Labyrinth - raport pracy zaliczeniowej

Hubert Gąsior, Mateusz Szczęśniak
EIT, Design Lab, poniedziałek 11.30

20.01.2025

Wstęp

Labyrinth to gra inspirowana mini-golfem, polegająca na manewrowaniu planszą w taki sposób, aby trafić kulą do celu, omijając przeszkody. Wariant prezentowany przez nas polega na sterowaniu planszą poprzez kontrolowanie ruchu telefonem - strzałkami lub z pomocą akcelerometra, zmieniając nachylenie pola gry. Rozgrywka kończy się, gdy kula trafi do środka planszy.

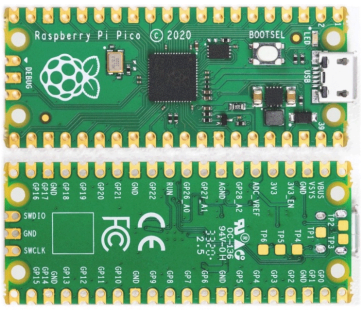
Realizacja

Przygotowano dwie plansze gry – z tektury oklejonej papierem technicznym oraz plastikowej podstawki z lekkimi ścianami wykonanymi z pianki. Obie zostały przygotowane z uwzględnieniem ich wagi, by nie ograniczać wydajności modułu sterującego.
Sterowanie jest zapewnione poprzez dwa silniczki, umocowane w sposób umożliwiający sterowanie nachyleniem osi poziomej oraz obrotem planszy ”dookoła” jej środka, kula do gry jest koralik typu marble. Komunikację między mikrokontrolerem, a aplikacją smartfonową zapewnia moduł Bluetooth. Zasilanie stanowią dwa zestawy trzech baterii umieszczonych w koszyczku. Całość jest zamontowana na płytce breadboard.

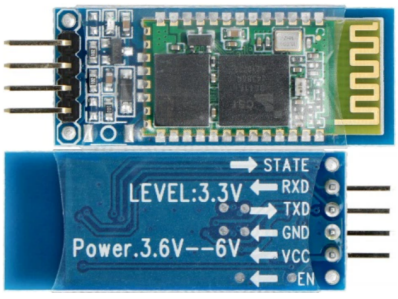
Wykorzystany sprzęt oraz oprogramowanie

Do projektu wykorzystano mikrokontroler *Raspberry Pi Pico 2020*, połączony z modułem Bluetooth *HC-06*. Podstawę zapewnia płytka stykowa justPi400, koszyki na baterie 3×AAA firmy *BOTLAND*. Silniczki to *Servo-sg90*, jeden z osią 180°, drugi 360°.
Oprogramowanie przygotowano korzystając ze środowiska ***Thonny***, za pomocą odmiany języka *MicroPython*, specjalnie dobranego dla *Raspberry Pi Pico 2020*. Aplikacja smartfonowa wykonano na podstawie ***RoboReboDemo***, umożliwiającej integrację komunikacji Bluetooth z modułami dostępymi w telefonach.

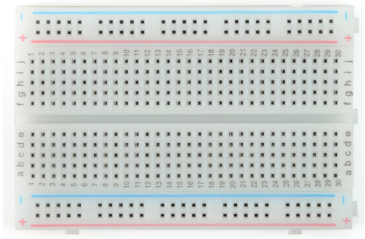
Wszystkie pliki skryptowe oraz dokumentacja jest dostępna na stronie projektu https://github.com/mszczesniak02/design_lab.



Rysunek 1: Mikrokontroler Raspberry Pi Pico

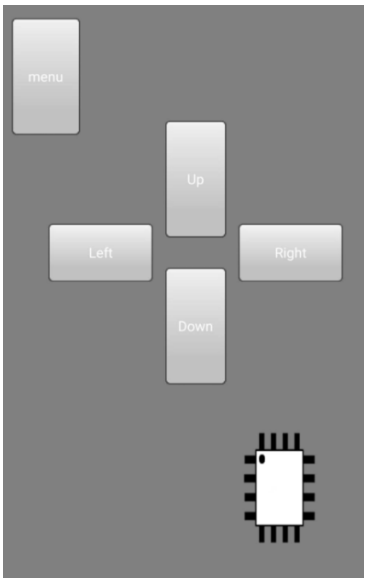


Rysunek 2: Moduł Bluetooth HC-06

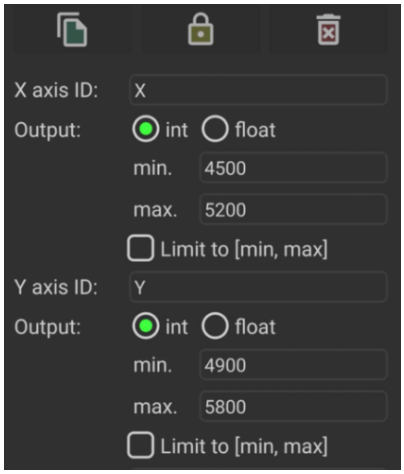


Rysunek 3: Płytka stykowa

Rysunek 4: Silnik Servo



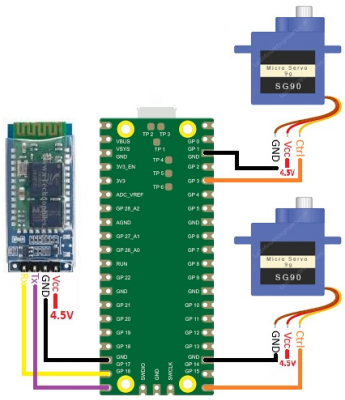
Rysunek 5: Aplikacja smartfonowa



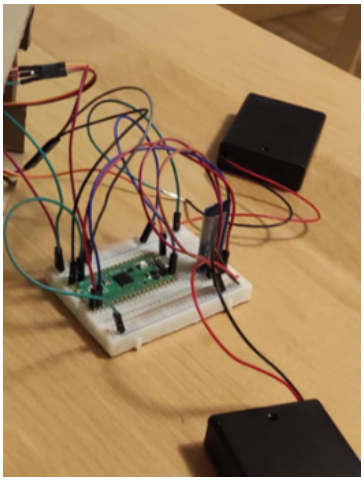
Rysunek 6: Aplikacja smartfonowa c.d.

Schemat połączenia oraz rzeczywiste połączenia

Dla przejrzystości schematu pominięto montowanie na płytce oraz podpięcie zasilania.



Rysunek 7: Schemat połączeniowy



Rysunek 8: Rzeczywiste połączenie

Opis wybranych fragmentów kodu

```python
def main():
    print("Startowanie...\n")
    uart = UART(UARTNUM, BAUDRATE )
    uart.init(tx=Pin(TXPIN), rx=Pin(RXPIN) )
    # set initial X engine
    XPWM.duty_u16(XSTOP)
    time.sleep(0.5)
    YPWM.duty_u16(YMID)
    curr = YMID
    print("Startowanie: OK")
    while True:
        direction = detect_command(uart).strip()
        if direction == 'R':
            XPWM.duty_u16(XRIGHT)
            time.sleep(0.5)
        elif direction  == "L":
            XPWM.duty_u16(XLEFT)
            time.sleep(0.5)
        elif direction  == "G":
            curr += 50
            print(curr)
            if curr <= YMAX and curr >= YMIN:
                YPWM.duty_u16(curr)
                time.sleep(0.5)
            else:
                curr = YMAX
                YPWM.duty_u16(curr)
                time.sleep(0.5)
        elif direction  == "D":
            curr -= 50
            print(curr)
            if curr <= YMAX and curr >= YMIN:
                YPWM.duty_u16(curr)
                time.sleep(0.5)
            else:
                curr = YMIN
                YPWM.duty_u16(curr)
                time.sleep(0.5)
        else:
            XPWM.duty_u16(XSTOP)
```

```python
def main():
    print("Startowanie...\n")
    uart = UART(UARTNUM, BAUDRATE )
    uart.init(tx=Pin(TXPIN), rx=Pin(RXPIN) )
    x_val_array = [0] * 100
    index_x = 0
    y_val_array = [0] * 100
    index_y = 0

    # set initial X engine
    XPWM.duty_u16(XSTOP)
    time.sleep(0.5)
    YPWM.duty_u16(YMID)
    curr = YMID

    print("Startowanie: OK")
    while True:
        direction = detect_command(uart).strip()

        if not direction:
            continue
        utf = is_utf8(direction)
        if utf == False:
            continue
        if len(bts(direction)) != 11:
            continue
        x_byte = direction[1:5]
        x_val = x_byte.decode('utf-8')

        y_byte = direction[7:11]
        y_val = y_byte.decode('utf-8')

        print(x_val)
        print(y_val)

        XPWM.duty_u16(int(x_val))
        YPWM.duty_u16(int(y_val))
        time.sleep(0.1)

```

Kod po lewej stronie to fragment wersji *main.py*, która pozwala na sterowanie grą używając strzałek w aplikacji, natomiast po prawej widnieje opcja pozwalająca na używanie akcelerometru. W obu wariantach konieczne jest zainicjowanie protokołu UART poprzez przypisanie pinów GPIO do ich funkcji, a następnie uruchomienie silników sterujących oraz ustawienie ich pozycji wyjściowej. W wersji sterowanej strzałkami odbierany komunikat odpowiada kierunkom: R – Right, L – Left, G – Góra, D – Dół, co pozwala na poruszenie odpowiedniego silnika. W wariancie z akcelerometrem kierunek przesyłany jest za pośrednictwem Bluetooth jako liczba z zakresu: dla osi Y od 4900 do 5800, a dla osi X od 4500 do 5200. Otrzymane dane są przetwarzane, a na ich podstawie sygnał PWM ustala kierunek oraz czas zmiany.

Prezentacja złożonego układu z planszami

Rysunek 9: Plansza 1

Rysunek 10: Plansza 2

Podsumowanie

Projekt „Labyrinth” pozwolił na stworzenie interaktywnej gry, spełniającej założenia dotyczące sterowania planszą za pomocą aplikacji protokołu przesyłu danych bezprzewodowo. W trakcie pracy udało się rozwiązać trudności związane z komunikacją Bluetooth i stabilnością mechaniki.

Dokumentacja techniczna użytego sprzętu

Noty katalogowe zostały pobrane od stron producentów oraz umieszczone w repozytorium projektu. Dostęp do odnośników z dnia 19.01.2025.

- Moduł Bluetooth HC-06https://github.com/mszczesniak02/design_lab/blob/main/dokumentacja_sprz%C4%99tu/HC-05%20Datasheet.pdf
- Raspberry Pi Pico https://github.com/mszczesniak02/design_lab/tree/main/dokumentacja_sprz%C4%99tu/pi_pico.pdf
- Silnik Servo https://github.com/mszczesniak02/design_lab/blob/main/dokumentacja_sprz%C4%99tu/Joystick_Iduino%20ST1079.pdf

Bibliografia

Dostęp do odnośników aktualny na dzień 19.01.2025.

- https://www.cisco.com/c/en/us/solutions/enterprise-networks/what-is-power-over-ethernet.html
- https://forum.arduino.cc/t/arduino-uno-hc-06-not-working/629934
- https://docs.micropython.org/en/latest/library/machine.UART.html
- https://www.youtube.com/watch?v=kVExDkRgeOE
- https://www.youtube.com/watch?v=ObmIWkHmN7g
- https://www.youtube.com/watch?v=8DOg8onUvks