “wygral gracz 2“, “zaswieca sie czerwona 2”, “gasnie czerwona 2”.



Rys.11 Test przycisku prawego – otrzymano oczekiwany rezultat

4. Znalezione błędy i ich możliwe rozwiązania

Znaleziony błąd	Propozycje rozwiązania
Wciśnięcie i przytrzymanie przycisku	• Komunikat o błędzie i zatrzymanie gry, gdy przycisk jest zbyt długo w stanie wysokim • Wymuszenie zmiany stanu przycisku • Automatyczna wygrana drugiego gracza, gdy zarejestrowane zostało wciśnięcie przycisku w trakcie świecenia zielonej diody

5. Podsumowanie

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