Temat projektu i wymagania szczegółowe

Układ z czujnikiem wykrywającym przechylenie pojazdu w dwóch osiach (X, Y), co można zauważyć na podpiętym ekranie po poruszeniu układem. W przypadku przechylenia pojazdu słychać buzzer, który przyspiesza wraz z coraz większym kątem przechylenia. Układ posiada przycisk do wyboru początkowej wartości przechylenia przydatnej przy postoju na nierównym terenie(wzniesienia, krawężniki, nierówna nawierzchnia).

Analiza rozwiązania

Rozwiązanie można było oprzeć na dowolnym mikrokontrolerze. Popularnym rozwiązaniem jest użycie mikrokontrolerów z rodziny ATmega, ATtiny, bądź rodziny ESP. Do wykonania projektu użyto mikrokontrolera AVR ATmega328P-U DIP ze względu na niski pobór energii, niską cenę, dostępność oraz dużą ilość pomocnych informacji w internecie wynikających z implementacji tego mikrokontrolera w modelach Arduino.

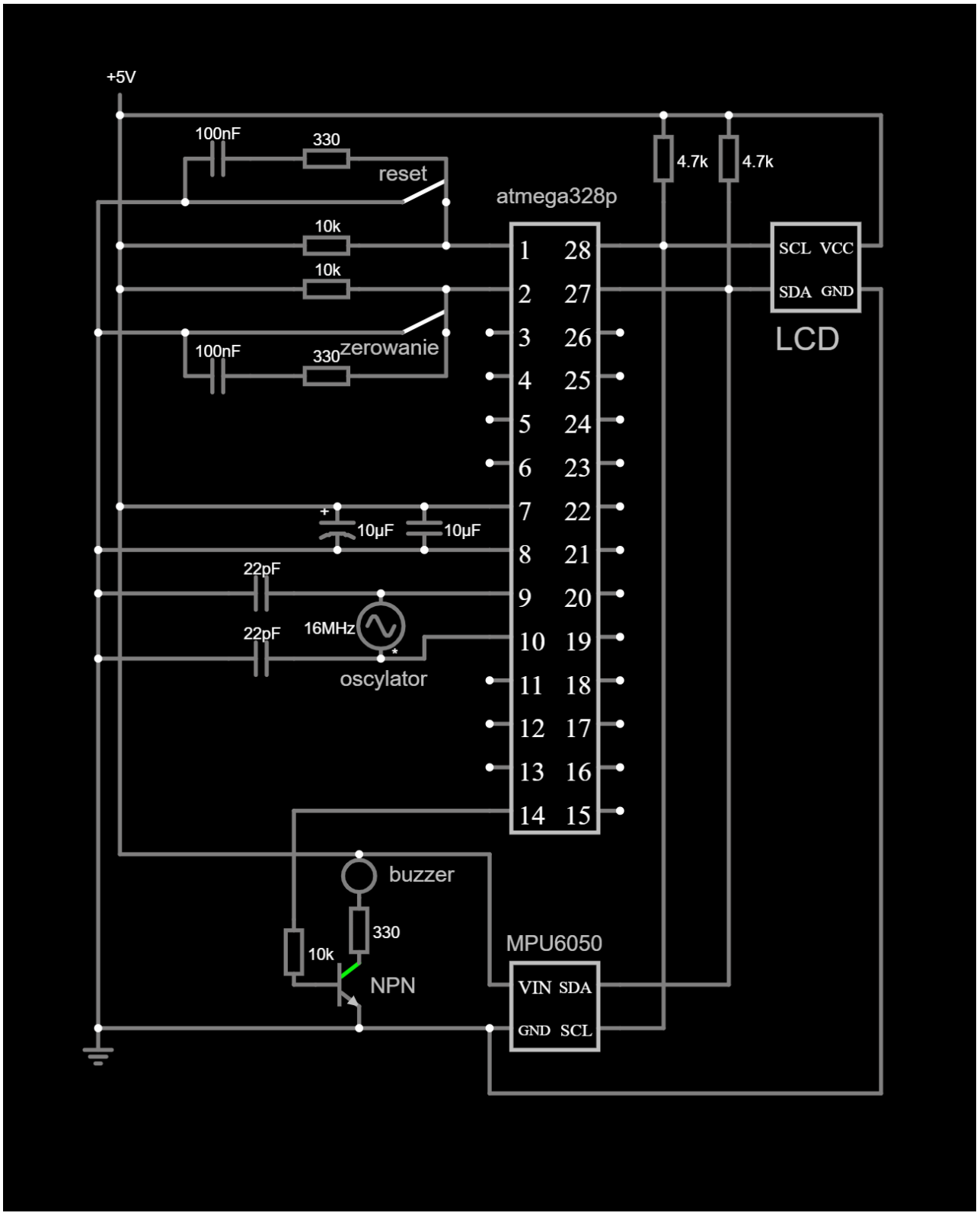
Do pomiaru przechyleń pojazdu dostępna jest cała gama różnych czujników. Początkowo do pomiaru przechylenia pojazdu zakupiono zwykły czujnik przechylenia, wykrywający jedynie przechylenie, bądź jego brak(działanie zerojedynkowe). Po dalszej analizie okazało się, że tego typu czujnik może być niewystarczający. Potrzebnymi informacjami okazał się konkretny kąt oraz oś przechylenia. Użyto akcelerometru, który umożliwia uzyskanie dokładniejszych informacji.

Układ jest zasilany poprzez microUSB, ponieważ umożliwia ono zasilanie 5V, które jest potrzebne do zasilenia mikrokontrolera ATmega328, akcelerometru oraz wyświetlacza. Zrezygnowano z opcji zasilania bateryjnego, do którego potrzebna by była przetwornica, a także baterie. Takie rozwiązanie wydaje się być mało optymalne, ponieważ wewnątrz pojazdu bardzo łatwo można podłączyć zasilanie przez przewody USB.

Do zasygnalizowania przechylenia pojazdu użyto najprostszego buzzera, który jedynie sygnalizuje przechylenie dźwiękowo w małej skali. Przy użyciu w konkretnym pojeździe można by podpiąć układ do alarmu.

Przechylenie również można zauważyć na ekranie wyświetlacza LCD, który jest podpięty jako możliwość dodatkowego dokładnego sprawdzenia aktualnego przechylenia układu.

Schemat ideowy układu

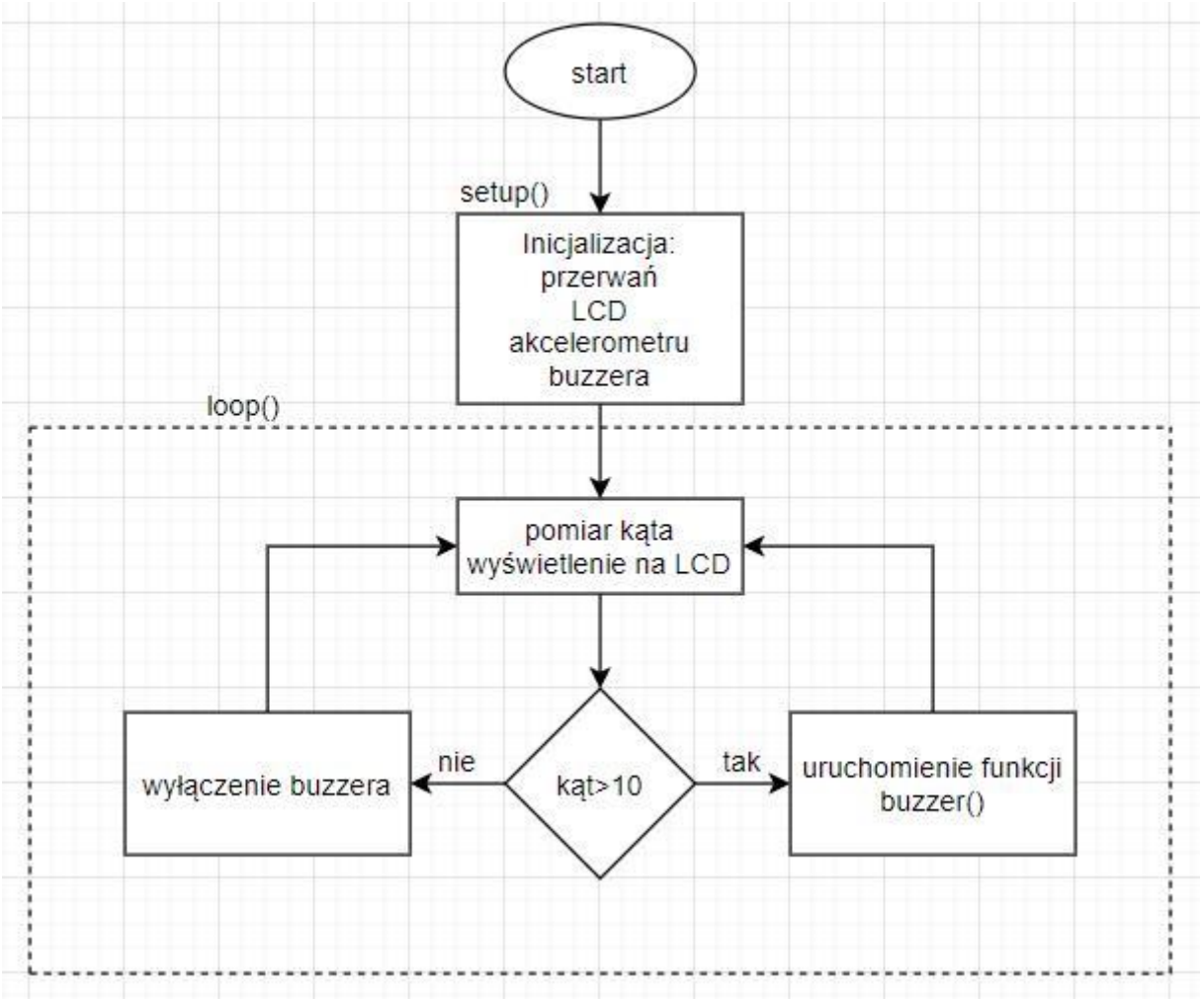


Lista użytych elementów

- 1x Tranzystor bipolarny NPN 2N3904 40V/0,2A
- 1x MPU-6050 3-osiowy akcelerometr i żyroskop I2C - moduł DFRobot
- 2x Tact Switch 6x6mm / 4,3mm THT
- 1x Mikrokontroler AVR - ATmega328P-U DIP
- 1x Buzzer z generatorem 5V 12mm - THT
- 1x Rezonator kwarcowy 16MHz - HC49 - niski
- 1x Wyświetlacz LCD 2x16 znaków niebieski + konwerter I2C LCM1602

- 1x Moduł z gniazdem microUSB
- 3x rezystor 10kΩ
- 2x rezystor 4.7kΩ
- 3x rezystor 330Ω
- 2x kondensator ceramiczny 100nF
- 1x kondensator ceramiczny 10nF
- 2x kondensator ceramiczny 22pF
- 1x kondensator elektrolityczny 10uF

Schemat blokowy programu



Zastosowane rozwiązania programowe

Działanie programu oparto na osobnych funkcjach wykonujących podprogramy. Użyto również funkcji wbudowanych oraz funkcji z zewnętrznych bibliotek. Użyte biblioteki: *LiquidCrystal_I2C.h*, *Wire.h*, *Math.h*, *MPU6050.h*.

Kod programu (fragment)

Domyślne wartości *aktualnyCzas* i *aktualnyCzas2* wynoszą *0* i oznaczają gotowość do pomiaru czasu.

```
void buzzer(){ //funkcja realizująca sygnały dźwiękowe
  buzzF = (1/maxAngle)*10000; //wyznaczenie okresu między sygnałami
  if(aktualnyCzas == 0) aktualnyCzas = millis(); //pomiar czasu do okresu między sygnałami

  if(aktualnyCzas2 != 0 && aktualnyCzas2+200<millis()){ //sprawdzenie czy dokonano pomiaru i czy minęło 200ms włączenia buzzera
    digitalWrite(8, LOW); //wyłączenie buzzera
    aktualnyCzas = 0; //koniec realizacji cyklu dzwonienia
    aktualnyCzas2 = 0; //zerowanie zmiennych, przygotowanie do kolejnego cyklu
 }else if(aktualnyCzas > 0 && aktualnyCzas+buzzF<millis()){ //sprawdzenie czy dokonano pomiaru czasu (aktualnyCzas>0) i czy minął wyznaczony okres między sygnałami (aktualnyCzas+buzzF)
    digitalWrite(8, HIGH); //włączenie buzzera
    aktualnyCzas = -1; //zmienna niegotowa, buzzer wydaje dźwięk
    aktualnyCzas2 = millis(); //pomiar czasu potrzebny do odmierzenia 200ms włączenia buzzera
  }
}
```

Działanie układu

Układ, po podpięciu do zasilania 5V przewodem microUSB uruchamia się samoczynnie. Kąt nachylenia jest wyliczany na podstawie wskazań akcelerometru ze wzorów:

$$\sphericalangle x = \arctan\left(\frac{a_x}{\sqrt{a_y^2 + a_z^2}}\right)$$

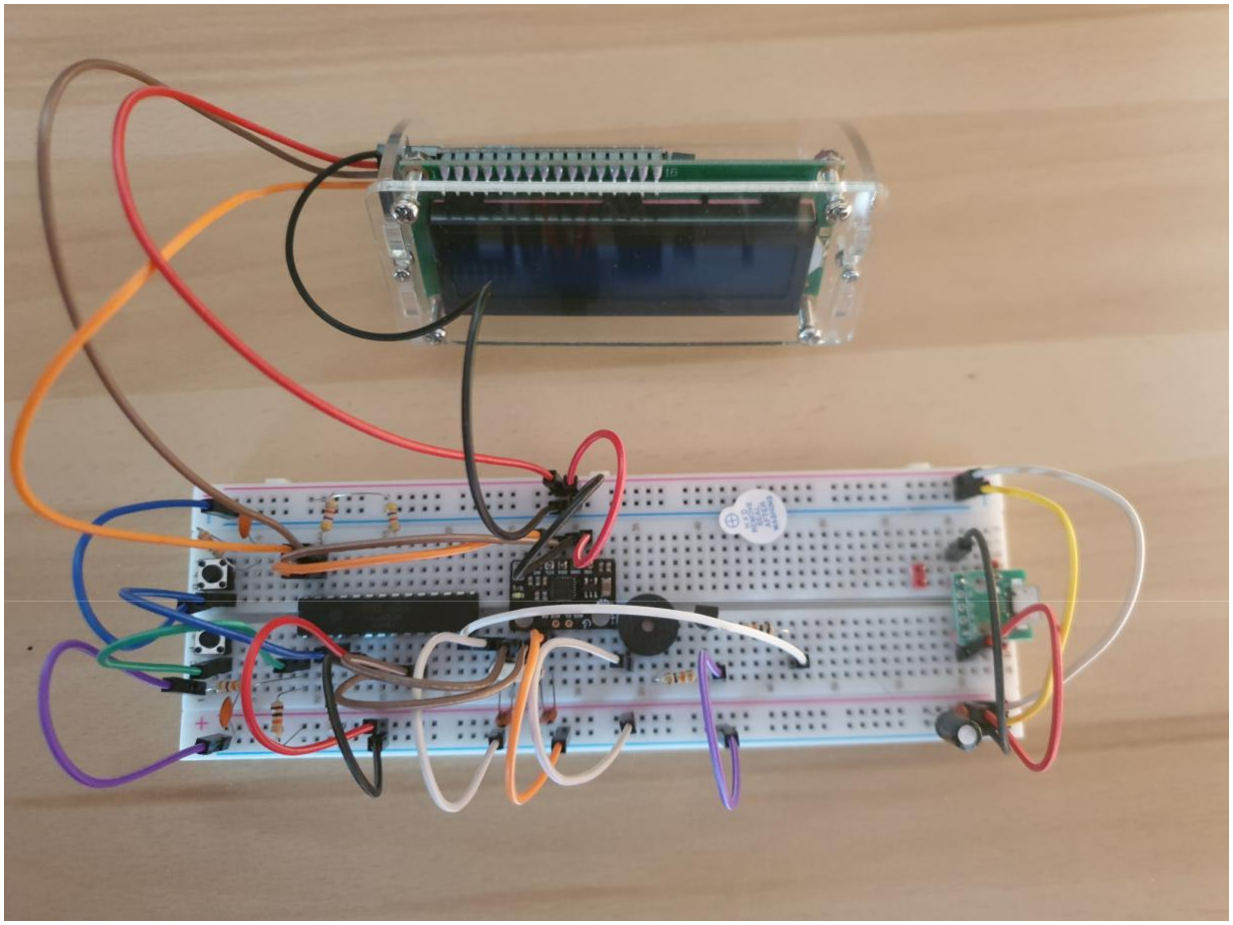
$$\sphericalangle y = \arctan\left(\frac{a_y}{\sqrt{a_x^2 + a_z^2}}\right)$$

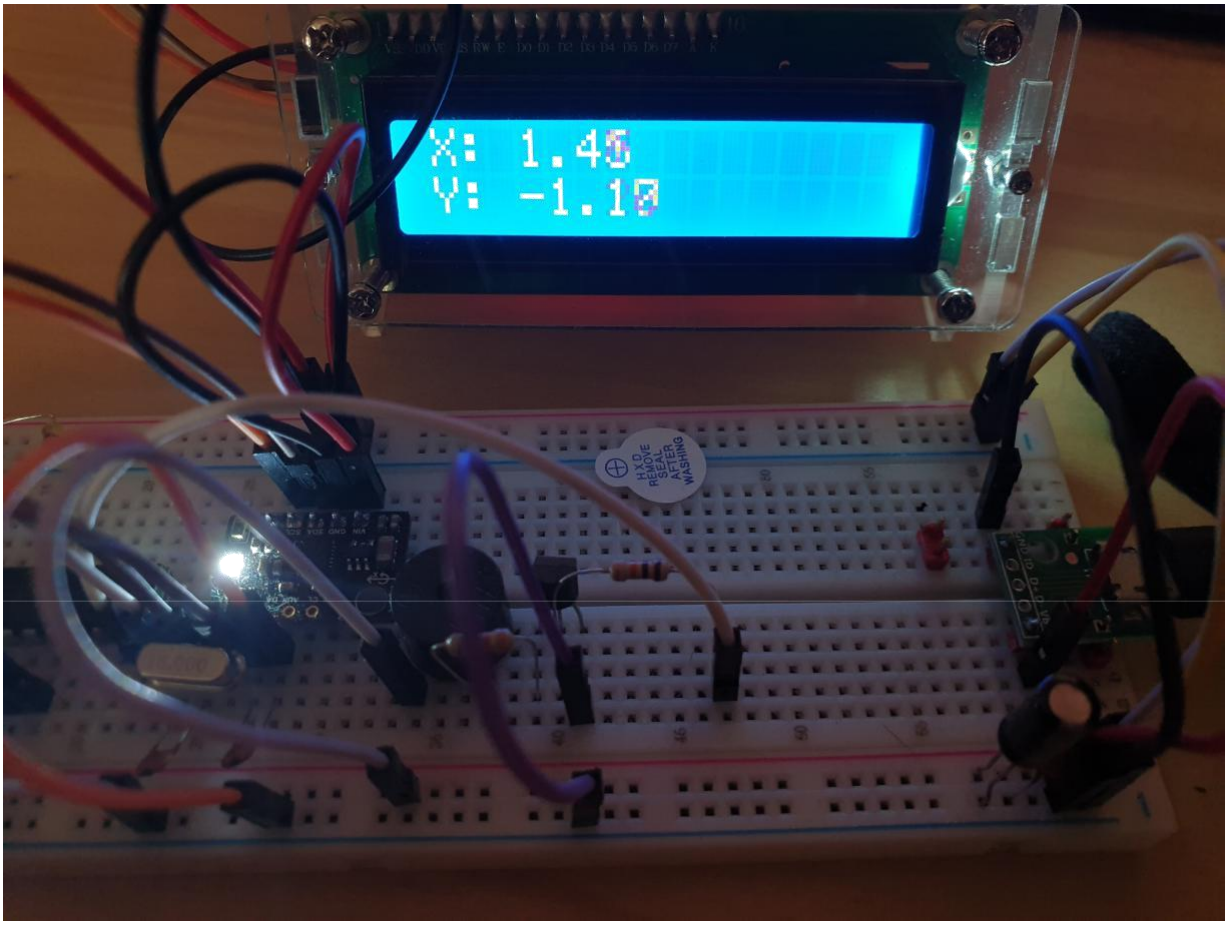
Mikrokontroler wylicza średnią arytmetyczną ze 100 pomiarów i wyświetla wynik na wyświetlaczu LCD 16x2. Przy przechyleniu >10^o buzzer zaczyna wydawać dźwięki trwające 200ms. Czas między kolejnymi sygnałami dźwiękowymi wyznaczany jest ze wzoru:

$$buzzT = \left(\frac{1}{\max(\sphericalangle x, \sphericalangle y)}\right) * 10000[ms]$$

Im większe nachylenie, tym szybciej następują kolejne sygnały. Układ został dokładnie przetestowany i reaguje poprawnie na zmiany w obu osiach oraz zmianę domyślnego ustawienia(kąta). Zakres pomiarów wynosi 0^o-90^o. Po przekroczeniu 90^o wartości maleją ponownie do 0^o.

Wygląd układu





Wnioski

Projekt został wykonany zgodnie z początkowymi założeniami i działa poprawnie. Najwięcej problemów przysporzył mikrokontroler ATmega328P, który podpięty do komputera uniemożliwiał wgranie programu przez błąd wykrywania sterowników. Układ jest możliwy do użycia w pojeździe poprzez zamontowanie go na stałe w jednym miejscu i podpięcie do zasilania. Dalszy rozwój projektu mógłby być przeprowadzony poprzez użycie alarmu pojazdu jako sygnału dźwiękowego, a także automatycznego ustawiania domyślnej wartości przechylenia pojazdu po wygaśnięciu silnika.

Literatura

http://home.agh.edu.pl/~bartus/index.php?action=efekty&subaction=arduino&item=21
https://forbot.pl/blog/kursy

Projekt

Projekt jest dostępny na platformie GitHub:
https://github.com/blackbird998/Czujnik-nachylenia