Politechnika Łódzka

Wydział Fizyki Technicznej, Informatyki
i Matematyki Stosowanej

Kierunek: Informatyka
Przedmiot: Systemy wbudowane

Dokumentacja gry zręcznościowej Procesja™ LPC1343 + Expansion Board

lider: mgr Filip Turoboś

Nr albumu: 210344801147

Jan Filipowicz

Nr albumu: 203875

Krzysztof Wierzbicki

Nr albumu: 210347

Łódź,

Spis treści

1 Podsumowanie i technikalia

1.1 Skład zespołu

Funkcja:	Imię i nazwisko:	Nr albumu:
lider	Filip Turoboś	210344, 801147
członek	Jan Filipowicz	203875
członek	Krzysztof Wierzbicki	210347

1.2 Wykorzystane funkcjonalności i ich autorzy

- Timer

1.2.1 Dokumentacja

Opis poszczególnych funkcjonalności: Filip Turoboś Skład i opracowanie całości dokumentu: Filip Turoboś

1.2.2 Procentowy udział poszczególnych członków zespołu w tworzeniu końcowej wersji projektu

Imię i nazwisko:	Procentowy udział:
Filip Turoboś	30%
Jan Filipowicz	38%
Krzysztof Wierzbicki	32%

2 Skrótowy opis działania programu

Podczas gry w Procesja^{TM} gracz manewruje przy pomocy joysticka i/lub akcelerometru białą łamaną symbolizującą procesję i stara się poruszać nią w taki sposób, aby:

- nie przecinać tworzonej przez siebie łamanej;
- zbierać pojawiające się na planszy kropki symbolizujące zbłąkane owieczki.

W przypadku gdy gracz najedzie łamaną na pulsującą kropkę, całość procesji zostaje przedłużona. Radując się z ilości zgromadzonych wiernych możemy odgrywać pieśń sakralną *"Barka"* autorstwa Jana Pawła II.

2.1 Timer

Układ, który dekrementuje lub inkrementuje wartość jednego ze swoich rejestrów w zależności od częstotliwości otrzymywanego sygnału nazywamy **timerem**. Każdy timer jest wyposażony w dwa podstawowe rejestry – licznik timera i rejestr kontroli timera. W przypadku płytki LPC 1343 timer jest dodatkowo wyposażony w preskaler. Zwiększenie licznika timera (TC) – *timer counter* następuje po spełnieniu następujących warunków:

- Wartość rejestru preskalera (PR) jest ustawiona na pewną wartość $K \in \mathbb{N}_0$ – domyślnie $K = 0$;
- 32-bitowy licznik preskalera (PC) osiągnie wartość $K + 1$;

Po inkrementacji (TC) następuje wyzerowanie licznika preskalera.

2.2 GPIO

Skrót ($GPIO$) oznacza *General purpose input/output*, czyli **interfejs wejścia /wyjścia ogólnego przeznaczenia**. Przy pomocy ($GPIO$) obsługiwany jest joystick, w który wyposażona jest płytka. Służy on do poruszania się instancją obiektu typu wąż, umożliwiając tym samym granie w grę.

Aby używać joysticka musimy najpierw ustalić rolę odpowiednich pinów na *INPUT*. W naszym przypadku stosowane jest port nr 2 i piny od 1 do 4, odpowiadających za poszczególne kierunki (odpowiednio dół/prawo/góra/lewo).

Posuszanie wężem polega na sprawdzaniu stanów poszczególnych pinów. Przykładowo, gdy ostatnio odczytany stan wysoki wystąpił na pinie nr 2, nasz wąż zacznie poruszać się w prawo (o ile jest to możliwe, tj. nie poruszał się uprzednio w lewo). Jeżeli nie jest odczytywany obecnie stan wysoki na żadnym z pinów, to wąż kontynuuje poruszanie się w ostatnio wybranym kierunku. W przypadku, gdy odczytany ruch jest przeciwny do obecnego, sygnał odebrany z ($GPIO$) zostanie zignorowany.

2.3 ADC

Analog-Digital Converter, czyli **Konwerter analogowo cyfrowy** (ADC) jest przetwornikiem pozwalającym na zmianę zewnętrznego sygnału analogowego (w naszym przypadku napięcia) na ciąg 10 bitów. ADC na stosowanym przez nas mikrokontrolerze używa algorytmu sukcesywnej aproksymacji, którego krótki opis załączony zostanie w podrozdziale **Algorytm sukcesywnej aproksymacji w** ADC.

Przy pomocy ADC gracz jest w stanie kontrolować prędkość poruszanej przez siebie procesji. Precyzyjnie, jeden krok procesji zajmuje

$$\text{new\_snek\_speed} = \left(\left\lfloor \frac{4000}{\text{ADCread(0)}} \right\rfloor + 20 \right) \ ms.$$

Do odczytu wartości z ADC stosowana jest funkcja *ADCread(0)*, która odczytuje wartość napięcia na zerowym pinie. Odczytywane przez nas wartości z ADC są liczbami całkowitymi z przedziału $[39; 1023]$. Dzięki temu nigdy nie dochodzi do dzielenia przez zerową wartość. Funkcja odczytująca wartość wywoływana jest przy każdym obrocie głównej pętli programu. Dzięki temu wartość prędkość procesji jest dostosowywana na bieżąco do aktualnego napięcia na zerowym pinie ADC, które kontroluje użytkownik przy pomocy pokrętła A22K. Zgodnie z [1] napięcie to mieści się w zakresie od 0 do 3.6 V.

2.3.1 Algorytm sukcesywnej aproksymacji *ADC* [2]

Algorytm sukcesywnej aproksymacji jest w istocie sprzętową wersją przeszukiwania binarnego. Rejestr SAR (*Successive Approximation Register*), do którego zapisywana bedzie odczytana z konwertera wartość jest pierwotnie zainicjalizowany samymi zerami, zaś najbardziej znaczący bit jest przestawiany na wartość 1. Następnie wartość rejestru jest konwertowana na sygnał analogowy, po czym komparator porównuje napięcie wejściowe z konwertera z napięciem odpowiadającym ciągowi bitów z rejestru SAR. Jeżeli napięcie wejściowe w konwerterze (czyli sygnał otrzymywany dzięki ustawieniu pokrętła na odpowiedniej pozycji) przekracza wartość napięcia wygenerowanego na podstawie rejestru – bit wiodący pozostaje w stanie 1, w przeciwnym razie przypisana zostaje mu wartość 0. Następnie analogiczne postępowanie zostaje przeprowadzane dla sukcesywnego bitu. Procedura powtarzana jest dziesięciokrotnie, do zapełnienia całego rejestru.

2.4 SPI/SSP

Serial Peripheral Interface / Synchronous Serial Port, interfejs ten jest używany, by komunikować się z wyświetlaczami. Jak wskazuje nazwa, komunikacja przebiega synchronicznie poprzez trzy równoległe linie. Są nimi

- *MOSI* - *Master Input Slave Output* pozwala na wysyłanie danych do zewnętrznego układu (w naszym wypadku wyświetlaczy);

- *MISO* - *Master Output Slave Input* służy do pobierania danych z układu zewnętrznego;

- *SCLK* - *Serial CLocK* zegar taktujący – precyzyjniej sygnał zegarowy.

2.5 I^2C

2.6 Wyświetlacz siedmiosegmentowy

W zaprojektowanej grze wyświetlacz siedmiosegmentowy służy jako element dekoracyjny i nie ma wpływu na rozgrywkę. Przez cały czas trwania rozgrywki wyświetla on kolejne litery napisu.

1. Inicjalizacja wyświetlacza siedmiosegmentowego polega na ustawieniu kierunku *out* w pierwszym porcie GPIO na 11 bicie

```
LPC_GPIO1->DIR |= (0x1<<11);
```

a następnie ustawienie na tym bicie wartości 1 – sprawia to, że wyświetlacz jest gotowy do użytku i nie wyświetla jeszcze żadnego znaku.

```
 GPIOShadowPort1 |= (1<<11);
```

2. Deklarujemy wartość zmiennej letters_speed, która opisywać będzie co ile iteracji głównej pętli programu nastąpi zmiana wyświetlanego znaku (poprzedzona jednoiteracyjną przerwą):

```
const uint8_t letters_speed = 10;
```

Tablica znaków do wyświetlania przez wyświetlacz siedmiosegmentowy ma następującą deklarację:

```
const uint8_t letters[] = "NO STEP ON SNECC ";
```

3. W pojedynczej iteracji głównej pętli programu wykonujemy następujące kroki:

3.1 Upewniamy się, że zmienna kontrolująca wartość wyświetlanej litery letters_state nie przekroczy wartości **sizeof**(letters)·letters_speed.

```
letters_state %= letters_size * letters_speed;
```

3.2 Wybieramy instrukcją warunkową czy obecnie wyświetlany jest znak z tablicy letters, czy też następuje przerwa, która czyni napis czytelniejszym dla użytkownika. Zestaw stosowanych w programie znaków pozwala wyświetlić każdy z nich na wyświetlaczu siedmiosegmentowym. Fałsz przekazywany jako drugi argument funkcji sprawia, że nie wchodzimy w tryb surowego przesyłu danych (tryb w którym korzystamy z funkcji led7seg_setChar sprawia, że litery odczytywane są jako znaki ASCII):

```
led7seg_setChar(letters_state % letters_speed ?
letters[letters_state / letters_speed] : ' ', FALSE);
```

Tu trochę wyjaśnienia dotyczacego przesyłania danych przez SSP.

3.3 Zwiększamy wartość letters_state o jeden.

```
letters_state++;
```

2.7 Wyświetlacz LCD

2.8 Przerwania (Interrupts)

2.9 DAC

2.10 Akcelerometr

3 Analiza FMEA

Możliwa awaria	Prawdopodobieństwo	Reakcja	Istotność
Uszkodzenie wyświetlacza LCD	mrg@duke.edu	xD	Krytyczna
Uszkodzenie joysticka	mje7@duke.edu	Xd	Wysoka
Uszkodzenie akcelerometru	mje7@duke.edu	Xd	Średnia
Uszkodzenie wyświetlacza 7segmentowego	xD	XD	Znikoma

3.1 Uszkodzenie wyświetlacza LCD

3.1.1 Rozpoznanie

3.1.2 Reakcja

3.2 Uszkodzenie joysticka

3.2.1 Rozpoznanie

3.2.2 Reakcja

3.3 Uszkodzenie akcelerometru

3.3.1 Rozpoznanie

3.3.2 Reakcja

3.4 Uszkodzenie wyświetlacza 7segmentowego

3.4.1 Rozpoznanie

3.4.2 Reakcja

4 Wykorzystane noty katalogowe, dokumentacja i literatura

Bibliografia

[1] *UM10375 LPC1311/13/42/43 User manual*, 21 June 2012, **Rev. 5**,

[2] *Understanding SAR ADCs: Their Architecture and Comparison with Other ADCs* 02 October 2001 **Maxim Integrated Products, Inc.**