Informatyka, studia dzienne, I st. semestr IV

Systemy wbudowane **2018/2019**

Prowadzący: dr inż. Michał Morawski środa, 12:15

Grupa D07

Przemysław Rudowicz	216879	216879@edu.p.lodz.pl - lider
Konrad Jaworski	216782	216782@edu.p.lodz.pl
Jakub Plich	216866	216866@edu.p.lodz.pl

Dokumentacja projektu gry Snake LPC1768/9

Spis treści

1. Podział obowiązków

1.1. Wykorzystane funkcjonalności

Funkcjonalność	**Osoba za nią odpowiedzialna**
GPIO (joystick)	Konrad Jaworski
Akcelerometr	Konrad Jaworski
Głośnik	Konrad Jaworski
Timer	Przemysław Rudowicz
OLED	Przemysław Rudowicz
SSP/SPI	Przemysław Rudowicz
Czujnik światła	Jakub Plich
pca9532	Jakub Plich
I^2C	Jakub Plich
Rotacyjny przełącznik kwadraturowy	Jakub Plich

1.2. Podział obowiązków

Imię i nazwisko	**Procentowy udział w pracy**
Konrad Jaworski	33%
Przemysław Rudowicz	34%
Jakub Plich	33%

2. Opis działania programu

2.1. Instrukcja użytkownika

Po podłączeniu płytki do zasilania na wyświetlaczu OLED zostanie narysowany wąż (reprezentowany przez ciąg pikseli tworzący linię łamaną) oraz owoc reprezentowany przez grupę 4 pikseli (kwadrat 2x2 w lewej, górnej części ekranu).

Aby rozpocząć grę należy wcisnąć przycisk z prawej strony joysticka.

Do zmiany kierunku poruszania się węża można użyć joysticka lub akcelerometru (przechylać płytkę).

Celem gry jest zebranie/zjedzenie jak największej liczy owoców. Dokonuje się to poprzez zderzenie się głowy węża oraz ciała owocu. Aktualny wynik gracza (liczba zjedzonych owoców) jest przedstawiany w postaci binarnej na diodach LED expandera.

Gracz przegrywa w momencie, gdy głowa węża uderzy w ścianę lub w ciało węża (również, gdy gracz spróbuje 'cofnąć się' - zmienić kierunek poruszania się na przeciwny do obecnego).

Gracz może kontrolować szybkość poruszania się węża przy pomocy rotacyjnego przełącznika kwadraturowego (obrót w lewo zwiększa prędkość, a w prawo zmniejsza).

Po zakryciu czujnika światła, kolory na ekranie zostają odwrócone.

Aby zagrać kolejny raz należy ponownie wcisnąć przycisk z prawej strony joysticka.

2.2. Opis algorytmu

Program został napisany w języku C. Wąż porusza się poprzez usuwanie ostatniej części swojego ciała i dodawanie nowego punktu na pozycji (obok głowy) wskazanej przez kierunek w którym ma się poruszać. Kierunek jest ustalany przez gracza poprzez wybranie go joystickiem lub przechylenie płytki w wybraną stronę.

W głównej pętli funkcji main() sprawdzamy stan rotacyjnego przełącznika kwadraturowego i jeżeli program wykryje jego obrót, następuje inkrementacja lub dekrementacja zmiennej odpowiedzialnej za czas przestoju węża. Następnie pobierany jest stan akcelerometru i joysticka. Jeżeli stan joysticka zgadza się z wybranymi kierunkami (lewo, prawo, góra, dół) następuje przypisanie tego kierunku do kierunku poruszania się węża. Jeżeli wartości oczytane z akcelerometru przekroczą zadaną wartość, również nastąpi przypisanie do kierunku poruszania się węża, kierunku, w którym płytka została przechylona. Jeżeli wąż nie jest w stanie zablokowanym (nie uderzył wcześniej w ściany lub w samego siebie), wywoływana jest funkcja odpowiedzialna za poruszanie się węża, a następnie rysowanie go i sprawdzanie kolizji. Jeżeli program wykryje przysłonięcie czujnika światła, następuje odwrócenie kolorów rysowanych obiektów. Sprawdzany jest również stan wciśnięcia przycisku odpowiedzialnego za rozpoczynanie nowej tury gry.

3. Opis działania wykonanego sprzętu

Nie było wykonanego sprzętu.

4. Funkcjonalności

4.1. GPIO

GPIO (oznacza general-purpose input/output) - interfejs wejścia/wyjścia ogólnego przeznaczenia. Należy ustawić kierunki wejścia/wyjścia pinów GPIO (0 - gdy chcemy skonfigurować pin jako wejście, lub 1 - jako wyjście).

4.1.1. Głośnik

Głośnik jest obsługiwany przy pomocy pinów GPIO. Jako, że głośnik nie będzie wysyłał danych, piny ustawiamy na wyjście. W tym celu ustawiamy wartość 1 w rejestrach FIODIR0 i FIODIR2 w miejscach odpowiadających pinom głośnika (każdy bit rejestru odpowiada jednemu pinowi GPIO, każdy port GPIO ma swój rejestr FIODIR). A więc 1 należy ustawić na 28, 27, 26 bicie FIODIR0 i 13 bicie FIODIR2.

Wzmacniacz analogowy LM4811, który znajduje się na płytce LPCXpresso Base Board potrzebuje następujących pinów z mikrokontrolera:
— CLK
— UP/DN
— SHUTDN
— VIN1/VIN2

Ze specyfikacji LM8411 [2] dowiadujemy się, że piny CLK (CLOCK) oraz UP/DN są odpowiedzialne za sterowanie głośnością brzęczyka.

Pin SHUTDN aktywuje funkcję zmniejszającą pobór prądu przez wzmacniacz (Nie korzystamy z tej funkcji).

Piny VIN1/VIN2 odpowiadają za generację sygnału wprawiającego membranę brzęczyka w drgania (generowanie dźwięków).

Sposób połączenie pinów wzmacniacza analogowego do pinów GPIO:

Piny LM4811	**Piny GPIO**
CLK	P0.27
UP/DN	P0.28
SHUTDN	P2.13
VIN1/VIN2	P0.26

Podczas inicjalizacji głośnika czyszczone są wartość na pinach P0.27, P0.28, P2.13 (ustawiamy 1 w rejestrach FIOCLR dla portu 0 i 2 w miejscach odpowiadających wymienionym pinom).

Generowanie dźwięku przez brzęczyka odbywa się poprzez podawaniu zmiennego napięcia na pin P0.26 tak aby wprowadzić membranę brzęczyka w drgania. Pozwala to na generowanie prostych nut.

Aby zagrać nutę 'C', należy wprowadzić membranę brzęczka w drgania o częstotliwości $f = 262Hz$. A więc okres drań $T = \frac{1}{f} = 3816\mu s$. Stąd na pinie

P0.26 należy ustawić stan wysoki przez czas równy $\frac{T}{2} = 1908\mu s$ oraz stan niski analogicznie przez $\frac{T}{2}$. Cykl należy powtarzać w zależności od tego jak długo chcemy odtwarzać dźwięk. Do ustawiania stanów wysokich i niskich na pinach GPIO używamy rejestru FIOSET i FIOCLR. Za generowanie dźwięku odpowiada pin P0.26. Analogicznie postępujemy w przypadku innych nut.

W celu ustawienia stanu wysokiego na pinie P0.26 należy ustawić 1 na 26 bicie rejestru FIOSET (ustawianie zera na tym rejestrze nie ustawia stanu niskiego). Aby odwołać stan wysoki należy wpisać 1 na 26 bicie rejestru FIOCLR.

4.1.2. Joystick

Joystick również jest obsługiwany przy pomocy pinów GPIO. Natomiast w przeciwieństwie do głośnika, joystick wysyła dane do mikrokontrolera, a więc podczas jego inicjalizacji ustawiamy wszystkie piny na wejście.

W tym celu ustawiamy wartość '0' na 15, 16 i 17 bicie rejestru FIODIR0 oraz na 3 i 4 bicie rejestru FIODIR2.

Pozycja joysticka	**wartość**	**Piny GPIO**
JOYSTICK_CENTER	0x01	P0.17
JOYSTICK_UP	0x02	P2.3
JOYSTICK_DOWN	0x04	P0.15
JOYSTICK_LEFT	0x08	P2.4
JOYSTICK_RIGHT	0x10	P0.16

Stany podłączonych pinów odpowiadają stanom wciśnięcia joysticka (odpowiednio tak jak w tabeli powyżej). W celu odczytania stanu joysticka sprawdzane są wartości na kolejnych pinach (odpowiednio tak jak w tabeli powyżej) i jeżeli jego wartość to '0', zmienna przechowująca stan joysticka przyjmuje wartość koniunkcji bitowej tego stanu i odpowiadającej wartości (patrz tabela powyżej) przypisanej do pozycji joysticka.

4.2. Akcelerometr

Użyty przez nas akcelerometr MMA7455L [1] jest inicjalizowany (za pomocą I^2C (adres urządzenia - 0x1D)) przez zdefiniowanie trybu pomiaru i czułości. Dokonujemy tego poprzez wpisanie do rejestru MCTL (Mode Control o adresie 0x16) wartości 1 na bicie zerowym (odpowiada za ustawienie czułości na 2g) oraz wartości 1 na bicie drugim (odpowiada za ustawienie trybu pomiarowego (Measurement Mode)).

Adres	Nazwa rejestru	Bit 3 (GLVL[1])	Bit 2 (GLVL[0])	Bit 1 (MOD[1])	Bit 0 (MOD[0])
0x16	MCTL	0	1	0	1

Aby odczytać wartości wskazań akcelerometru najpierw zostaje sprawdzona wartość z rejestru przechowujące informację o statusie akcelerometru (rejestr o adresie 0x09). Na bicie zerowym tego rejestru (DRDY) przechowywana jest informacja, czy dane są gotowe do odczytania.

Wartość na bicie DRDY rejestru 0x09	Znaczenie
1	dane są gotowe do odczytania
0	dane nie są gotowe do odczytania

Jeżeli dane są gotowe do odczytania, z rejestru XOUT8 (o adresie 0x06) pobierane jest 8 bitów, które opisują składową X wektora przyspieszenia. Następnie, kolejno odczytywane są wartości z rejestrów YOUT8 (0x07) oraz ZOUT8 (0x08), które opisują odpowiednio składowe Y i Z.

Adres rejestru	Nazwa rejestru
0x06	XOUT8
0x07	YOUT8
0x08	ZOUT8

4.3. Timer

4.4. OLED

4.5. SSP/SPI

4.6. Czujnik światła

Czujnik światła jest urządzeniem peryferyjnym przyłączonym do płytki magistralą I^2C. Program wkorzystuje odczytane natężenie światła do odwrócenia kolorów na wyświetlaczu w momencie w którym odczytana wartość natężenia światła będzie mniejsza niż 25 luksów. Uruchomienie czyjnika światła polega na przesłaniu przez I2C pod adres (0x44) kolejno wartości (0x00) oraz (1«7) . To powoduje ustawienie wartości 1 na 7 bicie rejestru Command Register(0x00) i w konsekwencji uruchomienie przetwornika analogowo-cyfrowego w czujniku. Zakres odczytu czujnika jest domyślny i wynosi od 0 do 1000 luksów. Odczyt wartości pomiaru czujnika wymaga odczytania zawartości dwóch rejestrów: LSB-Sensor - zawiera dolny bajt ostatniego odczytu sensora(adres 0x04), MSB-Sensor - zawiera górny bajt ostatniego odczytu sensora(adres 0x05). Wynik wyrażony w luksach jest obliczany z następującego wzoru: E = 973 * odczytana-wartość / (1«16)

4.7. PCA9532

Expander PCA9532 wyposażony w 16 diód LED jest wkorzystywany do reprezentacji wyniku w danym momencie gry. Zapalone diody przedstawiają wynik w postaci binarnej. W celu zapalenia odpowiednich diód należy ustalić 16 bitową maskie w której wartości 1 oznaczają zapaloną diodę. Następnie przez I2C pod adres rejestru kontrolnego(0x60) zostaje wysłany bajt kontrolny. Trzeci bit bajtu kontorlnego onacza flagę inkrementacji która zwiększa o 1 adres podany w pozostałych 4 bitach po każdym przesłanym bajcie. Potem przesłane zostają 4 bajty które zostają kolejno wpisane do 4 8-bitowych rejestrów(LS0,LS1,LS2,LS3) w których każde 2 bity odpowiadają jednej diodzie. Program korzysta jedynie ze stanów OFF(00) oraz ON(01).

4.8. I²C

I2C (Inter-Integrated Circuit) to szeregowa, dwukierunkowa magistrala do przesyłania danych. Program wykorzystuje ją do komunikacji z PCA9532, czujnikiem światła oraz akcelerometrem. Początek inicjalizacji magistrali zaczyna się od konfiguracji pinów które będą pełniły fukcję lini SCL(linia zegara) i SDA(linia danych). Dla interfejsu I2C2 są one ustawione odpowiednio na pinach P0[11] oraz P0[10]. W tym celu ustawiamy w rejestrze PINSEL0 wartosci 1 i 0 kolejno dla bitów 21 i 20(P0.10 SDA2) oraz wartość 1 i 0 dla bitów 23 i 22(P0.11 SCL). Następnie następuje włączenie zasilania dla I2C poprzez ustawienie wartości 1 na 26 bicie rejestru PCONP(Power Control for Peripherals Register). Ustawiamy dzielnik zegara PCLK na 2 ustawiając w rejestrze PCLKSEL 20 i 21 bit na wartości kolejno 0 i 1. Następnie należy ustawić wartość rejestrów I2SCLH i I2SCLL na żądaną ilość cykli zegara PCLK. Obie te wartości są sobie równe.(...)

Na koniec występuje ustawienie na wartość 1 bitów 2(flaga AA), 5(flaga STA) i 6(flaga I2EN) rejestru CONCLR. Na koniec należy w rejestrze I2CONSET ustawiamy wartość 1 na 6 bicie aby włączyć interfejs I2C2.

4.9. Rotacyjny przełącznik kwadraturowy

Rotacyjny przełącznik kwadraturowy jest wykorzystywany w naszym programie do regulacji prędkości poruszania się "węża". Obrót przełącznika w prawo zmniejsza prędkość, a obór w lewo zwiększa. Inicjalizacja ogranicza się do ustawienia na porcie 0 GPIO wartości 0 na 24 i 25 bicie rejestru FIODIR(zerowy i pierwszy bit rejestru FIO0DIR3). Wartość 0 oznacza że pin jest ustawiony na wejście. Program odczytuje inforamcję o 4 stanach(są ustalone na podstawie położenia przełącznika). W zależności od kolejności występowania stanów ustalany jest kierunek ruchu przełącznika.

5. Analiza FMEA

Ryzyko	Prawdopodobieństwo	Znaczenie	(Samo)Wykrywalność	Iloczyn	Reakcja
Uszkodzenie joysticka					

Literatura

[1] *±2g/±24g/±28g Three Axis Low-g Digital Output Accelerometer*, Rev 8, 07/2009, **Freescale Semiconductor**

[2] *LM4811 Dual 105mW Headphone Amplifier with Digital Volume Control and Shutdown Mode Datasheet*, December 2002, **National Semiconductor**