Gra na płytce Open1768 – dokumentacja

I. Opis projektu

Projekt zakłada stworzenie gry przygodowej z wykorzystaniem płytki Open1768 oraz ekranu LCD. Poniżej przedstawiamy opis gry:

"Poszukiwacz Skarbu" to wciągająca gra labiryntowa, w której stajesz się odważnym poszukiwaczem skarbów. Twoim zadaniem jest nawigowanie po pełnym tajemnic labiryncie, gdzie każdy ruch wymaga sprytu i strategicznego myślenia. Twoim celem jest zebranie wszystkich ukrytych monet, a z każdą zebraną monetą zbliżasz się do otwarcia sekretnego przejścia, które poprowadzi Cię na zewnątrz labiryntu. Czy dasz radę zgarnąć złoto, zanim czas się skończy? Przygotuj się i rozpocznij przygodę w grze " Poszukiwacz Skarbu"!

II. Elementy sprzętu i sposób ich połączenia

Do projektu potrzebujemy:

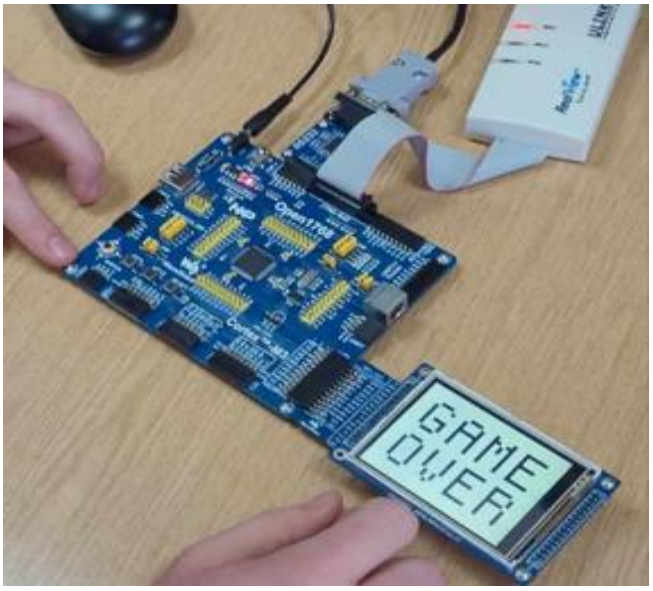
- Płytka Open1768
- Element UART
- Wyświetlacz 3,2" TFT (320x240)

Do płytki podłączamy UART w miejsce UART 0 oraz wyświetlacz w odpowiednim miejscu.

Konfigurujemy odpowiednio projekt, załączając odpowiednie biblioteki (USART, startup, joystick) w sekcji "Manage RunTime Environment" oraz załączyć biblioteki: "LCD_ILI9325.h" oraz "LPC17xx.h" do folderu ze źródłem programu. Uruchamiamy piny 1 i 2 UART0 oraz ustawiamy peripheral clock na: clk/2

III. Opis aplikacji

Za pomocą joysticka wbudowanego w płytce poruszamy się po planszy naszym poszukiwaczem. Gdy zbierzemy złotą monetę, znika ona z planszy, a gra śledzi liczbę zebranych monet. Po zebraniu wszystkich monet wyjście z labiryntu się otwiera (wcześniej nie możemy wyjść) i możemy ukończyć grę wychodząc na zewnątrz.



IV. Opis algorytmu

Po zainicjalizowaniu zmiennych startowych oraz funkcji rozpoczynamy pętlę gry a w niej wykonujemy takie czynności jak:

Czytanie Stanu Joysticka:
Na początku każdej iteracji pętli, program odczytuje stan joysticka. Ten stan określa, w którą stronę gracz chce się poruszyć (góra, dół, lewo, prawo).

Sprawdzanie Możliwości Ruchu:
Program sprawdza, czy zaproponowany ruch jest możliwy. Sprawdzenie obejmuje to, czy nowa pozycja nie znajduje się poza granicami labiryntu i czy nie jest zablokowana przez ścianę labiryntu.

Aktualizacja Pozycji Gracza:
Jeśli ruch jest możliwy, pozycja gracza na planszy jest aktualizowana. Stara pozycja gracza jest "czyszczona", a nowa jest ustawiana jako aktualna lokalizacja gracza.

Zbieranie Monet:
Jeśli nowa pozycja gracza zawiera monetę, jest ona "zbierana". Licznik zebranych monet jest inkrementowany.

Sprawdzanie Warunków Zwycięstwa:
Program weryfikuje, czy gracz zebrał wszystkie monety. Jeśli tak, przejście do wyjścia z labiryntu się otwiera umożliwiając zakończenie gry.

Aktualizacja Wyświetlacza:
Stan gry jest wyświetlany na ekranie LCD. Obejmuje to aktualizację pozycji gracza oraz stanu labiryntu.

Komunikacja przez UART:
Stan gry lub wykonane ruchy mogą być również wysyłane przez UART, co pozwala na zewnętrzną obserwację postępów gry (np. w terminalu TeraTherm).

Opóźnienie:
prowadzane jest krótkie opóźnienie, aby umożliwić czytelne śledzenie rozgrywki i zapewnić odpowiednią reakcję na wejście użytkownika.

V. Opisy funkcji

main():
Konfiguruje piny i UART.
Inicjalizuje SysTick i LCD.
Ustawia początkowe wartości i rozpoczyna pętlę gry. W pętli gry odczytuje stan joysticka, aktualizuje pozycję gracza i wyświetla informacje o stanie gry.

delay(uint32_t x):
Stwarza opóźnienie w działaniu programu. Używa zmiennej msTicks, która jest inkrementowana w funkcji SysTick_Handler. Opóźnienie trwa aż do osiągnięcia określonego czasu.

SysTick_Handler(void):
Obsługuje przerwania od zegara systemowego (SysTick).
Inkrementuje msTicks przy każdym przerwaniu.

send(char* str):
Wysyła ciąg znaków przez UART (transmisja szeregowa). Służy do komunikacji z innymi urządzeniami lub interfejsem komputera.

update_game():
Aktualizuje stan gry na ekranie LCD. Rysuje elementy gry w zależności od wartości w tablicy game_tab.

finish_game():
Wywoływana, gdy gra się kończy. Pokazuje ekran końca gry.

update_move(int new_man_x,int new_man_y):
Aktualizuje pozycję gracza na ekranie i wykonuje odpowiednie działania w zależności od rodzaju pola, na które gracz się przesunął.

check_if_move_possible(int new_man_x,int new_man_y):
Sprawdza, czy ruch gracza na dane pole jest możliwy (np. czy pole nie jest ścianą).

draw_the_man(int mx, int my):
Rysuje reprezentację gracza na ekranie w określonym miejscu.