# Dokumentacja techniczno- ruchowa

# Pojazd autonomiczny A-Cars Mk 1

# Tor przeszkód

**Spis treści**

Spis treści

[Dane techniczne 3](#_Toc51368564)

[Kod źródłowy 4](#_Toc51368565)

[Schemat ideowy 9](#_Toc51368566)

[Problemy uruchomieniowe 10](#_Toc51368567)

[Projekt zderzaków 11](#_Toc51368568)

[Projekt chassis 12](#_Toc51368569)

# Dane techniczne

Zasilanie: 3 ogniwa 18650  
Pobór prądu: 3A  
Jednostka centralna: Arduino Uno Rev3  
Układ wykonawczy: Forbot RoboShield  
Napęd: 4WD DC  
Prędkość: 10 km/h  
Światła: 12x LED RGB WS2812B  
Zderzaki: Drukowane 3D, ekologiczne PLA  
Czujnik: HC-SR04 ultradźwiękowy  
Napęd czujnika: Serwomechanizm SG90  
Moduł dźwiękowy: DFPlayer Mini

# Kod źródłowy

#include <Adafruit\_NeoPixel.h>

#include <Servo.h>

#define LED\_PIN 7

#define LED\_COUNT 14

Adafruit\_NeoPixel strip**(**LED\_COUNT**,** LED\_PIN**,** NEO\_GRB **+** NEO\_KHZ800**);**

#define L\_PWM 5

#define L\_DIR 4

#define R\_PWM 6

#define R\_DIR 9

Servo Oczy**;**

int Trig **=** 13**;** // pin 13 Arduino połączony z pinem Trigger czujnika

int Echo **=** 12**;** // pin 12 Arduino połączony z pinem Echo czujnika

int CM**;** // odległość w cm

long CZAS**;** // długość powrotnego impulsu w uS

int center **=** 90**;** // Tutaj przetrzymujemy sobie pozycję serwa "na wprost"

int lewo **=** 180**;** // Tutaj przetrzymujemy sobie pozycję serwa "na lewo"

int prawo **=** 0**;** // Tutaj przetrzymujemy sobie pozycję serwa "na prawo"

int odlprawo **=** 0**;** // Odległość na prawo

int odllewo **=** 0**;** // Odległość na lewo

int odljazda **=** 50**;** // Odległość do przodu

int vmax **=** 105**;** // Prędkość pojazdu

int tobr **=** 300**;** // Czas obrotu pojazdu

int pos **=** 90**;** // Zmienna pozycji 0 serwa

void setup**()**

**{**

strip**.**begin**();** // INITIALIZE NeoPixel strip object (REQUIRED)

strip**.**show**();** // Pokaż stan pixeli

strip**.**setBrightness**(**250**);** // Ustaw jasność pixeli na 250 (max = 255)

pinMode**(**8**,** OUTPUT**);** // Ustaw pin A1 jako wyjście

digitalWrite**(**8**,** HIGH**);** // Ustaw stan pinu A1 na wysoki

Oczy**.**attach**(**11**);** // Do którego pinu podpięliśmy serwo? Ano do 11

Oczy**.**write**(**center**);** // Od razu ustawiamy serwo w pozycji "na wprost"

Serial**.**begin**(**9600**);** // inicjalizaja monitora szeregowego

pinMode**(**Trig**,** OUTPUT**);** // ustawienie pinu 2 w Arduino jako wyjście

pinMode**(**Echo**,** INPUT**);** // ustawienie pinu 3 w Arduino jako wejście

Serial**.**println**(**"Test czujnika odleglosci"**);**

**for(**int i**=**0**;** i**<**strip**.**numPixels**();** i**++)** **{** // Dla każdego pixela...

strip**.**setPixelColor**(**i**,** 0**,**0**,**255**);** // Ustaw kolor pixeli

strip**.**show**();** // Pokaż stan pixeli

delay**(**50**);** // Zatrzymaj się na 50ms

**}**

**for(**int i**=**0**;** i**<**strip**.**numPixels**();** i**++)** **{** // Dla każdego pixela...

strip**.**setPixelColor**(**i**,** 0**,**255**,**0**);** // Ustaw kolor pixeli

strip**.**show**();** // Pokaż stan pixeli

delay**(**40**);** // Zatrzymaj się na 50ms

**}**

**for(**int i**=**0**;** i**<**strip**.**numPixels**();** i**++)** **{** // Dla każdego pixela...

strip**.**setPixelColor**(**i**,** 255**,**0**,**0**);** // Ustaw kolor pixeli

strip**.**show**();** // Pokaż stan pixeli

delay**(**30**);** // Zatrzymaj się na 30ms

**}**

digitalWrite**(**L\_DIR**,** 0**);** // Ustawienie kierunku 'przód' na lewych kołach

digitalWrite**(**R\_DIR**,** 0**);** // Ustawienie kierunku 'przód' na prawych kołach

analogWrite**(**L\_PWM**,** vmax**);** // Ustawienie predkosci na lewych kołach

analogWrite**(**R\_PWM**,** vmax**);** // Ustawienie predkosci na prawych kołach

**}**

void loop**()**

**{**

**for(** pos **=** 180**;** pos **>=** 0 **;**pos **-=**30**){** //

Oczy**.**write**(**pos**);** //

odljazda **=** pomiar\_odleglosci**();** // Pomiar odległości

**if** **(**odljazda **<=** 30**){** // Jeżeli odległość jest mniejsza od 30cm...

analogWrite**(**L\_PWM**,**0**);** // Koła stop

analogWrite**(**R\_PWM**,**0**);** // Koła stop

delay**(**200**);**

rozgladaj**();** // Rozglądaj się

**if(**odllewo**>**odlprawo**){** // Jeżeli odległość z lewej jest większa od prawej...

obrotL**();}** // Obróć się w lewo

**else** **{**

obrotR**();**

**}**

// Obróć się w prawo

**}**

Serial**.**println**(**odllewo**);** // Pokaż odległość w lewo (Wysyłka danych przez port szeregowy)

Serial**.**println**(**odlprawo**);** // Pokaż odległość w prawo (Wysyłka danych przez port szeregowy)

delay**(**40**);** // Zatrzymaj się na 40ms

**}**

**for(** pos **=** 0**;** pos **<=** 180 **;**pos **+=**30**){** //

Oczy**.**write**(**pos**);** //

odljazda **=** pomiar\_odleglosci**();** // Pomiar odległości

**if** **(**odljazda **<=** 30**){** // Jeżeli odległość jest mniejsza od 30cm...

analogWrite**(**L\_PWM**,**0**);** // Koła stop

analogWrite**(**R\_PWM**,**0**);** // Koła stop

delay**(**200**);**

rozgladaj**();** // Rozglądaj się

**if(**odllewo**>**odlprawo**){** // Jeżeli odległość z lewej jest większa od prawej...

obrotL**();}** // Obróć się w lewo

**else** **{**

obrotR**();**

**}**

**}**

Serial**.**println**(**odllewo**);** // Pokaż odległość w lewo (Wysyłka danych przez port szeregowy)

Serial**.**println**(**odlprawo**);** // Pokaż odległość w prawo (Wysyłka danych przez port szeregowy)

delay**(**40**);** // Zatrzymaj się na 40ms

**}**

**}**

void jedzprosto**(){** // Funkcja jazdy prosto

digitalWrite**(**L\_DIR**,** 0**);** // Ustawienie kierunku 'przód' na lewych kołach

digitalWrite**(**R\_DIR**,** 0**);** // Ustawienie kierunku 'przód' na prawych kołach

analogWrite**(**L\_PWM**,** vmax**);** // Ustawienie predkosci na lewych kołach

analogWrite**(**R\_PWM**,** vmax**);** // Ustawienie predkosci na prawych kołach

**}**

int pomiar\_odleglosci **()** // Funckja pomiaru odległości

**{**

digitalWrite**(**Trig**,** LOW**);** // Ustawienie stanu wysokiego na 2 uS - impuls inicjalizujacy - patrz dokumentacja

delayMicroseconds**(**2**);**

digitalWrite**(**Trig**,** HIGH**);** // Ustawienie stanu wysokiego na 15 uS - impuls inicjalizujacy - patrz dokumentacja

delayMicroseconds**(**15**);**

digitalWrite**(**Trig**,** LOW**);**

digitalWrite**(**Echo**,** HIGH**);**

CZAS **=** pulseIn**(**Echo**,** HIGH**);**

CM **=** CZAS **/** 58**;**

**return** CM**;** // szerokość odbitego impulsu w uS podzielone przez 58 to odleglosc w cm - patrz dokumentacja

**}**

void rozgladaj**(){** // Funkcja rozglądania się

delay**(**200**);**

digitalWrite**(**8**,** LOW**);** // Ustaw stan pinu A1 na niski

delay**(**200**);** // Zatrzymaj się na 200ms

digitalWrite**(**8**,** HIGH**);** // Ustaw stan pinu A1 na wysoki

int R **=** random**(**0**,**255**);** // Losuj natężenie koloru czerwonego (od 0 do 255)

int G **=** random**(**0**,**255**);** // Losuj natężenie koloru zielonego (od 0 do 255)

int B **=** random**(**0**,**255**);** // Losuj natężenie koloru niebieskiego (od 0 do 255)

**for(**int i**=**0**;** i**<**strip**.**numPixels**();** i**++)** **{** // Dla każdego pixela...

strip**.**setPixelColor**(**i**,** R**,**G**,**B**);** // Set pixel's color (in RAM)

strip**.**show**();** // Update strip to match

delay**(**50**);** **}** // Pause for a moment

analogWrite**(**L\_PWM**,** 0**);** // Ustawienie predkosci na lewych kołach

analogWrite**(**R\_PWM**,** 0**);** // Ustawienie predkosci na prawych kołach

delay**(**200**);** // Zatrzymaj się na 200ms

digitalWrite**(**L\_DIR**,** 1**);** // Ustawienie kierunku 'przód' na lewych kołach

digitalWrite**(**R\_DIR**,** 1**);** // Ustawienie kierunku 'przód' na prawych kołach

analogWrite**(**L\_PWM**,** vmax**);** // Ustawienie predkosci na lewych kołach

analogWrite**(**R\_PWM**,** vmax**);** // Ustawienie predkosci na prawych kołach

delay**(**200**);** // Zatrzymaj się na 200ms

analogWrite**(**L\_PWM**,** 0**);** // Ustawienie predkosci na lewych kołach

analogWrite**(**R\_PWM**,** 0**);** // Ustawienie predkosci na prawych kołach

Oczy**.**write**(**lewo**);** // Popatrz się na lewo

delay**(**100**);**

odllewo **=** pomiar\_odleglosci**();** // Zmierz odległość w lewo

delay**(**400**);** // Zatrzymaj się na 400ms

Oczy**.**write**(**center**);** // Popatrz się prosto

delay**(**400**);** // Zatrzymaj się na 400ms

Oczy**.**write**(**prawo**);** // Popatrz się na prawo

delay**(**100**);**

odlprawo **=** pomiar\_odleglosci**();** // Zmierz odległość w prawo

delay**(**400**);** // Zatrzymaj się na 400ms

Oczy**.**write**(**center**);** // Popatrz się prosto...

pos **=** 90**;** // W pozycji 90\*

pomiar\_odleglosci**();** // Zmierz odległość

delay**(**200**);** // Zatrzymaj się na 200ms

**}**

void obrotL**(){** // Funkcja obrotu w lewo

Serial**.**println**(**"Jadę w lewo"**);**

digitalWrite**(**L\_DIR**,** 0**);** // Ustawienie kierunku 'tyl' na lewych kołach

digitalWrite**(**R\_DIR**,** 1**);** // Ustawienie kierunku 'przód' na prawych kołach

analogWrite**(**L\_PWM**,** 250**);** // Ustawienie predkosci na lewych kołach

analogWrite**(**R\_PWM**,** 250**);** // Ustawienie predkosci na prawych kołach

delay**(**tobr**);**

analogWrite**(**L\_PWM**,** 0**);** // Koła stop

analogWrite**(**R\_PWM**,** 0**);** // Koła stop

odllewo **=** 0**;**

odlprawo **=** 0**;**

delay**(**100**);**

digitalWrite**(**L\_DIR**,** 0**);** // Ustawienie kierunku 'przód' na lewych kołach

digitalWrite**(**R\_DIR**,** 0**);** // Ustawienie kierunku 'przód' na prawych kołach

analogWrite**(**L\_PWM**,** vmax**);** // Ustawienie predkosci na lewych kołach

analogWrite**(**R\_PWM**,** vmax**);** // Ustawienie predkosci na prawych kołach

**}**

void obrotR**(){** // Funkcja obrotu w prawo

Serial**.**println**(**"Jadę w prawo"**);**

digitalWrite**(**L\_DIR**,** 1**);** // Ustawienie kierunku 'przod' na lewych kołach

digitalWrite**(**R\_DIR**,** 0**);** // Ustawienie kierunku 'tyl' na prawych kołach

analogWrite**(**L\_PWM**,** 250**);** // Ustawienie predkosci na lewych kołach

analogWrite**(**R\_PWM**,** 250**);** // Ustawienie predkosci na prawych kołach

delay**(**tobr**);**

analogWrite**(**L\_PWM**,** 0**);** // Koła stop

analogWrite**(**R\_PWM**,** 0**);** // Koła stop

odllewo **=** 0**;**

odlprawo **=** 0**;**

delay**(**100**);**

digitalWrite**(**L\_DIR**,** 0**);** // Ustawienie kierunku 'przód' na lewych kołach

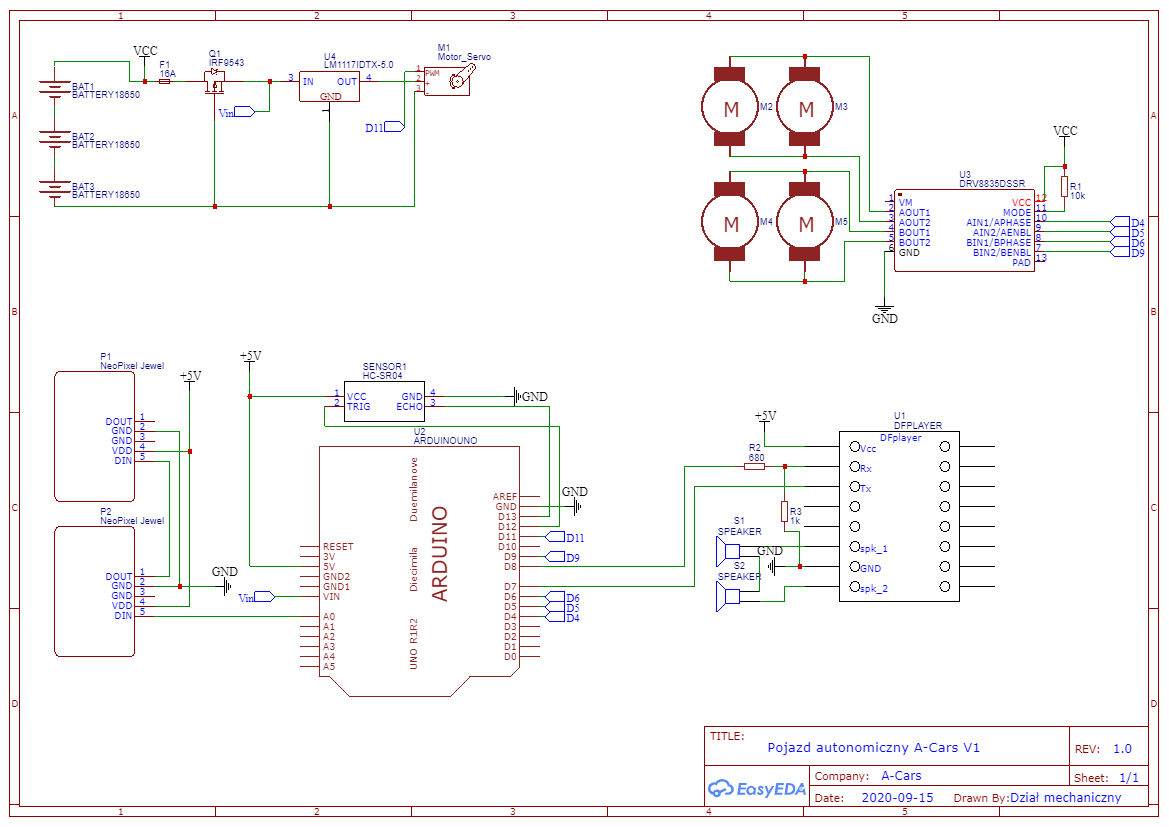
digitalWrite**(**R\_DIR**,** 0**);** // Ustawienie kierunku 'przód' na prawych kołach

analogWrite**(**L\_PWM**,** vmax**);** // Ustawienie predkosci na lewych kołach

analogWrite**(**R\_PWM**,** vmax**);** // Ustawienie predkosci na prawych kołach

**}**

# Schemat ideowy

****

# Problemy uruchomieniowe

W początkowej fazie projektu należało ustalić kierunki obrotów silników napędowych, aby później uprościć pracę z pisaniem funkcji ruchowych.

Niedokładny pomiar odległości- powodowany przez zbyt częste wywoływanie funkcji pomiaru, problem ten jest rodzajem paradoksu, ponieważ zmniejszenie częstości wykonywania pomiaru poprawiło ich dokładność.

Początkowo czujnik zamontowany na serwomechanizmie spoglądał odwrotnie do kierunku jazdy, należało poprawić mocowanie serwa.

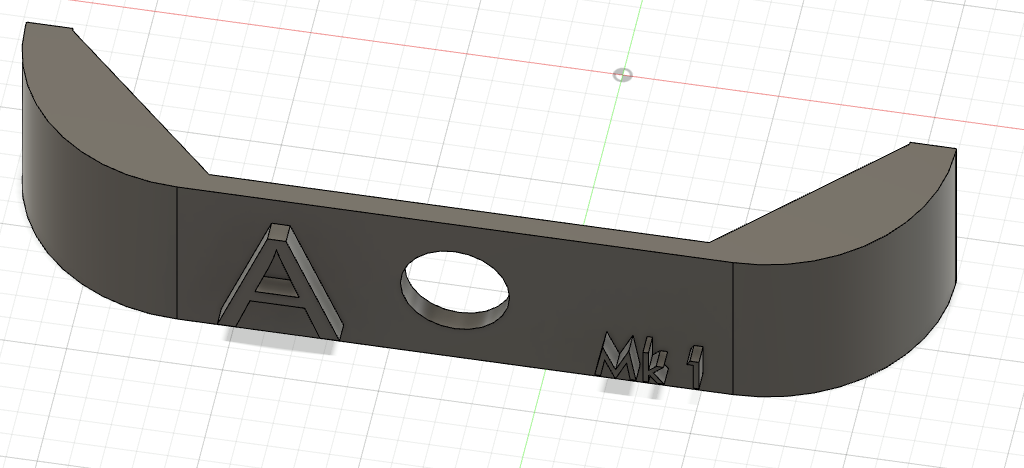
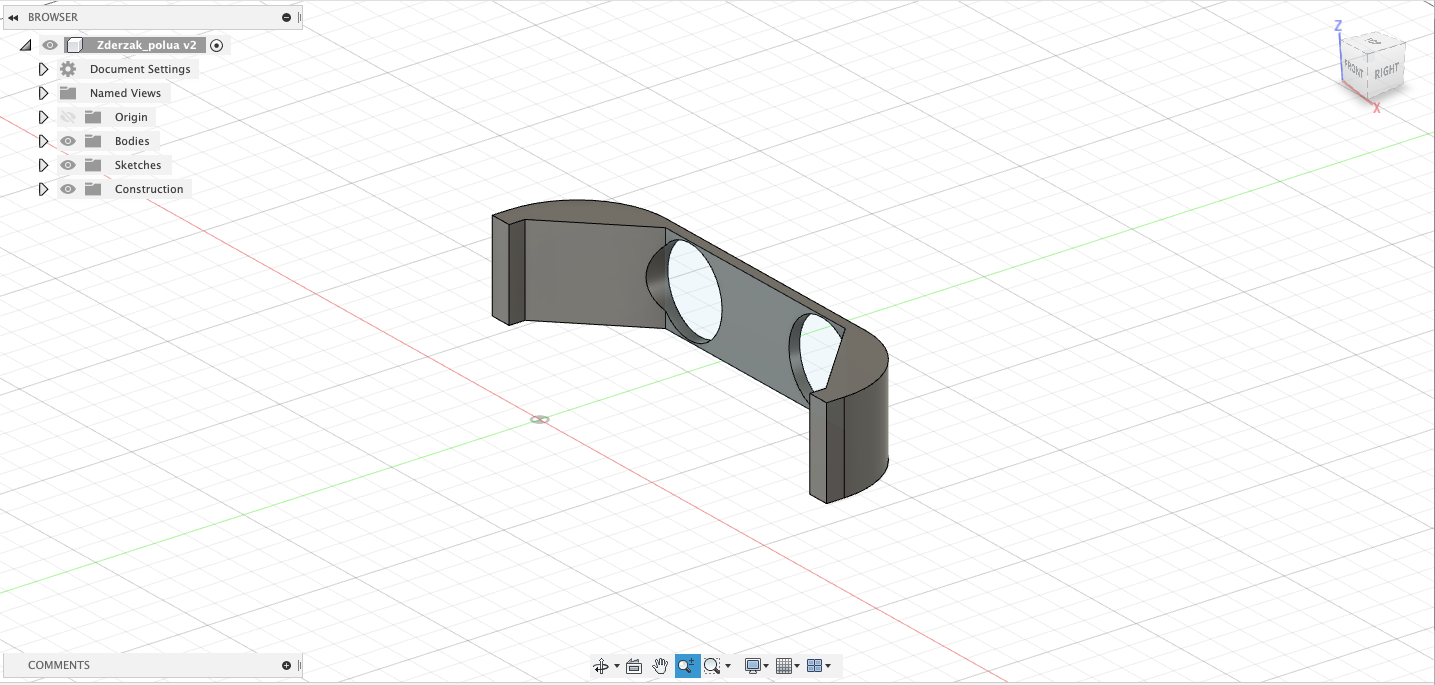
Moduł dźwiękowy dostosowany jest do pracy z napięciem 3.3V, natomiast Arduino pracuje w standardzie TTL. Celem zapewnienia kompatybilności zastosowano konwerter stanów logicznych.  
  
 Oświetlenie LED było zbyt prądożerne, problem rozwiązano poprzez zastosowanie przetwornicy typu StepDown. Zastosowanie typowego stabilizatora liniowego skracałoby czas pracy świateł do poziomu poniżej akceptowalnego. Przy stosowaniu zasilania bateryjnego konieczny jest dobór jak najwydajniejszych źródeł stabilnego napięcia.

Pojazd omijał przeszkody znajdujące się bezpośrednio przed nim, natomiast potrafił uderzyć w przeszkodę pod ukosem. Pod uwagę brano dwa rozwiązania, jednym z nich było zastosowanie dodatkowych czujników bocznych, jednak ze względu na to, iż byłoby to dodatkowe obciążenie dla baterii zdecydowano się zmodernizować sposób działania zamontowanego już czujnika HC-04SC. Po wykonaniu modernizacji pojazd stale się rozgląda, przez co ryzyko zderzenia się z przeszkodą zdecydowanie zmalało. W szczególnych sytuacjach zderzenie może nadal wystąpić.

Okazało się, że moduł DFPlayer ma uszkodzony port szeregowy, ze względu na ograniczenia czasowe w dostawie tego elementu zastosowano sterowanie bezpośrednie.

# Projekt zderzaków

Zderzaki zostały wykonane na wymiar. Zostały zaprojektowane przy pomocy narzędzi do modelowania 3D, po czym wydrukowane na drukarce 3D z biodegradowalnego PLA. W zderzakach widoczne są otwory montażowe na głośnik oraz lampy.



# Projekt chassis

 Chassis zostało wykonane z aluminium, jego wymiary to 170x 265. Otwory potrzebne do montażu elementów elektronicznych oraz napędowych wykonano po otrzymaniu zamówienia zgodnie z potrzebami.

# Tor przeszkód

Tor przeszkód składa się z czterech ścian o wymiarach 2100 x 140 oraz dwóch przenośnych przegród o wymiarach 1200x140. Taka budowa umożliwia zmianę ustawienia toru bez potrzeby używania narzędzi, pozwala skutecznie przetestować pojazd.

