

# Apacz Laser Tank

Damian Herman, SKN Teleinformatyki Apacz500,  
Wydział Elektryczny Zachodniopomorskiego Uniwersytetu Technologicznego  
Katedra przetwarzania sygnałów i inżynierii multimedialnej

22 stycznia 2019

# Rozdział 1

## Wprowadzenie do projektu

Apacz Laser Tank jest projektem zdalnie sterowanego pojazdu gąsienicowego opartego początkowo o platformę Raspberry Pi 3, a następnie ze względu na rozmiary - Arduino Mini Pro. Sterowanie odbywa się za pomocą wieloplatformowej aplikacji napisanej przy pomocy frameworka Xamarin, pozwalając na uruchomienie aplikacji na urządzeniach z Windows oraz Android.

Pojazd jest napędzany dwoma silnikami prądu stałego z przekładniami, sterowanymi przy pomocy mostka H. Komunikacja pomiędzy aplikacją, a pojazdem odbywa się przy pomocy Bluetooth. Projekt jest udostępniony w serwisie Github, na licencji MIT.

# Rozdział 2

## Instalacja

### 2.1 Wymagania

Instalacja projektu wymaga środowiska programistycznego Visual Studio 2017 w wersji przynajmniej Community. Wersja Express środowiska **nie działa**. Dodatkowo, wymagane jest zainstalowanie edytora Arduino IDE albo samego WinAVR. Zamiast Arduino IDE można użyć edytora Visual Studio Code/VSCodium z wtyczką PlatformIO. Aby aplikacja na Androida uruchomiła się, wymagany jest smartfon z wersją 6.0 Nougat lub wyższą, a w wypadku aplikacji UWP na platformę Windows 10 - wymagana jest aktualizacja Creator's Update.

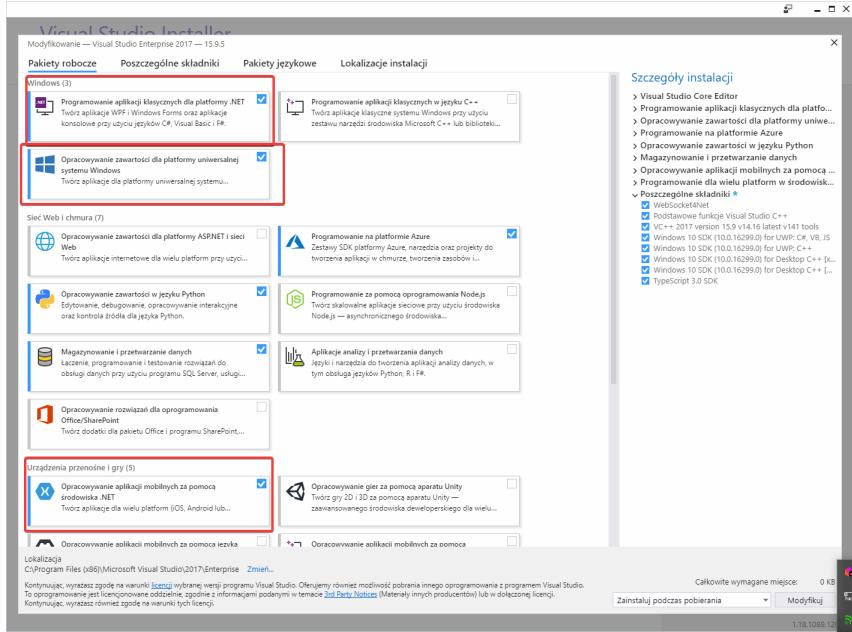
#### 2.1.1 Proces instalacji

Kod źródłowy aplikacji Xamarin oraz firmware Arduino jest dostępne są tutaj: <https://github.com/Alienwaren/Apacz500-Laser-Tank> pod postacią repozytorium git. Aby uzyskać dostęp do kodu wystarczy sklonować repozytorium przy pomocy polecenia `git clone https://github.com/Alienwaren/Apacz500-Laser-Tank`

#### Visual Studio Community 2017

Ze strony [visualstudio.com](https://visualstudio.com) należy pobrać instalator środowiska VS Community, a następnie go uruchomić. W nowym oknie zaznaczyć następujące opcje:

1. Opracowywanie zawartości dla platformy uniwersalnej systemu Windows
2. Programowanie aplikacji klasycznych dla platformy .NET
3. Opracowywanie aplikacji mobilnych za pomocą środowiska .NET



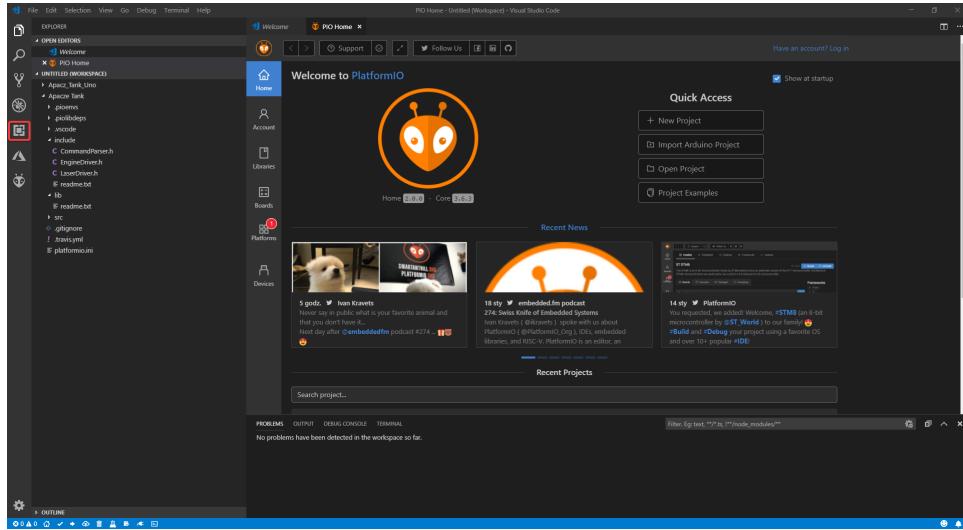
Rysunek 2.1: Okno instalatora Visual Studio, z zaznaczonymi zależnościami potrzebnymi do używania projektu

Po zatwierdzeniu instalacji, program pobierze i zainstaluje potrzebne komponenty.

## Platform IO

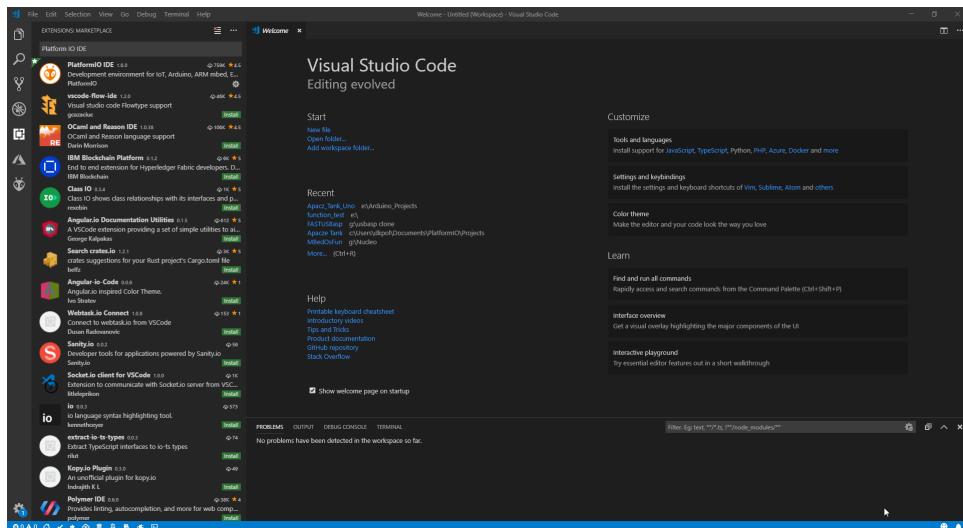
Zamiast Arduino IDE wykorzystano Platform IO, ponieważ oferuje lepsze wsparcie dla pisania kodu - podpowiedzi składni, wsparcie dla wielu platform embedded oraz integracja z git. Platform IO jest wydawany w postaci wtyczki do Visual Studio Code/VSCodium oraz edytora Atom.

Aby zainstalować wtyczkę do Visual Studio Code (zakładając, że edytor jest już zainstalowany), należy uruchomić edytor a następnie kliknąć na odpowiednią ikonę.



Rysunek 2.2: Okno główne edytora Visual Studio Code z zaznaczonym przyciskiem do instalacji rozszerzeń oraz widocznym PlatformIO

Następnie należy wpisać w wyszukiwarkę "Platform IO IDE" i kliknąć install.



Rysunek 2.3: Ekran z włączoną wyszukiwarką rozszerzeń edytora Visual Studio Code

# Rozdział 3

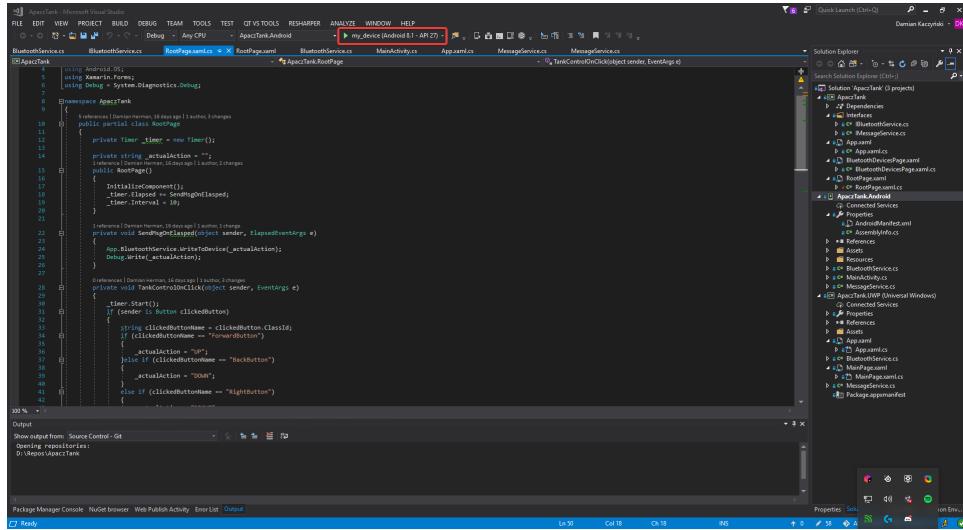
## Uruchomienie

Uruchomienie projektu będzie wymagało wgrania oprogramowania na zgodny telefon z Androidem oraz wgranie firmware do Arduino oraz ustawienia nazwy czołgu przy pomocy poleceń AT

### 3.1 Android

#### 3.1.1 Deploy aplikacji do urządzenia

Uruchom Visual Studio, a następnie otwórz plik projektu Xamarin. (Apacz-Tank.sln) Po załadowaniu projektu, jeżeli telefon jest podłączony i ma włączony tryb deweloperski z zezwoleniem na debugowanie USB, to powinien być widoczny jako urządzenie testujące. Skompiluj projekt a następnie poczekaj chwilę, aż zostanie on zainstalowany na urządzeniu docelowym.



Rysunek 3.1: Okno Visual Studio z załadowanym projektem. U góry zaznaczono przycisk komplikacji, zamiast "my device" powinno być urządzenie z Androidem

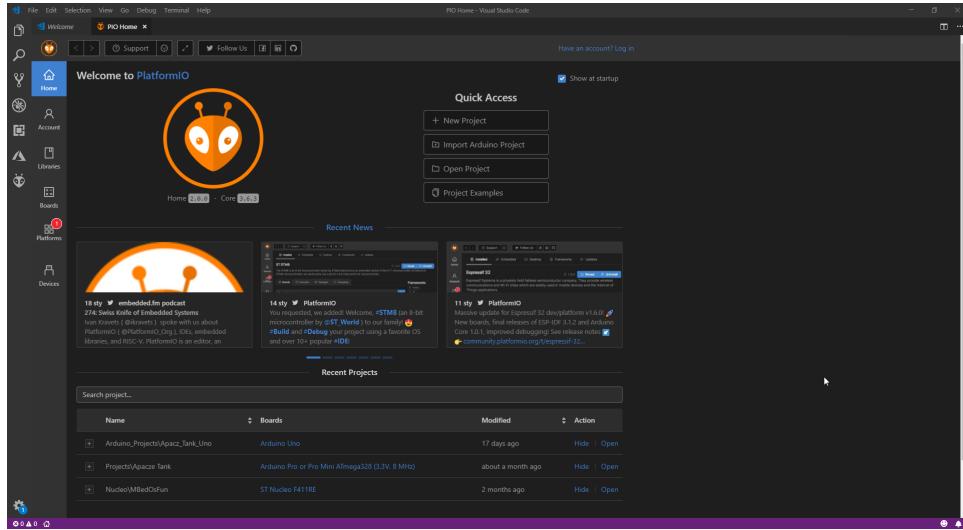
Aby połączyć się z pojazdem upewnij się, że:

1. Bluetooth jest włączony
2. Telefon jest sparowany z pojazdem
3. Poczekaj aż niebieska dioda na odbiorniku bluetooth przestanie migać

Zakładając poprawną instalację Platform IO, można przystąpić do wgrania firmware do Arduino.

## 3.2 Arduino

Otwórz edytor, a następnie kliknij *Open project*. W nowym oknie wybierz lokalizacje projektu i zaakceptuj wybór



Rysunek 3.2: Zrzut ekranowy ekranu powitalnego Platform IO

Po otwarciu projektu naciśnij *PlatformIO: Upload*. Od tego momentu naciśkaj przycisk resetu na Arduino Mini Pro, dopóki nie rozpocznie się wgrywanie firmware.



Rysunek 3.3: Pasek narzędziowy środowiska PlatformIO

Jeśli zobaczysz napis *AVRDude done. Thank you.*, to wtedy firmware został wgrany poprawnie.

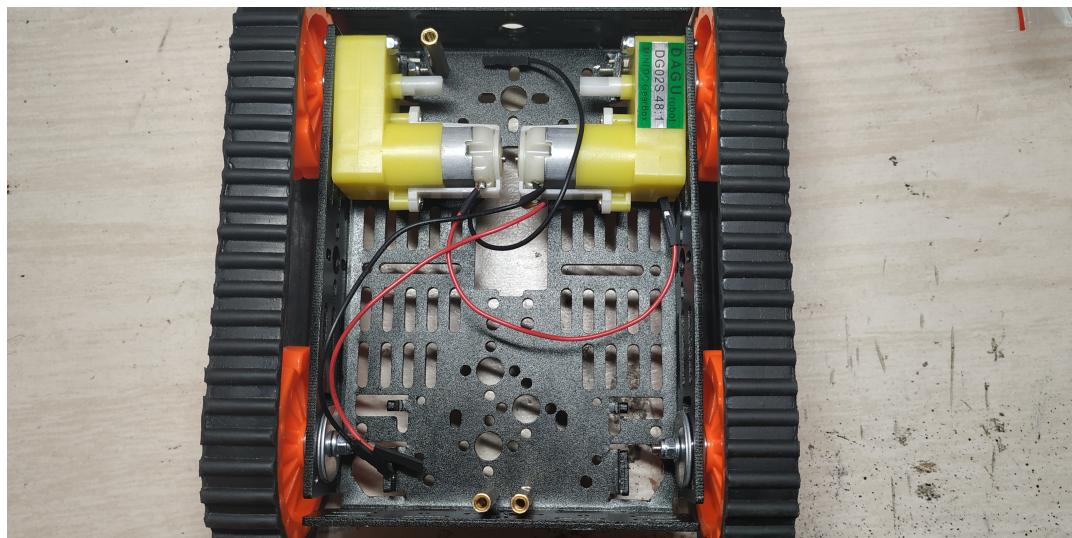
# Rozdział 4

## Opis modułów

### 4.1 Pojazd

#### 4.1.1 Mechanika

Bazą projektu jest podwozie Dagu Multichassis wyposażony w dwa silniki prądu stałego Dagu DG02S z przekładnią 48:1.



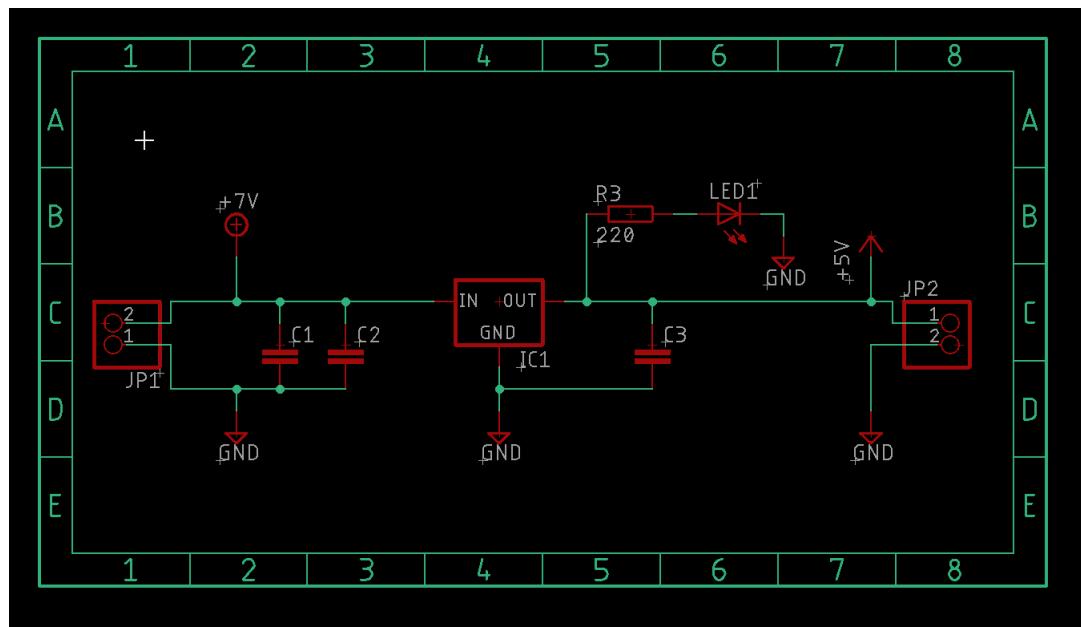
Rysunek 4.1: Podwozie pojazdu bez elektroniki

## 4.2 Elektronika

Układ sterujący jest realizowany przy pomocy platformy Arduino, a konkretnie modelu Mini Pro z użyciem zewnętrznego nadajnika-odbiornika Bluetooth HC-05. Za silniki odpowiada układ scalony DRV8835 / L298N. Schematy elektroniczne są wykonane za pomocą oprogramowania Autodesk Eagle.

### 4.2.1 Zasilanie

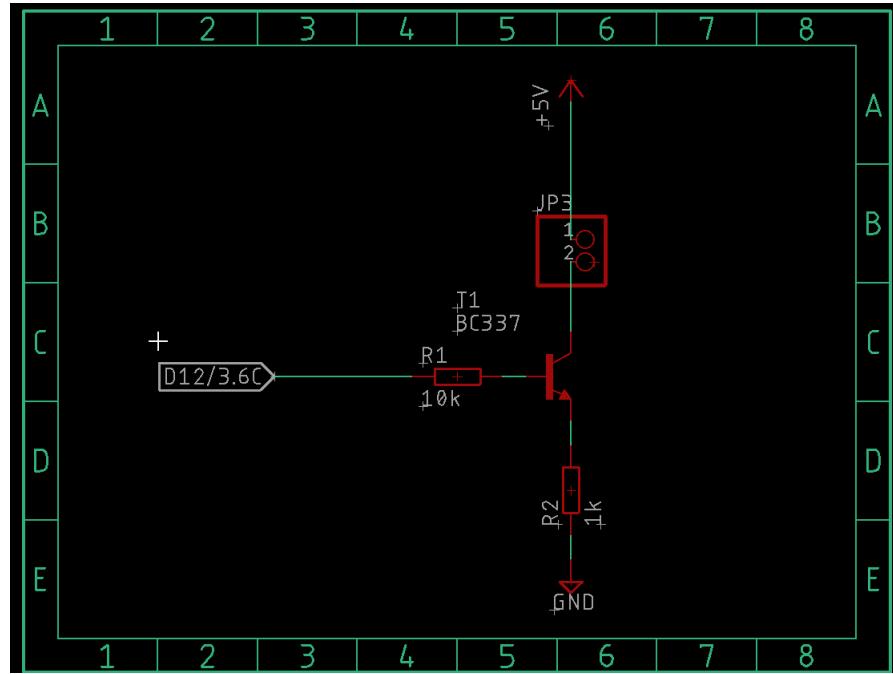
Projekt jest zasilany akumulatorem Li-Po 2S (7.2V). Pozwala to zmniejszenie masy pojazdu, przy jednoczesnym zwiększeniu prędkości. HC-05 oraz Arduino Mini Pro wymaga zasilania 5V, więc zastosowano liniowy regulator napięcia L7805.



Rysunek 4.2: Schemat elektroniczny układu zasilającego Arduino Mini Pro oraz HC-05

### 4.2.2 Laser

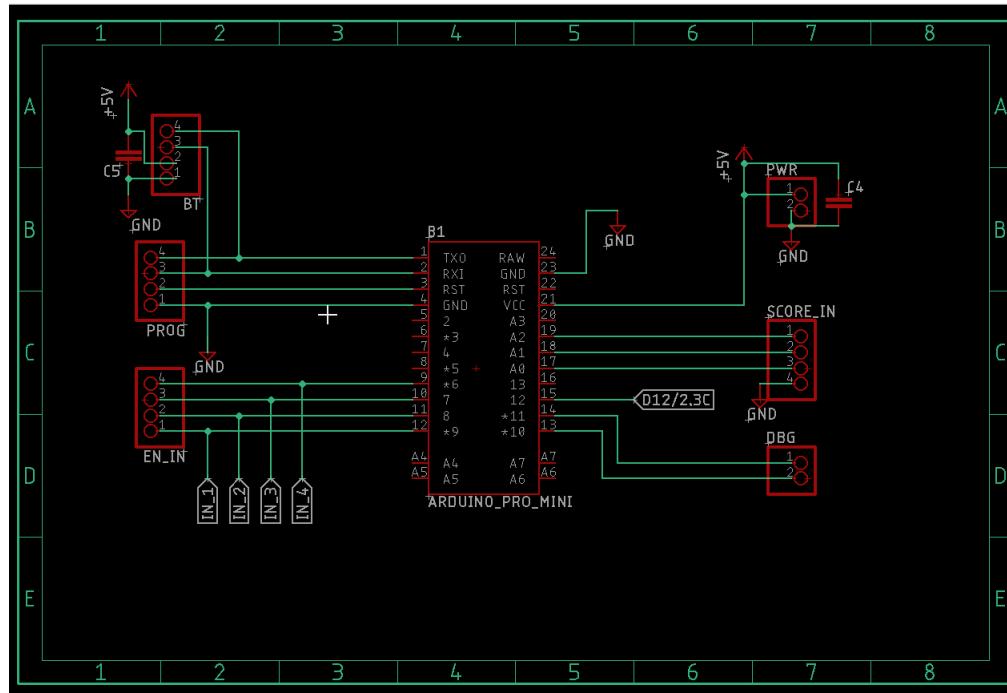
Diodą laserową steruje zwykły tranzystor bipolarny BC337. Prąd lasera to ok. 5mA



Rysunek 4.3: Schemat elektroniczny układu sterującego laserem półprzewodnikowym

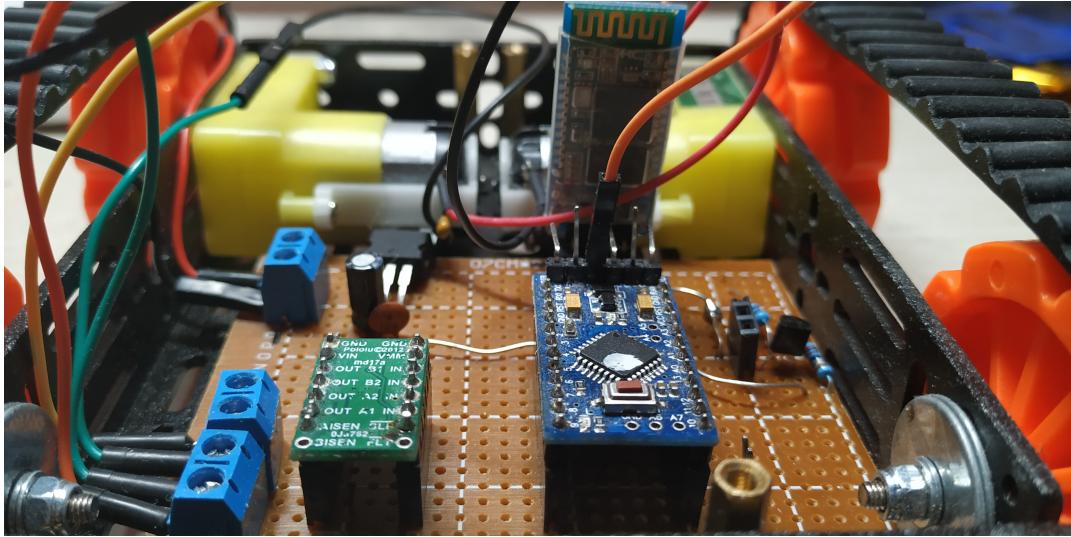
#### 4.2.3 Płyta główna oraz Bluetooth

Arduino Mini Pro komunikuje się z układem HC-05 przy sprzętowego (programowy **nie** działa; powstają błędy transmisyjne) układu UART wbudowanego w sam mikrokontroler. Zostały wyprowadzone piny RXI, TXO do programowania oraz komunikacji z HC-05, GPIO do sterowania sterownikiem silników elektrycznych prądu stałego, GPIO do sterowania laserem, GPIO programowego UART do debugowania kodu.



Rysunek 4.4: Schemat elektroniczny podłączeń do Arduino Mini Pro

Na ten moment wszystko jest przymocowane do płytka prototypowej; w planach jest wykonanie obwodu drukowanego.



Rysunek 4.5: Prototyp głównej płyty głównej projektu. Od lewej: L7805, HC-05, tranzystor BC337 sterujący laserem, DRV8835, Arduino Mini Pro

#### 4.2.4 Punktacja

Za zaliczanie punktacji odpowiadają fotorezystory połączone w dzielnik napięcia, a następnie podłączone do portu ADC. Jeżeli zostanie przekroczona wartość progowa, czołg został trafiony.

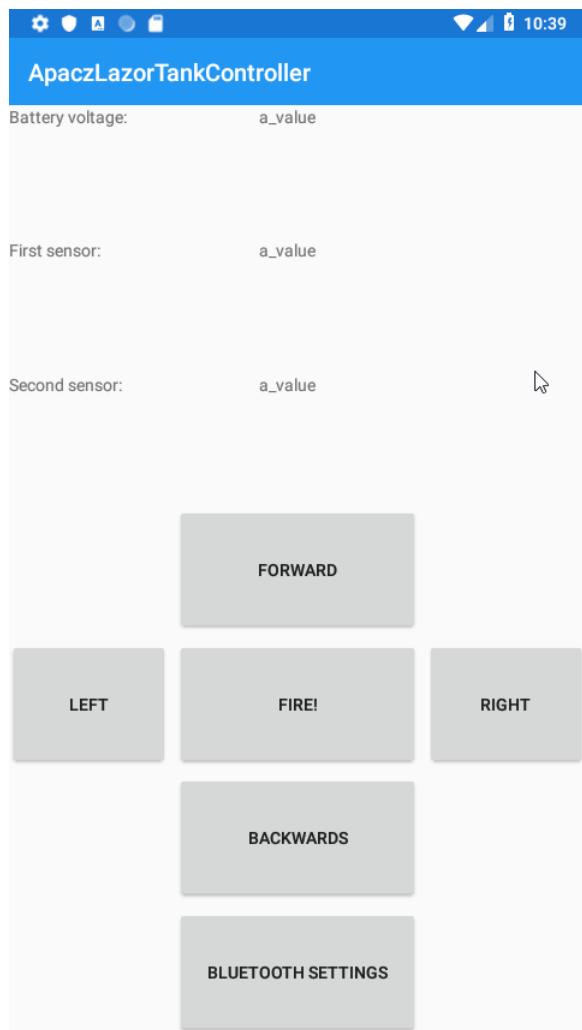
### 4.3 Software

Software projektu składa się z kilku części:

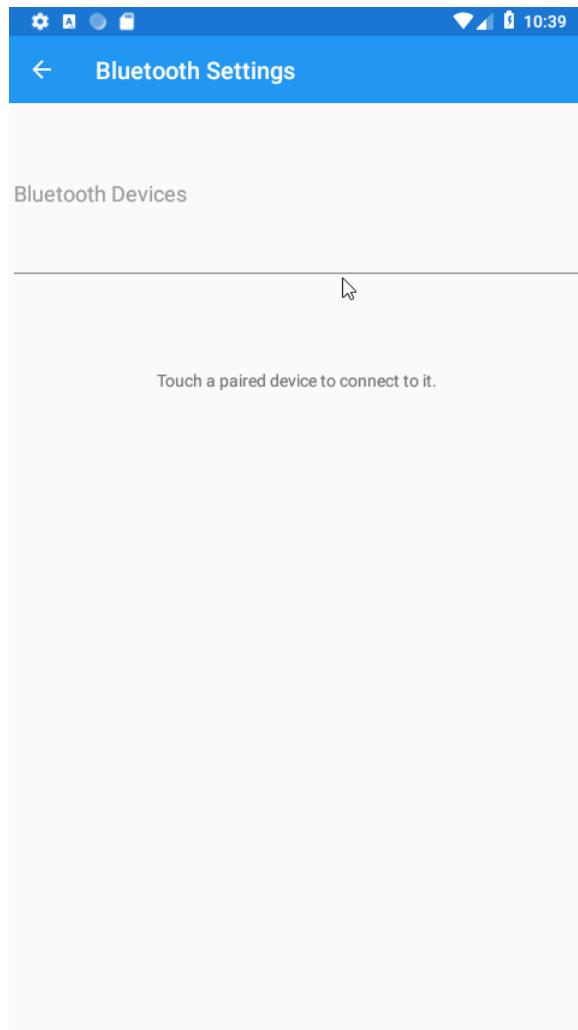
- Aplikacja Xamarin do sterowania pojazdem
- Firmware pojazdu napisany w oparciu o platformę Arduino

#### 4.3.1 Android.Xamarin

GUI aplikacji zostało zaprojektowane, tak aby było jak najproszsze. Wykonane zostało w technologii XAML zwalniając dewelopera z definicji układu kontrolek dla każdej z platform.



Rysunek 4.6: Główny ekran aplikacji, wyświetlający dane z sensorów, kontroliki do sterowania pojazdem oraz przycisk nawigujący do ustawień Bluetooth aplikacji



Rysunek 4.7: Ekran ustawień Bluetooth, wykrywający czy dane urządzenie wspiera Bluetooth. Po naciśnięciu na listę, zostanie wyświetlona lista sparowanych urządzeń Bluetooth

W momencie kliknięcia w kontrolkę sterującą, program wysyła co ok. 10 ms komendę do pojazdu docelowego.

#### 4.3.2 Arduino

Zasada działania firmware sterującego jest banalne - HC-05 wysyła cokolwiek dostanie na swój port szeregowy. Arduino oczekuje na dane na porcie a następnie odpowiednio porusza silnikami, uruchamia laser. Dodatkowo Ardu-

ino nasłuchuje na ADC, sprawdzając czy dostał w ścianę. 3 trafienia oznacza unieruchomienie czołgu.