#include <SPI.h>

#include <AS3935.h>

#include <SoftwareSerial.h>

#define LCD\_pin 5 // pin do którego podlaczony jest wyswietlacz

#define CS\_pin 3 // pin do którego podlaczona jest linia CS detektora

#define IRQ\_pin 2 // pin do którego podlaczone jest przerwanie z detektora

const int NoDetect**=**30**;**

int counter**;**

int NumDisturber**=**0**;**

SoftwareSerial LCD **=** SoftwareSerial**(**0**,** LCD\_pin**);**

void printAS3935Registers**();**

// Prototyp funkcji zapewniajacej komunikacje poprzez SPI z ukladem

// AS3935. Zdefiniowana jest ponizej. Funkcja wymagana jest przez biblioteke

// ukladu do komunikacji. To uzytkownik musi ja zaimplementowac, dzieki czemu

// biblioteka dla AS3935 jest w duzej mierze agnostyczna dla sprzetu.

// Biblioteka wymaga funkcji o dokladnie takiej sygnaturze, nie moze to byc

// metoda zadnej klasy w C++, jak jest z wiekszoscia funkcji w bibliotekach Arduino.

// Upewnij sie, ze funkcja ta nie uzywa pinu CS - biblioteka zajmuje sie nim sama.

byte SPItransfer**(**byte sendByte**);**

int tunecap**;** // zmienna dla wartosci rejestru kalibracyjnego

// Obsluga przerwania dla AS3935. Zmienne definiowane w przerwaniu musza

// posiadac slowo kluczowe volatile, inaczej program moze nie skompilowac

// sie poprawnie z uwagi na optymalizacje kodu.

void AS3935Irq**();**

// Pierwszy parametr - funkcja do transferu przez SPI transfer function,

// Drugi parametr - pin Arduino, wykorzystany jako linia CS

// Trzeci parametr - pin Arduino wykorzystany jako linia IRQ.

// Dobrze jest wykorzystac do przerwania pin, który predefiniowano do takiego zastosowania,

// dzieki temu mozliwe jest wykorzystanie wbudowanej funkcji attachInterrupt

// w szkicu, do wykrywania przerwania. Biblioteka wewnetrznie sprawdza wartosc tego pinu

// podczas kalibracji.

AS3935 AS3935**(**SPItransfer**,**CS\_pin**,**IRQ\_pin**);**

void setup**(){**

serLCDInit**();**

backlightOn**();**

clearLCD**();**

lcdPosition**(**0**,**3**);**

LCD**.**print**(**"Lightining"**);**

lcdPosition**(**1**,**5**);**

LCD**.**print**(**"Sensor"**);**

delay**(**1500**);**

clearLCD**();**

Serial**.**begin**(**9600**);**

// Najpierw uruchamiamy SPI, potem ustawiamy parametey

SPI**.**begin**();**

// uklad korzysta z trybu SPI MODE1

SPI**.**setDataMode**(**SPI\_MODE1**);**

// Maksymalny zegar SPI dla tego ukladu 2 MHz,

// ale nigdy nie ustawiajmy zegara na 500 kHz

// gdyz na tej czestotliwosci dziala detektor

// wyladowan i prowadzic moze to do interferencji.

SPI**.**setClockDivider**(**SPI\_CLOCK\_DIV16**);**

// Uklad wysyla najpierw bit MSB.

SPI**.**setBitOrder**(**MSBFIRST**);**

// Zresetujmy wszystkie wewnetrzne rejestry do stanu domyslnmego.

AS3935**.**reset**();**

delay**(**10**);**

AS3935**.**setOutdoors**();**

// Zapisujemy wartosc 2 w rejestrze odpowiedzialnym za poziom szumów

AS3935**.**registerWrite**(**AS3935\_NF\_LEV**,**2**);**

// i uruchamiamy kalibracje.

// Jesli uklad nie bedzie mógl zestroic obwodu odbiorczego z wymagana tolerancja, to

// funkcja kalibracyjna zwróci wartosc falszywa. Inaczej zwróci true.

**if(!**AS3935**.**calibrate**())**

Serial**.**println**(**"Strojenie sie nie powiodlo. Sprawdz okablowanie i sensor."**);**

// Teraz mozemy pobrac i wyswietlic wartosc rejestru kalibracyjnego TUN\_CAP

// który bedzie w zakresie od 0 do 15.

tunecap**=**AS3935**.**registerRead**(**AS3935\_TUN\_CAP**);** // Wewnetrzna kalibracja.

Serial**.**print**(**"Rejestr kondensatora strojacego wynosi "**);**

Serial**.**println**(**tunecap**);**

// Mozemy takze poinformowac AS3935, ze znajdujemy sie na dworzu. Odkomentujmy kolejna linie

// AS3935.setOutdoors();

// Dalej musimy uruchomic wykrywanie wyladowan atmosferycznych.

AS3935**.**enableDisturbers**();**

// roaz wypisac wartosc rejestrów konbfiguracyjnych.

printAS3935Registers**();**

attachInterrupt**(**0**,**AS3935Irq**,**RISING**);**

**}**

void loop**(){**

// W glównej petli sprawdzamy, czy przerwabue zostalo wyzwolone. Jest to troche

// kontrproduktywne - nie tak powinny dzialac przerwania, ale tak jest zrealizowane to w Arduino.

// Normalnie mozna by oprzec sie na sprzetowym przerwaniu, a caly uklad uspic na czas

// oczekiwania na przerwanie, aby zredukowac pobór mocy.

// Wiekszosc kodu obslugi znajduje sie w funkcji obslugujace przerwanie,

// w glównej petli programu piszemy jedynie o oczekiwaniu na wyzwolenie,

// aby upewnic sie, ze program dziala czaly czas poprawnie.

delay**(**1000**);**

Serial**.**println**(**"Oczekiwanie..."**);**

**if** **(**counter**==**0**){**

NumDisturber**=**0**;**

counter**=**NoDetect**;**

clearLCD**();**

lcdPosition**(**0**,**1**);**

LCD**.**print**(**"Brak wyladowania"**);**

lcdPosition**(**1**,**4**);**

LCD**.**print**(**"wykryto"**);**

**}** **else** **{**

counter**=**counter **-** 1**;**

**}**

**}**

void printAS3935Registers**(){**

int noiseFloor **=** AS3935**.**getNoiseFloor**();**

int spikeRejection **=** AS3935**.**getSpikeRejection**();**

int watchdogThreshold **=** AS3935**.**getWatchdogThreshold**();**

Serial**.**print**(**"Poziom szumu wynosi: "**);**

Serial**.**println**(**noiseFloor**,**DEC**);**

Serial**.**print**(**"Poziom odrzucania szpilek wynosi: "**);**

Serial**.**println**(**spikeRejection**,**DEC**);**

Serial**.**print**(**"Próg watchdoga wynosi: "**);**

Serial**.**println**(**watchdogThreshold**,**DEC**);**

**}**

// To pelna implementacja funkcji do transferu poprzez SPI, która podawana jest do

// biblioteki AS3935. Mozna to zmienic na wlasna implementacje komunikacji.

byte SPItransfer**(**byte sendByte**){**

**return** SPI**.**transfer**(**sendByte**);**

**}**

// To jest funkcja obslugujaca przerwanie z AS3935. Nic nie zwraca i nie przyjmuje

// zadnych argumentów. Kod tutaj powinien byc krótki i wykonywac sie szybko.

void AS3935Irq**(){**

// Caly kod, odpowiedzialny za wyswietlanie informacji na LCD przesuniety zostal do przerwania.

// Pierwszym krokiem jest sprawdzenie co jest zródlem przerwania, jak tylko informacja o nim

// dojdzie. Po odczytaniu rejestru przerwania pin IRQ przejdzie w stan niski.

int irqSource **=** AS3935**.**interruptSource**();**

// Wartosc ta definiuje co jest zródlem przerwania:

// 0 - zbyt wysoki poziom szumu

// 2 - wykryto zaklócenie

// 3 - wykryto wyladowanie atmosferyczne.

**if** **(**irqSource **&** 0b0001**)** Serial**.**println**(**"Poziom szumu zbyt wysoki, spróbuj zmienic ustawienie poziomu szumu."**);**

**if** **(**irqSource **&** 0b0100**){**

NumDisturber**+=**1**;**

Serial**.**println**(**"Wykryto zaklócenie"**);**

clearLCD**();**

lcdPosition**(**0**,**0**);**

LCD**.**print**(**"Liczba wyladowan: "**);**

LCD**.**print**(**NumDisturber**,**DEC**);**

counter**=**NoDetect**;**

**}**

**if** **(**irqSource **&** 0b1000**){**

// W tym przypadku uzyskac musimy nie tylko informacje o tym, ze wykryto wyladowanie, a takze

// ile ich wykryto, ale takze o odleglosci od wyladowania w kilometrach, gdzie 1 to burza

// ponizej minimalnej odleglosci detekcji, a 63 to wyladowanie powyzehj granicy. Wszystkie

// pozostale wartosci, to odleglosc w kilometrach.

int strokeDistance **=** AS3935**.**lightningDistanceKm**();**

**if** **(**strokeDistance **==** 1**){**

Serial**.**println**(**"Burza znajduje sie nad Toba, uwazaj!"**);**

lcdPosition**(**1**,**1**);**

Serial**.**println**(**"Burza nad Toba"**);**

lcdPosition**(**1**,**3**);**

Serial**.**println**(**"UWAZAJ!"**);**

counter**=**NoDetect**;**

**}**

**if** **(**strokeDistance **==** 63**){**

Serial**.**println**(**"Wykryto wyladowanie poza zasiegiem detekcji."**);**

lcdPosition**(**0**,**2**);**

Serial**.**println**(**"Poza zasiegiem"**);**

lcdPosition**(**1**,**0**);**

Serial**.**println**(**"wykryto wyladowanie"**);**

counter**=**NoDetect**;**

**}**

**if** **(**strokeDistance **<** 63 **&&** strokeDistance **>** 1**){**

Serial**.**print**(**"Wykryto wyladowanie w odleglosci "**);**

Serial**.**print**(**strokeDistance**,**DEC**);**

Serial**.**println**(**" kilometrów."**);**

lcdPosition**(**1**,**0**);**

LCD**.**print**(**"Odleglosc: "**);**

LCD**.**print**(**strokeDistance**,**DEC**);**

LCD**.**print**(**"km"**);**

counter**=**NoDetect**;**

**}**

**}**

**}**