Sprawozdanie z projektu “Kulka w labiryncie”

Systemy Wbudowane
Gabriel Naleźnik
Wojciech Gomułka
27.01.2021

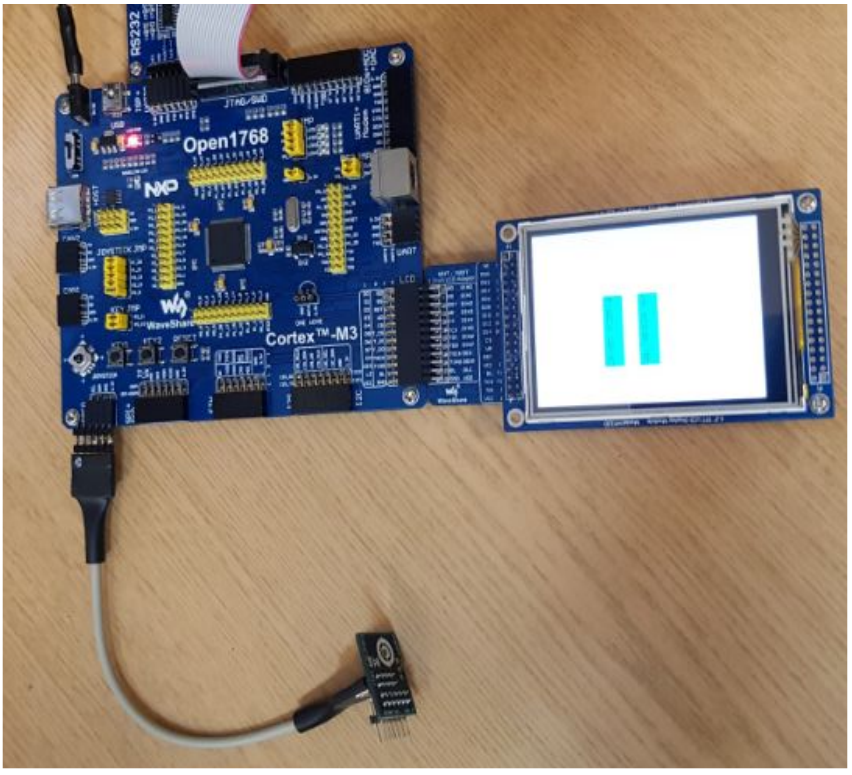
1. Opis projektu i założenia

Założeniem naszego projektu było stworzenie gry na mikrokontrolerze LPC1768, polegającej na rozwiązywaniu labiryntu za pomocą żyroskopu. Dodatkową funkcjonalność stanowić miały wybór rozmiaru labiryntu oraz zapisywanie i wyświetlanie dzięki rejestrom nieulotnym przynajmniej jednego, najlepszego rezultatu (najkrótszego czasu przejścia labiryntu).

2. Użytkowanie programu

2.1. Instrukcja budowy projektu

W celu uruchomienia aplikacji należy połączyć mikrokontroler LPC1768 z wyświetlaczem LCD 3.2` oraz żyroskopem Pmod GYRO. Żyroskop łączymy z płytką za pomocą zworek i rezystorów podciągających z pinami I2C0 naszej płytki. Używany jest interfejs I2C0, który linie SDA i SCL ma wyprowadzone na pinach P0.27 i P0.28. Podłączenia możemy dokonać w prostszy sposób widoczny na zdjęciu, ponieważ linie I2C0 dociągnięto do boku płytki. Gotowy zestaw został przedstawiony na poniższym obrazku.

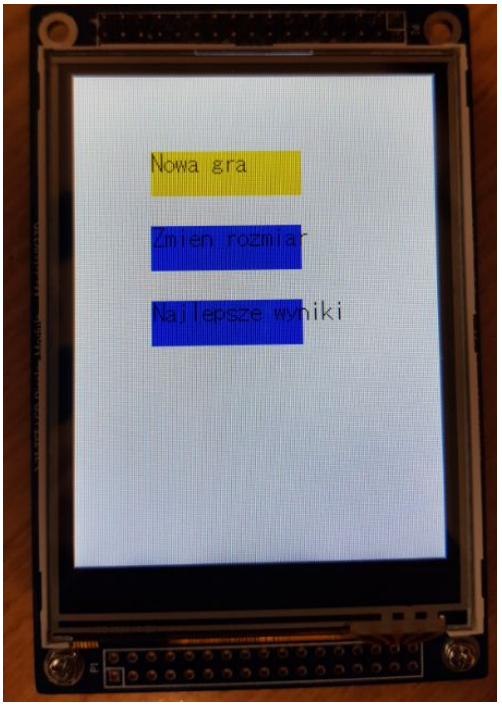


Rys. 1. Zdjęcie gotowego układu, widoczny sposób podłączenia żyroskopu do I2C0.

Rys. 2. Przybliżenie na podłączenie żyroskopu z rezystorami podciągającymi do I2C0

2.2. Menu główne

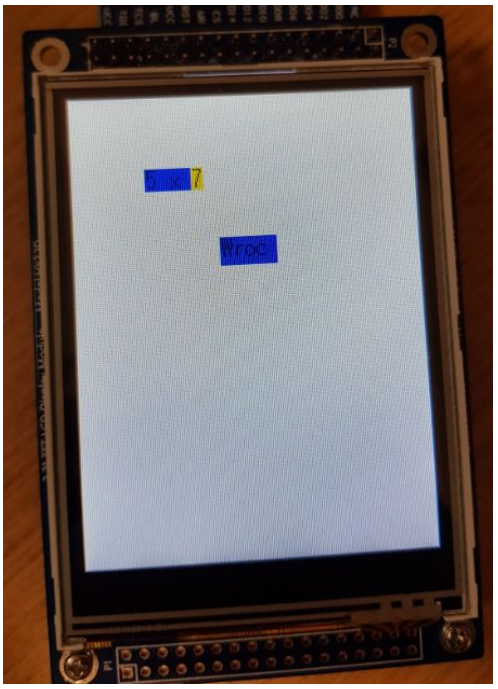
Po uruchomieniu aplikacji, naszym oczom ukazuje się menu główne interfejsu użytkownika z którego możemy wybrać 3 opcje: rozpocząć grę, zmienić rozmiar generowanego labiryntu oraz wyświetlić najlepszy wynik osiągnięty w grze. Sterowanie pomiędzy opcjami zostało zrealizowane za pomocą Joysticka wbudowanego w płytkę. Wybór opcji za pomocą wychylenia drążka, a zatwierdzenie wyboru poprzez naciśnięcie go.



Rys. 3. Menu główne - punkt wejścia aplikacji

2.3. Zmiana rozmiaru labiryntu

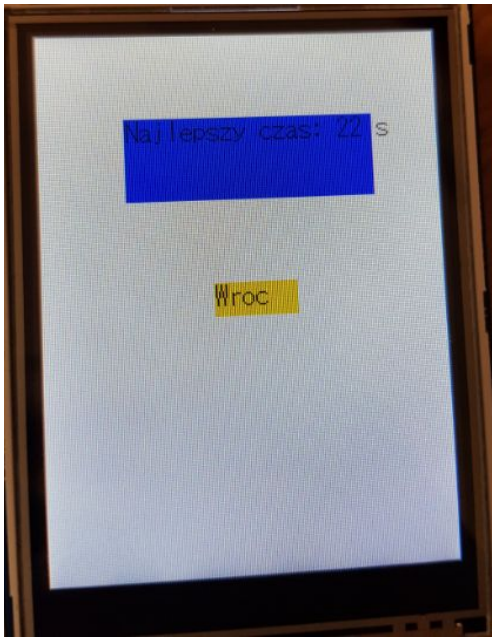
W tej opcji za pomocą Joysticka można wybrać rozmiar (szerokość x wysokość) generowanego labiryntu. Przejście do kolejnej opcji następuje po wciśnięciu drążka.



Rys. 4. Ekran wyboru rozmiaru labiryntu

2.4. Najlepsze wyniki

Na tej stronie prezentowany jest najlepszy czas przejścia labiryntu uzyskany przez użytkownika od czasu pierwszego resetu aplikacji.

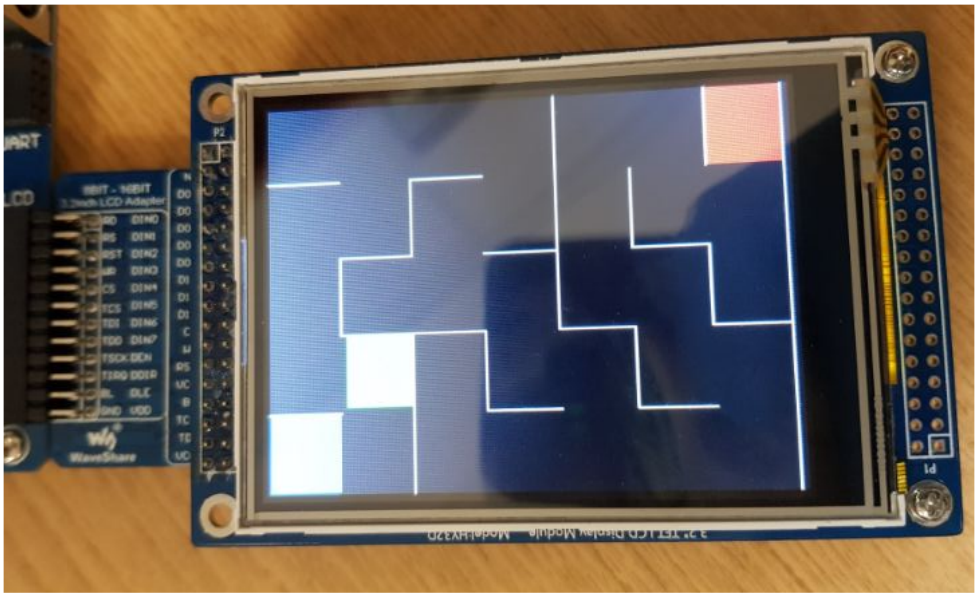


Rys. 5. Ekran wyświetlający najlepszy uzyskany wynik

2.5. Gra

W trakcie gry, na ekranie możemy wyróżnić 3 kwadraty: niebieski - start, zielony - gracza, czerwony - metę. Wejście gracza na metę jest równoznaczne z zakończeniem gry i zapisaniem wyniku. Sterowanie zostało zrealizowane na dwa sposoby:

- Za pomocą wbudowanego w płytkę joysticka (dokładne sterowanie graczem)
- Za pomocą żyroskopu (Ze względu na problemy z osiągnięciem dokładności zwracanych przez urządzenie wartości, ta forma sterowania jest nieprecyzyjna)



Rys. 6. Przykładowy labirynt wygenerowany dla rozgrywki

3. Wykorzystane komponenty i technologie

3.1. PmodGYRO

Żyroskop PmodGyro z wbudowanym chipem L3G4200D to układ mający w założeniu posłużyć do sterowania graczem podczas przechodzenia labiryntu. Żyroskop może komunikować się z innymi urządzeniami poprzez interfejs I2C lub SPI.
Zgodnie z założeniami projektu został skutecznie skonfigurowany do współpracy poprzez interfejs I2C.

Ideą było sterowanie kierunkiem ruchu gracza poprzez wychylanie żyroskopu w dwóch wymiarach. Po odczytaniu z żyroskopu wartości prędkości kątowej, przekraczającej ustaloną wartość progową, następuje inkrementacja lub też dekrementacja zmiennej global_y lub global_z informującej o aktualnym stanie wychylenia żyroskopu względem danej osi.
Ze względu na specyfikę układu (wysoką niedokładność zwracanych wartości) nie udało się zrealizować tego zadania, pomimo poprawnej komunikacji z mikrokontrolerem poprzez I2C.

3.2. 3.2inch Touch LCD

Wyświetlacz stanowił istotne peryferium dla naszego projektu. Pozwolił zwizualizować menu i przede wszystkim wyświetlić wygląd labiryntu, pozycje gracza, start oraz metę.

W projekcie nie wykorzystano touchpada umieszczonego na LCD, oddając sterowanie menu dla joysticka wbudowanego w płytkę ewaluacyjną.

Do rysowania kształtów: linii oraz prostokątów wykorzystano własną prostą bibliotekę graficzną (pliki graphics.h i graphics.c) ułatwiającą cały proces. Do rysowania linii posłużył algorytm Bresenhama implementowany na jednych z laboratoriów.

3.3. I^2C

Komunikacja z żyroskopem została zrealizowana poprzez interfejs I^2C. W pliku i2c.c została zaimplementowana maszyna stanów wywoływana w funkcji obsługi przerwania generowanego przez I^2C - I2C0_IRQHandler. Odczytuje ona status przerwania z rejestru I2STAT i na jego podstawie podejmuje odpowiednią akcję. Potrafi ona zapisywać dane do rejestrów w żyroskopie oraz odczytywać z nich wartości.

3.4. SysTick

Systick został skonfigurowany do wyzwalania przerwania co 0.1s, natomiast warunki na zmienną systick_counter pozwalają dostosować liczbę wywołań głównej pętli programu na sekundę. Domyślnie w naszej aplikacji jest to 0.6 sekundy.

SysTick jest timerem odpowiedzialnym za uruchamianie głównej funkcji aplikacji, nie wykorzystano go do mierzenia czasu przejścia labiryntu - o tym wspomnimy w następnym rozdziale.

3.5. RTC i rejestry nieulotne

Licznik czasu rzeczywistego został wykorzystany do liczenia czasu jaki gracz spędził na rozwiązywaniu labiryntu. Liczenie rozpoczyna się wraz z rozpoczęciem nowej gry i kończy się w momencie wejścia gracza na metę. RTC został skonfigurowany do generowania przerwania co jedną sekundę, a zliczanie czasu następuje w procedurze obsługującej przerwanie zegara czasu rzeczywistego RTC_IRQHandler. Po rozwiązaniu labiryntu, jeśli czas gracza jest lepszy od aktualnie najlepszego czasu zapisanego w tabeli wyników - rejestr nieulotny RTC, trzymający aktualnie najlepszy wynik, zostaje nadpisany nowym czasem.
Program można rozszerzyć, by mógł przechowywać większą liczbę wyników. Najprościej można to uzyskać poprzez wykorzystanie pozostałych czterech dostępnych rejestrów nieulotnych. Dla przechowania jeszcze większej liczby można by użyć operacji bitowych (przechowywać w jednym rejestrze na ustalonych bitach wyniki).

4. Bibliografia

- dokumentacje wykorzystanego żyroskopu oraz wykorzystywanego przez niego chipu L3G4200D:
  https://reference.digilentinc.com/_media/reference/pmod/pmodgyro/pmodgyro_rm.pdf
  https://reference.digilentinc.com/_media/reference/pmod/pmodgyro/stmicroelectronics-l3g4200d-datasheet.pdf
- instrukcje potrzebne do implementacji algorytmu generowania labiryntu (algorytm z powrotami, implementacja iteracyjna):
  https://en.wikipedia.org/wiki/Maze_generation_algorithm#Iterative_implementation
- do symulowania losowości przy generowaniu labiryntu wykorzystano generator pseudolosowy mieszany (multiplikatywno-addytywny)
  https://en.wikipedia.org/wiki/Lehmer_random_number_generator