PukPuk – uniwersalny sterownik aktywowany stuknięciem

Zastosowanie akcelerometru MPU 6050 i magistrali I^2C

AGH

AKADEMIA GÓRNICZO-HUTNICZA
IM. STANISŁAWA STASZICA
W KRAKOWIE

Systemy mikroprocesorowe 2 – projekt końcowy

Filip Polednia i Mateusz Kaczmarczyk
Elektronika III rok

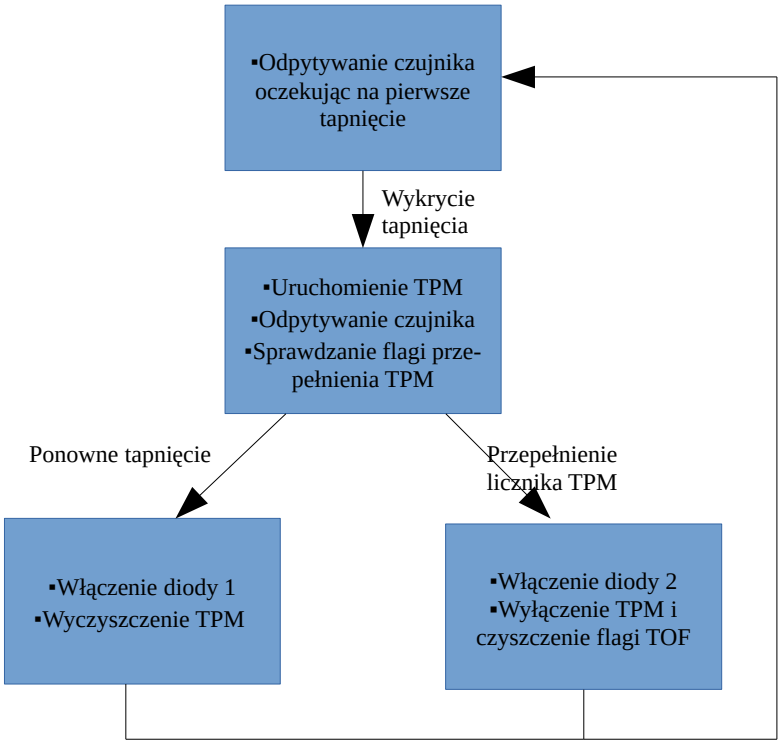
1 Faza projektu

Do realizacji postawionego przed nami celu jakim było wykonanie urządzenia demonstrujące możliwości czujnika MPU 6050 wykorzystywaliśmy oprogramowanie Keili uVision oraz analizator stanów logicznych Salae wraz z oprogramowaniem Logic. Całość zaimplementowana została na płytce rozwojowej FRDM-KL46Z firmy freescale/nxp. Plan wykonania projektu zakładał dwie wersje podstawową i rozwiniętą, pierwsza zakładała pracę całości w trybie ciągłym i stałe odpytywanie czujnika. Druga polegała na implementacji przerwań w projekcie co pozwoliło ograniczyć zużycie energii przez procesor który może być uśpiony i oczekiwać na wyzwolenie pochodzące z czujnika. W całej dokumentacji używamy zapożyczenia "tapnięcie" zamiennie z polskim odpowiednikiem stuknięcie i puknięcie.

2 Opis programu

I. Podejście 1

Założyliśmy, iż sterowanie będzie odbywało się jednym lub dwoma tapnięciami w urządzenie, które będą włączały i wyłączały odpowiadające im diody na płytce mikroprocesora. Diagram 1 obrazuje sposób działania programu i kolejne fazy jego wykonywania w zależności od tego ile razy użytkownik puknie w urządzenie.



Schemat 1: Podstawowy schemat stanów programu

II. Podejście 2

Z uwagi na wysoki poziom komplikacji podczas implementacji poprzedniej wersji zdecydowaliśmy się zastosować możliwość automatycznej generacji przerwań przez czujnik w momencie wykrycia stuknięcia. Postanowiliśmy wykorzystać oba piny przerwań sensora: INT1 informuje o nowym odczycie przyspieszenia (co w projekcie nie jest wykorzystywane a służy jedynie do prezentacji), INT2 natomiast przekazuje informację o wykryciu tapnięcia. Oba przerwania obsługuje „handler" portu C i D ustawiając dwie zmienne odpowiadające wykryciu któregoś przerwania co wykorzystujemy później, odczytując w pętli main właściwe rejestry. Testując rejestr PULSE_SRC można łatwo ustalić w której osi wykryto puknięcie, jego kierunek oraz to czy było ono pojedyncze czy podwójne. Prostota wyboru tego trybu pracy była wręcz zadziwiająca a efekty więcej niż zadowalające.

3 Zebrane dane

Poprawne przygotowanie kodu wymagało ustalenia progu wykrywania puknięcia jak również jego trwania, optymalne okazały się przyspieszenie ok 0,25 g wyzwalające wykrywanie tapnięcia a jego czas trwania nie może przekroczyć 3,75 ms. Dodatkowo ustaliliśmy opóźnienie między kolejnymi puknięciami w płytkę na 180ms, którego przekroczenie wyklucza wykrycie podwójnego tapnięcia.

4 Konfiguracja akcelerometru

Nazwa rejestru	Wartość	Ustawienie
XYZ_DATA_CFG_REG	0x00	Zakres czułości 2g, wyłączony HPF
PULSE_CFG_REG	0x71	Włączenie wykrywania stuknięć w osi X i Z oraz zatrzaskiwania danych
PULSE_THSZ_REG	0x50	Próg stuknięcia w osi Z
PULSE_THSX_REG	0x40	Próg stuknięcia w osi X
PULSE_TMLT_REG	0x06	Limit czasu trwania stuknięcia to 3,75 ms
PULSE_WIND_REG	0x50	Okno czasowe podwójnego stuknięcia to 180 ms
CTRL_REG3	0x00	Silne wymuszenia na pinach przerwań, zaprzeczona logika
CTRL_REG4	0x09	Włączenie przerwań od nowych danych i wykrycia stuknięcia
CTRL_REG5	0x01	Routing pinów przerwań: nowe dane – PTC5, stuknięcie – PTD1
CTRL_REG1	0xC5	Output data rate – 800 Hz, zmniejszony szum i wejście w tryb active

5 Opis realizacji i działania

Z uwagi na zatrważającą ilość problemów podczas uruchamiania czujnika MPU 6050 zdecydowaliśmy się wykorzystać akcelerometr MMA8451Q umieszczony na płytce mikroprocesora. Mimo rozdzielczości niższej o 2 bity efekty pracy były bardzo zadowalające a działanie płynne, dodatkowy pin przerwań pozwolił znacznie uprościć program.

By nie wgłębiać się zbytnio w logikę sterowania którą można prosto rozbudować o dodatkowe puknięcia przyjęliśmy, że do potrzeb prezentacji wystarczy rozróżnianie między kierunkami oraz pojedynczym a podwójnym puknięciem w czujnik. Aby zademonstrować poprawność odczytu przypisaliśmy do stuknięć w osi X przełączanie stanu diod, w osi Z pojedyncze odpowiada za toggle obu diod a podwójne zeruje liczniki oraz generuje prostą sekwencję mrugnięć.

6 Doświadczenie

Wykonując ten projekt nauczyliśmy się następujących rzeczy:

- Analizator stanów logicznych jest niezwykle przydatny w debugowaniu urządzeń cyfrowych.
- Mimo pozornej prostoty protokołu I2C wykorzystanie go w praktyce bywa niezwykle problematyczne.
- Flaga Transfer Complete rejestru statusowego I2C wcale nie jest ustawiana po zakończeniu transferu.