Politechnika Wrocławska

Wydział Elektroniki, Fotoniki i Mikrosystemów

KIERUNEK: Automatyka i Robotyka (AIR)

PRACA DYPLOMOWA INŻYNIERSKA

TYTUŁ PRACY:

Budowa małego mobilnego robota laboratoryjnego klasy (1,2)

AUTOR:

Tomasz Lubelski

PROMOTOR:

Dr inż. Robert Muszyński,
Katedra Cybernetyki i Robotyki

WROCŁAW 2023

Streszczenie

Celem pracy jest budowa małego mobilnego robota laboratoryjnego klasy (1,2), pozwalającego na badanie algorytmów sterowania za pomocą dedykowanego interfejsu. Zakres prac obejmuje projekt i wykonanie konstrukcji mechanicznej, projekt i wykonanie dedykowanych układów elektronicznych, opracowanie programu dla zastosowanych mikrokontrolerów oraz przygotowanie interfejsu sterowania robotem.

Układ pracy jest następujący. W rozdziale drugim dokonano analizy kinematycznej robota. W trzecim rozdziale opisana została fizyczna realizacja konstrukcji mechanicznej robota. Czwarty rozdział zawiera szczegółowy opis układów elektroniki robota. W rozdziale piątym przedstawiono oprogramowanie wbudowane robota wraz z opisem protokołów komunikacyjnych. Rozdział szósty zawiera opis działania biblioteki udostępniającej interfejs sterowania robotem oraz przedstawiono przykłady wykorzystania tejże biblioteki. Rozdział siódmy podsumowuje pracę.

Cel pracy został osiągnięty. Wykonany został robot klasy (1,2), spełniający założenia projektu. W przyszłości robot powinien posłużyć do przeprowadzania badań w obszarze algorytmów sterowania robotów mobilnych. Może także zostać wykorzystany jako stanowisko do przeprowadzania ćwiczeń laboratoryjnych dla studentów.

Słowa kluczowe: robot mobilny, platforma mobilna klasy (1,2), kinematyka robota mobilnego, systemy wbudowane, zdalne sterowanie, napęd różnicowy

Abstract

The aim of the work is to build a small mobile laboratory robot of class (1,2), allowing the study of control algorithms using a dedicated interface. The scope of work includes the design and execution of the mechanical structure, design and implementation of dedicated electronic systems, development of a program for microcontrollers used and preparation of the robot control interface.

The workflow is as follows. In the second chapter focuses on the kinematic analysis of the robot. The third chapter describes the physical implementation of the mechanical structure of the robot. The fourth chapter contains detailed description of the robot's electronics. Chapter five introduces the software built-in robot along with a description of communication protocols. Chapter six contains a description of the library providing the robot control interface and includes examples of using this library. Chapter seven summarizes the work.

The goal of the work has been achieved. A mobile platofrm (1,2) was made, meeting the assumptions of the project. In the future the robot should be used to conduct research on control algorithms of mobile robots. It can also be used as a station for conducting laboratory exercises for students.

Keywords: mobile robot, mobile platform (1,2), mobile robot kinematics, embedded systems, remote control, differential drive

Spis treści

Do składu pracy wykorzystano system przygotowania dokumentów LaTeX, opracowany przez L. Lamporta [4], będący nakładką systemu TeX, [1,2]. Matematyczne czcionki o nazwie AMS Euler, których używamy w tej pracy, zostały przygotowane przez H. Zapfa [3], przy współpracy z D. Knuthem i jego studentami, na zlecenie Amerykańskiego Towarzystwa Matematycznego. Wybrane czcionki składu tekstu, Antykwa Toruńska [6] – jeden z nielicznych krojów pisma zaprojektowany specjalnie dla języka polskiego w sposób uwzględniający jego rytm – w odczuciu autora doskonale współgrają z kształtem czcionki AMS Euler, pozwalając na uzyskanie harmonijnej całości. Składu bezszeryfowego tekstu maszynowego dokonano z użyciem opracowanej przez R. Leviena czcionki o nazwie Inconsolata

[1] D. E. Knuth, The TeXbook, volume A of Computers and Typesetting. Addison-Wesley, Reading, 1986.

[2] D. E. Knuth, TeX: The Program, volume B of Computers and Typesetting. Addison-Wesley, Reading, 1986.

[3] D. E. Knuth i H. Zapf, AMS Euler — A new typeface for mathematics. Scholary Publishing, 20:131–157, 1986.

[4] L. Lamport, LaTeX: A Document Preparation System. Addison–Wesley, Reading, 1994.

[5] R. Levien, Inconsolata. https://levien.com/type/myfonts/inconsolata.html, 2015.

[6] J. Nowacki, Antykwa Toruńska — od początku do końca polska czcionka. *Biuletyn Polskiej Grupy Użytkowników Systemu TeX*, 9:26–27, 1997.

Rozdział 1

Wprowadzenie

Tematem pracy jest budowa małego robota laboratoryjnego klasy (1,2). Roboty mobilne można podzielić na klasy ze względu na liczbę niezależnie napędzanych i niezależnie sterowanych kół. Klasy te mają różny stopień trudności w sterowaniu. Najłatwiejsze do sterowania są roboty klasy (2,0), kolejno można ustawić roboty klasy (1,1), a za najtrudniejsze uznaje się roboty klasy (1,2). Z tego względu roboty klasy (1,2) są często obiektem badań prowadzonych w zakresie algorytmów sterowania. Roboty tej klasy realizowane są zazwyczaj jako pojazdy kołowe [1] lub pojazdy z napędem typu HOG z wirującymi półsferami [2].

Robot klasy (1,2) posiada dwa niezależnie sterowane koła i jedno koło niezależnie napędzane. Można go zobrazować w postaci roweru, w którym oba koła byłyby skrętne. W tej pracy robot został zbudowany w postaci czterokołowej konstrukcji opartej o dwie platformy mobilne klasy (2,0) połączone ze sobą w sposób pozwalający na ich swobodny obrót wokół własnej osi. Ten rodzaj konstrukcji zapewnia odpowiednią stabilność w czasie jazdy, przy zachowaniu takich samych parametrów kinematycznych sterowania jak w przypadku robota dwukołowego.

Celem pracy jest skonstruowanie laboratoryjnego robota mobilnego klasy (1,2), pozwalającego na badanie algorytmów sterowania za pomocą dedykowanego interfejsu. Zakres prac obejmuje projekt i wykonanie konstrukcji mechanicznej, projekt i wykonanie dedykowanych układów elektronicznych, opracowanie programu dla zastosowanych mikrokontrolerów oraz przygotowanie interfejsu sterowania robotem.

Przeznaczeniem opisywanej konstrukcji jest testowanie algorytmów sterowania, ponieważ klasa (1,2) przez specjalistów uważana jest za trudną w sterowaniu. Dodatkowo w celu umożliwienia przeprowadzania badań nad innymi klasami, robot ten pozwala użytkownikowi na zmianę jego klasy w prosty sposób.

Układ pracy jest następujący. W rozdziale drugim dokonano analizy kinematycznej robota, ze zwróceniem uwagi na możliwość operowania w różnych klasach. W trzecim rozdziale opisana została fizyczna realizacja konstrukcji mechanicznej robota. Czwarty rozdział zawiera szczegółowy opis układów elektroniki robota. W rozdziale piątym przedstawiono oprogramowanie wbudowane robota wraz z opisem protokołów komunikacyjnych. Rozdział szósty zawiera opis działania biblioteki udostępniającej interfejs sterowania robotem oraz przedstawia przykłady wykorzystania tejże biblioteki. Rozdział siódmy podsumowuje pracę.

Rozdział 2

Analiza kinematyczna robota

W rozdziale przedstawiono kinematykę dwukołowego robota klasy (1,2). W pewnych konfiguracjach robot tej klasy staje się robotem klasy (1,1) lub (2,0), które także zostały opisane w kolejnych podrozdziałach. Przedstawiono również model zastępczy toczącego się koła w postaci dwukołowego robota z napędem różnicowym, wykorzystany w procesie sterowania.

2.1 Sposób modelowania kinematyki

Aby opisać kinematykę robota mobilnego, należy określić wektor współrzędnych uogólnionych $q = (q_1, q_2, \ldots, q_n)^T \in \mathbb{R}^n$, gdzie n jest wymiarem przestrzeni konfiguracyjnej oraz prędkości $\dot{q} = (\dot{q}_1, \dot{q}_2, \ldots, \dot{q}_n) \in \mathbb{R}$. Na ruch robota nakładamy ograniczenia (więzy) wynikające zazwyczaj z założenia o braku poślizgu poprzecznego i wzdłużnego kół. Następnie formułujemy je w postaci Pfaffa

$$A(q)\dot{q} = 0 \tag{2.1}$$

z macierzą $A(q)$ o rozmiarze $l \times n$, gdzie l oznacza liczbę niezależnych ograniczeń. Kolejny etap to wyznaczenie modelu bezdryfowego układu sterowania w postaci

$$\dot{q} = G(q)u \tag{2.2}$$

o wektorze sterowań $u \in \mathbb{R}^m$, $m = n - l$, gdzie macierz $G(q)$ wyznaczana jest z zależności

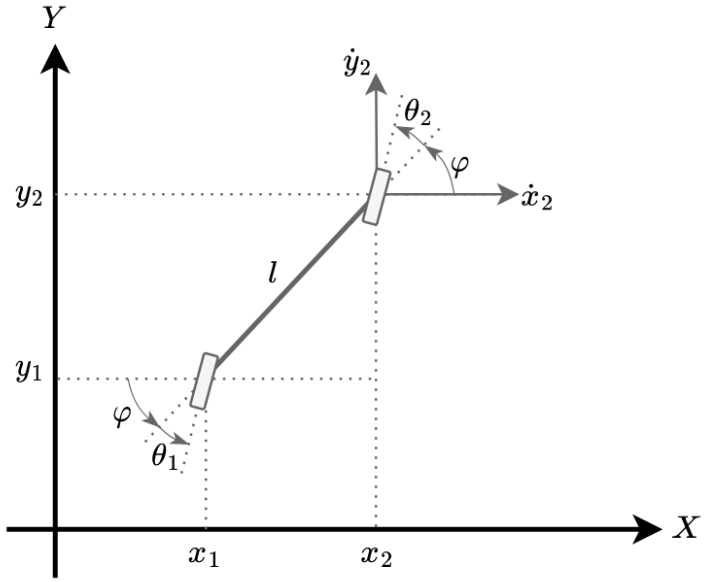
$$A(q)G(q) = 0, \tag{2.3}$$

co oznacza, że tworzące macierz wektory $g_i \in \ker A(q)$, $i = 1, \ldots, m$.
Opracowano na podstawie [3].

2.2 Robot klasy (1,2)

Robot klasy (1,2) (zobacz rysunek 2.1) posiada dwa koła niezależnie sterowane i jedno koło niezależnie napędzane. Korzystając z oznaczeń na rysunku, można wyznaczyć ograniczenie na brak poślizgu poprzecznego koła pierwszego

$$\dot{x}_1 \sin(\varphi + \theta_1) - \dot{y}_1 \cos(\varphi + \theta_1) = 0, \tag{2.4}$$



Rysunek 2.1 Robot klasy (1,2)

oraz na brak poślizgu koła drugiego

$$\dot{x}_2 \sin(\varphi + \theta_2) - \dot{y}_2 \cos(\varphi + \theta_2) = 0. \tag{2.5}$$

Przyjmując wektor konfiguracji robota w postaci $q = (x, y, \varphi, \theta_1, \theta_2)^T$ z $x = x_1$ i $y = y_1$, możemy zapisać oba ograniczenia za pomocą następującej macierzy Pfaffa

$$A(q) = \begin{bmatrix} \sin(\varphi + \theta_1) & -\cos(\varphi + \theta_1) & 0 & 0 & 0 \\ \sin(\varphi + \theta_2) & -\cos(\varphi + \theta_2) & -l\cos\theta_2 & 0 & 0 \end{bmatrix}. \tag{2.6}$$

Wybierając trzy wektory $g_1(q), g_2(q), g_3(q) \in \ker A(q)$ w postaci $g_1(q) = (l\cos(\varphi + \theta_1)\cos\theta_2, l\sin(\varphi + \theta_1)\cos\theta_2, \sin(\theta_2 - \theta_1), 0, 0)^T$, $g_2(q) = (0, 0, 0, 1, 0)^T$ i $g_3(q) = (0, 0, 0, 0, 1)^T$ otrzymujemy układ sterowania (2.2) w postaci

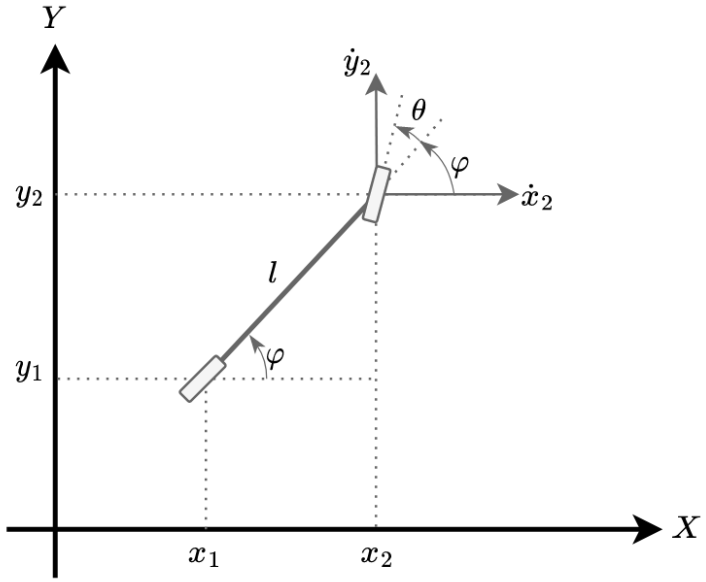
$$\begin{cases} \dot{x} = u_1 l\cos(\varphi + \theta_1)\cos\theta_2 \\ \dot{y} = u_1 l\sin(\varphi + \theta_1)\cos\theta_2 \\ \dot{\varphi} = u_1 \sin(\theta_2 - \theta_1) \\ \dot{\theta}_1 = u_2 \\ \dot{\theta}_2 = u_3 \end{cases}, \tag{2.7}$$

gdzie $u = (u_1, u_2, u_3)$ – wektor sterowań układu bezdryfowego.

2.3 Robot klasy (1,1)

Robot klasy (1,1) (zobaczy rysunek 2.2) posiada jedno koło niezależnie sterowane i jedno koło niezależnie napędzane. Jest on nazywany również samochodem kinematycznym. Korzystając z oznaczeń na rysunku, można wyznaczyć ograniczenie na brak poślizgu poprzecznego koła tylnego w postaci

$$\dot{x}_1 \sin\varphi - \dot{y}_1 \cos\varphi = 0 \tag{2.8}$$



Rysunek 2.2 Robot klasy (1,1) - samochód kinematyczny

oraz na brak poślizgu poprzecznego koła przedniego w postaci

$$\dot{x}_2 \sin(\varphi + \theta) - \dot{y}_2 \cos(\varphi + \theta) = 0. \tag{2.9}$$

Przyjmując wektor konfiguracji robota w postaci $q = (x, y, \varphi, \theta)^T$ z $x = x_1$ i $y = y_1$ oraz wyznaczając prędkości $\dot{x}_2$, $\dot{y}_2$ z prędkości $\dot{x}_1$, $\dot{y}_1$ możemy zapisać oba ograniczenia za pomocą następującej macierzy macierzy Pfaffa

$$A(q) = \begin{bmatrix} \sin\varphi & -\cos\varphi & 0 & 0 \\ \sin(\varphi+\theta) & -\cos(\varphi+\theta) & -l\cos\theta & 0 \end{bmatrix}. \tag{2.10}$$

Wybierając dwa wektory $g_1(q), g_2(q) \in \ker A(q)$ w postaci $g_1(q) = (l\cos\varphi\cos\theta, l\sin\varphi\cos\theta, \sin\theta, 0)^T$ i $g_2(q) = (0,0,0,1)^T$ otrzymujemy układ sterowania (2.2) w postaci

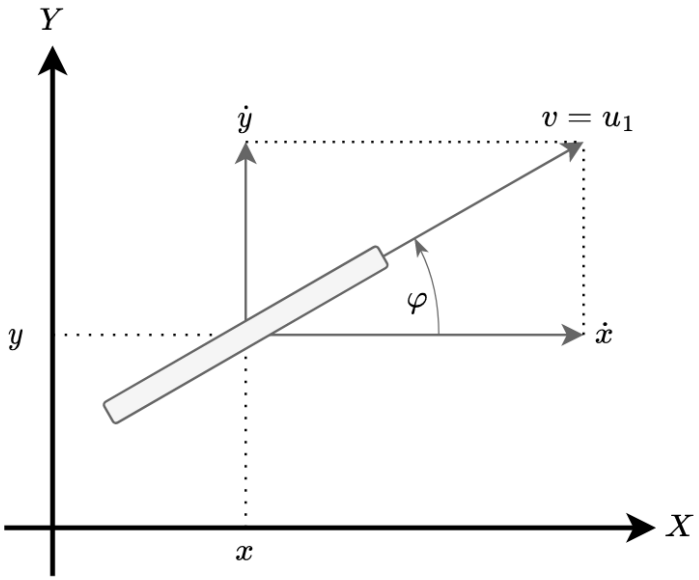
$$\begin{cases} \dot{x} = u_1 l \cos\varphi\cos\theta \\ \dot{y} = u_1 l \sin\varphi\cos\theta \\ \dot{\varphi} = u_1 \sin\theta \\ \dot{\theta} = u_2 \end{cases}, \tag{2.11}$$

gdzie $u = (u_1, u_2)$ – wektor sterowań układu bezdryfowego.

2.4 Robot klasy (2,0)

Robota klasy (2,0) można zamodelować jako toczące się koło (zobacz rysunek 2.3). Wybierając wektor konfiguracji robota w postaci $q = (x, y, \varphi)$, gdzie x i y to położenie środka koła zaś φ orientacja, ograniczenie wynikające z braku poślizgu poprzecznego koła można zapisać w postaci

$$\dot{x}\sin(\varphi) - \dot{y}\cos(\varphi) = 0. \tag{2.12}$$



Rysunek 2.3 Robot klasy (2,0) - monocykl

Stąd macierz Pfaffa to

$$A(q) = \begin{bmatrix} \sin\varphi & -\cos\varphi & 0 \end{bmatrix}. \tag{2.13}$$

Wybierając wektory $g_1(q) = (\cos\varphi, \sin\varphi, 0)^T$ i $g_2(q) = (0,0,1)^T$ otrzymujemy układ sterowania (2.2) w postaci

$$\begin{cases} \dot{x} = u_1 \cos\varphi \\ \dot{y} = u_1 \sin\varphi \\ \dot{\varphi} = u_2 \end{cases}, \tag{2.14}$$

gdzie $u = (u_1, u_2)$ – wektor sterowań układu bezdryfowego.

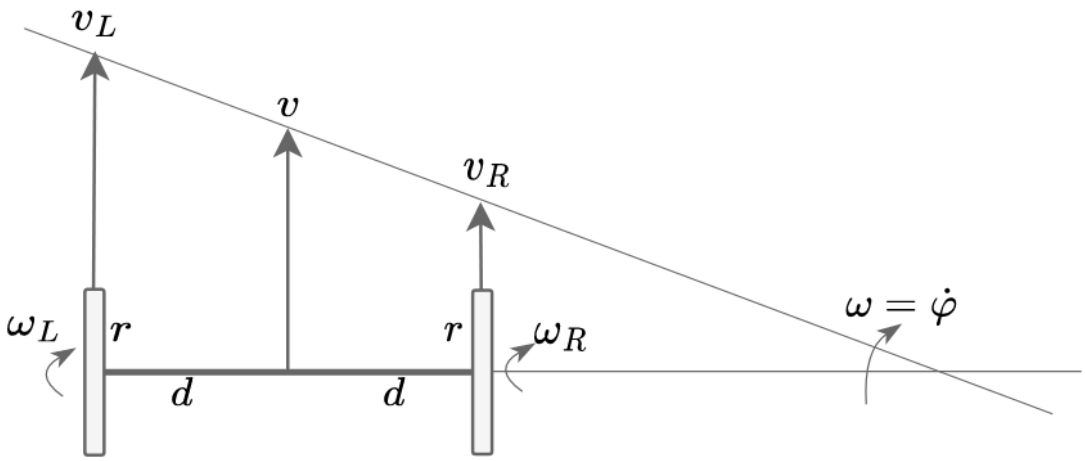
2.5 Napęd różnicowy

W celu przełożenia ruchu pojedynczego koła na ruch dwukołowego modułu robota należy wyznaczyć zależności wiążące liniowe i kątowe prędkości ruchu opisanego w podrozdziale 2.4 koła z liniowymi prędkościami ruchu kół robota o napędzie różnicowym (zobacz rysunek 2.4). Na rysunku v oznacza prędkość liniową środka robota o napędzie różnicowym, v_L i v_R to prędkości liniowe jego kół, a ω to prędkość kątowa wokół chwilowego środka obrotu. Wówczas wyznaczono prędkości liniowe kół

$$\begin{cases} v_L = v + d\omega \\ v_R = v - d\omega \end{cases}, \tag{2.15}$$

gdzie d oznacza odległość koła od środka robota. Zakładając brak poślizgu wzdłużnego kół, prędkości liniowe przełożono na prędkości obrotu kół wokół ich osi

$$\begin{cases} \omega_L = \frac{v_L}{r} \\ \omega_R = \frac{v_R}{r} \end{cases}, \tag{2.16}$$



Rysunek 2.4 Dwukołowy robot klasy (2,0)

gdzie r oznacza promień koła robota, a ω_L, ω_R to prędkości obrotowe koła lewego i prawego.

Rozdział 3

Konstrukcja mechaniczna robota

Robot mobilny klasy (1,2) może zostać zamodelowany jako połączone ze sobą dwa monocykle (klasy (2,0)). Opisywana konstrukcja została jednak zrealizowana poprzez połączenia ze sobą dwóch platform (2,0). Tym sposobem przy zastosowaniu napędu różnicowego można uzyskać ten sam sposób sterowania co w pierwszym przypadku, lecz przy braku problemów ze stabilnością w czasie jazdy. Niniejszy rozdział opisuje elementy konstrukcyjne robota oraz technologie zastosowane do ich wytworzenia. Na rysunku 3.1 przedstawiono zmontowaną konstrukcję mechaniczną robota.

3.1 Zarys ogólny

Robot został podzielony na dwa moduły posiadające własny napęd oraz zasilanie. Połączone one zostały ogniwem zwanym dalej korpusem, zamocowanym na swobodnie obracających się przegubach. W skład pojedynczego modułu robota wchodzą:

- płytka drukowana z elektroniką,
- para silników z zamontowanymi kołami,
- łożyskowana tuleja przegubu,
- kulki podporowe typu caster,
- pokrywa spodnia.

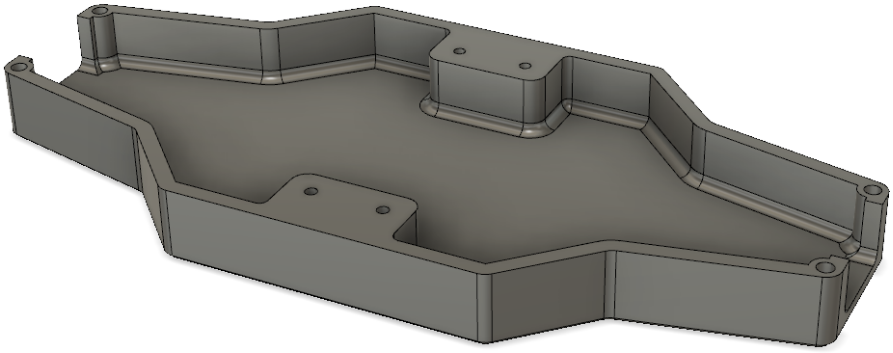
Elementy takie jak korpus, tuleje przegubów oraz pokrywa wykonane zostały w techonlogii druku 3D typu FDM, a użyty materiał to poliaktyd (PLA) w kolorze czarnym. W tym celu zaprojektowano ich trójwymairowe modele przy pomocy oprogramowania CAD/CAM – `Autodesk Fusion360` [4] dostępnego dla studentów na licencji edukacyjnej. Wykonane modele zostały przygotowane do wydruku w programie `Ultimake Cura` [5], gdzie wygenerowano pliki G-CODE zawierające niskopoziomowe instrukcje dla maszyny CNC. Dokładne rysunki techniczne wykonanych części zostały załączone w dodatku A, a modele 3D znajdują się na dołączonej płycie CD.

Rysunek 3.1 Złożona konstrukcja mechaniczna

3.2 Napęd robota

Napęd robota stanowią cztery zespolone z przekładniami szczotkowe silniki prądu stałego – Pololu 2203 [6] Posiadają one następujce cechy:

- wymiary – 24 × 10 × 12 mm,
- przełożenie przekładni – 50:1,
- przedłużony wał o długości 4,5 mm do montażu enkodera magnetycznego,
- maksymalna prędkość – 250 obr/min po przekładni,
- maksymalny moment obrotowy – 0,049 Nm,
- napięcie zasilania 3–9 V,
- prąd ciągły 40 mA,
- prąd maksymalny 360 mA.



Rysunek 3.2 Dolna pokrywa modułu

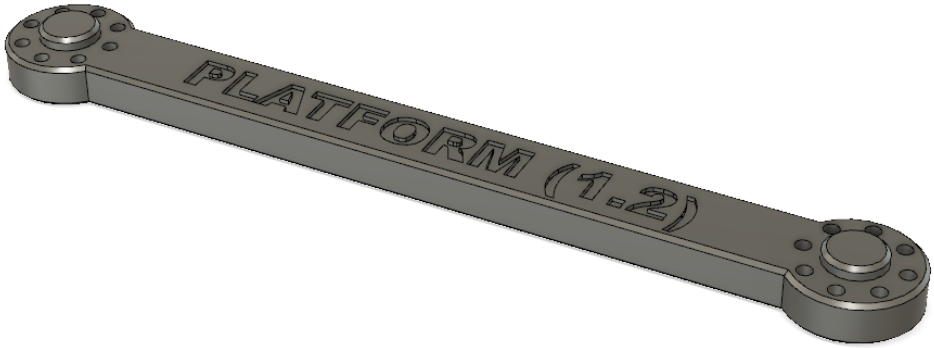
Na osi każdego silnika zamontowane zostało koło o średnicy 40 mm, co pozwala na rozwinięcie prędkości liniowej 0,52 m/s.

3.3 Budowa modułu robota

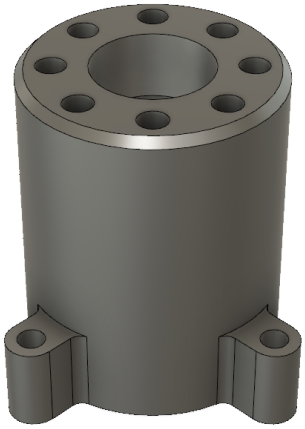
Podstawę do mocowania elementów konstrukcyjnych modułu stanowi płytka drukowana wykonana z wytrzymałego laminatu FR4 o wymiarach 160 × 75 mm. Silniki z kołami zostały przykręcone śrubami M2 za pomocą dedykowanych uchwytów do spodniej strony płytki, skrajnie po obu stronach modułu. Rozstaw kół wynosi 170 mm. Pod płytką znalazł się również centralnie zamocowany akumulator. Dolna część modułu została przykryta maskownicą (zobacz rysunek 3.2) przykręcaną w czterech punktach za pomocą śrub M2. W celu zapewnienia stabilności jazdy, w momencie kiedy osie kół obu modułów znajdą się w jednej linii, zastosowano kulki wspierające typu caster o średnicy 1/8", Przykręcono je do dolnej osłony. Przeguby obrotowe zostały przymocowane centralnie do górnej strony płytki za pomocą czterech śrub M3. Długość ogniwa łączącego moduły (zobacz rysunek 3.3) wynosi 190 mm.

3.3.1 Przeguby obrotowe

Przegub obrotowy został zbudowany z wydrukowanej tulei (zobacz rysunek 3.4), w której zamocowano na wcisk dwa nałożone na siebie łożyska kulkowe typu 608ZZ o średnicy wewnętrznej 8 mm i zewnętrznej 22 mm. W celu zapobiegnięcia przesuwania się łożysk w tulei zastosowano dystans pomiędzy zewnętrznym pierścieniem a płytką drukowaną. Przegub z korpusem został połączony aluminiowym wałkiem o długości 40 mm i średnicy 8 mm. Od strony przegubu został on zamocowany w łożyskach na wcisk, a od strony korpusu zastosowano połączenie na gorąco poprzez wtopienie wałka w plastik. Na końcu wałka znajdującym się wewnątrz tulei, za pomocą wydrukowanego adaptera został zamocowany magnes neodymowy spolaryzowany wzdłuż średnicy. Pozwoliło to na bezdotykowy odczyt położenia kątowego modułu względem korpusu.



Rysunek 3.3 Ogniwo łączące moduły – korpus



Rysunek 3.4 Tuleja przegubu

3.4 Mechanizm zmian klasy

Aby umożliwić testowanie na robocie algorytmów sterowania dla różnych klas robotów mobilnych, przewidziano możliwość szybkiej zmiany klasy robota. Uzyskano to przez blokowanie ruchu w przegubach łączących moduły z korpusem w 4 różnych pozycjach. Blokada ta polega na przełożeniu śrub M3 przez otwory w korpusie, a następnie wkręceniu ich w gwintowane wkładki wtopione w górną płaszczyznę tulei przegubu. Można wyróżnić następujące możliwe konfiguracje robota:

- żadna z osi nie jest zablokowana – klasa (1,2),
- zablokowany jest jeden moduł z osią obrotu kół prostopadle do korpusu – klasa (1,1) (*samochód kinematyczny*),
- zablokowane są oba moduły z osiami kół równoległymi do korpusu – klasa (2,0) (*monocykl*),
- zablokowane są oba moduły z osiami obrotu kół prostopadle do korpusu – platforma wielokołowo-ślizgowa typu *skid-steering*,
- zablokowany jeden lub oba moduły pod kątem $\pm 45^\circ$ względem korpusu.

Rozdział 4

Układy elektroniki robota

W związku z tym, że konstrukcja mechaniczna robota została podzielona na dwa niezależne, prawie identyczne moduły, układy elektroniczne robota rozmieszczono na dwóch identycznych płytkach drukowanych. Ponieważ jeden moduł robota jest modułem nadrzędnym (pozwala na komunikację robota na zewnątrz), elementy wymagane w nim zostały przylutowane tylko na jednej płytce, natomiast w module podrzędnym pozostały wolne miejsca na układy.

Niniejszy rozdział opisuje projekt oraz wykonanie układów elektroniki robota. Schematy ideowe oraz montażowe zostały wykonane w programie `KiCad` [7], który jest oprogramowaniem wolnym i otwartoźródłowym. Zostały one zamieszczone w dodatku B.

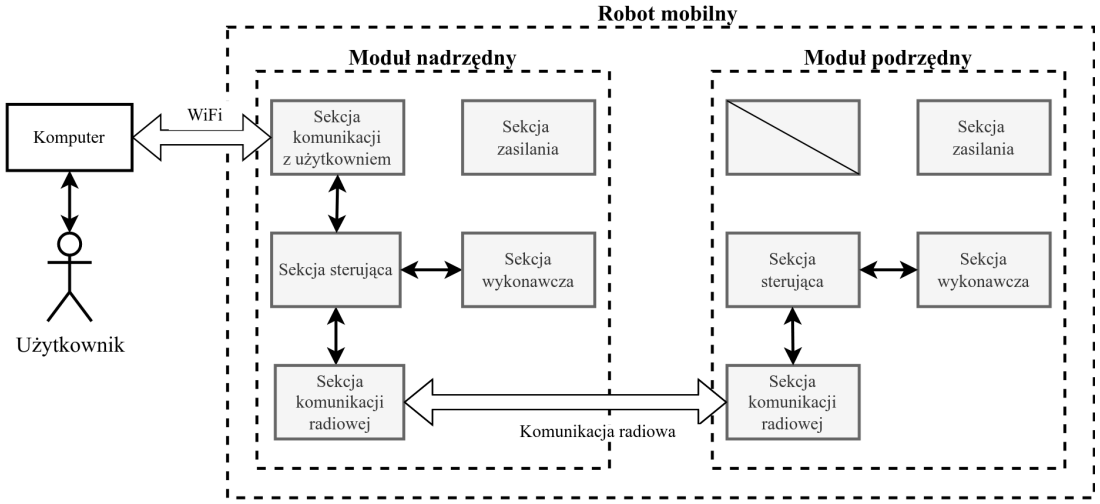
4.1 Opis funkcjonalny

Diagram podziału robota na moduły przedstawiono na rysunku 4.1. Każdy moduł poza kontrolowaniem własnych układów wykonawczych uczestniczy także w wymianie informacji zgodnie z ustaloną hierarchią. Użytkownik poprzez program komputerowy zadaje sterowanie za pomocą komunikacji bezprzewodowej WiFi do modułu nadrzędnego. Moduł nadrzędny kolejno kieruje otrzymane sterowanie do modułu podrzędnego za pomocą komunikacji radiowej. Powrót informacji o aktualnym stanie robota następuje w odwrotnej kolejności – od modułu podrzędnego, przez moduł nadrzędny, z powrotem do użytkownika.

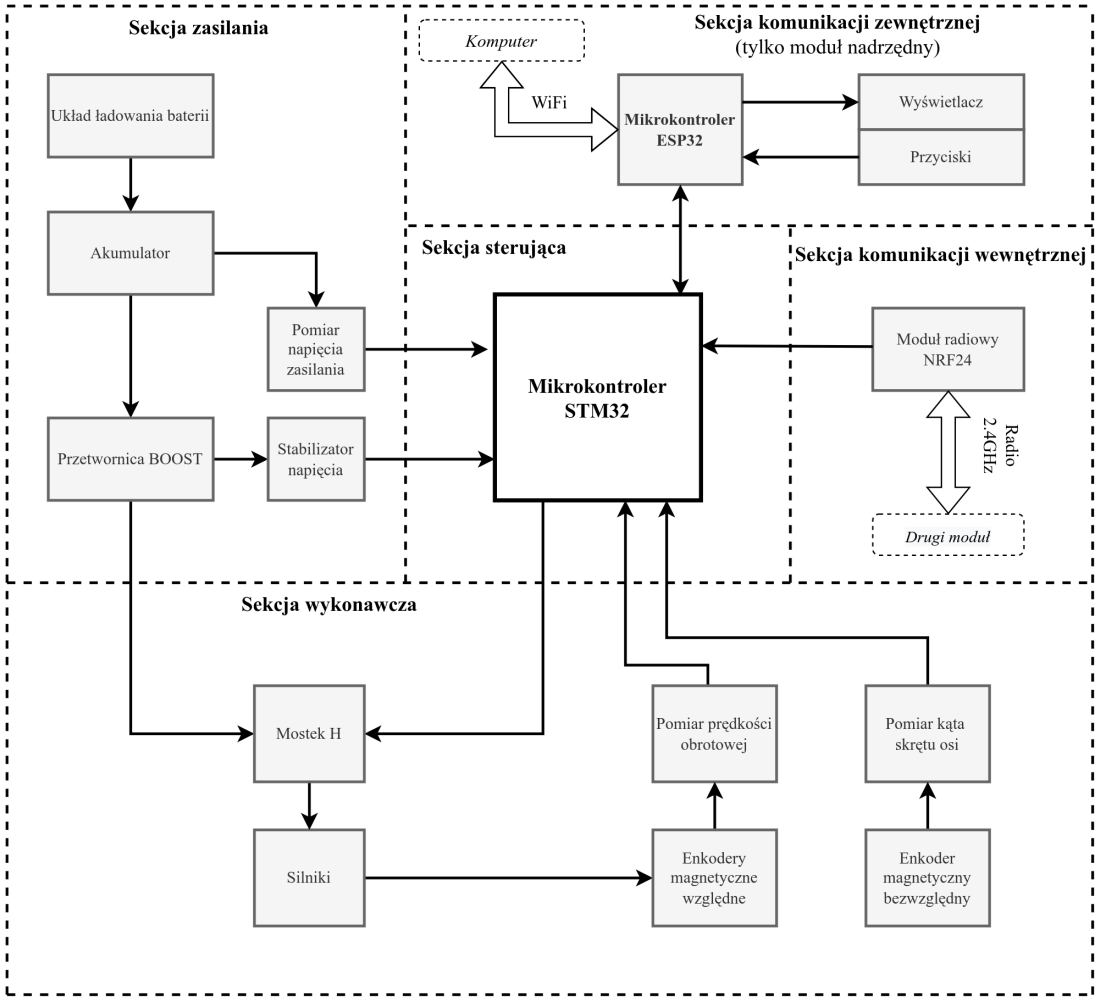
Pojedynczy moduł robota podzielony został na następujące sekcje powiązanych ze sobą układów elektronicznych:

- sterująca,
- komunikacji zewnętrznej (tylko moduł nadrzędny),
- komunikacji wewnętrznej,
- wykonawczą,
- zasilania.

Na rysunku 4.2 pokazane zostały układy wchodzące w skład poszczególnych sekcji oraz połączenia między nimi.



Rysunek 4.1 Diagram podziału robota na moduły



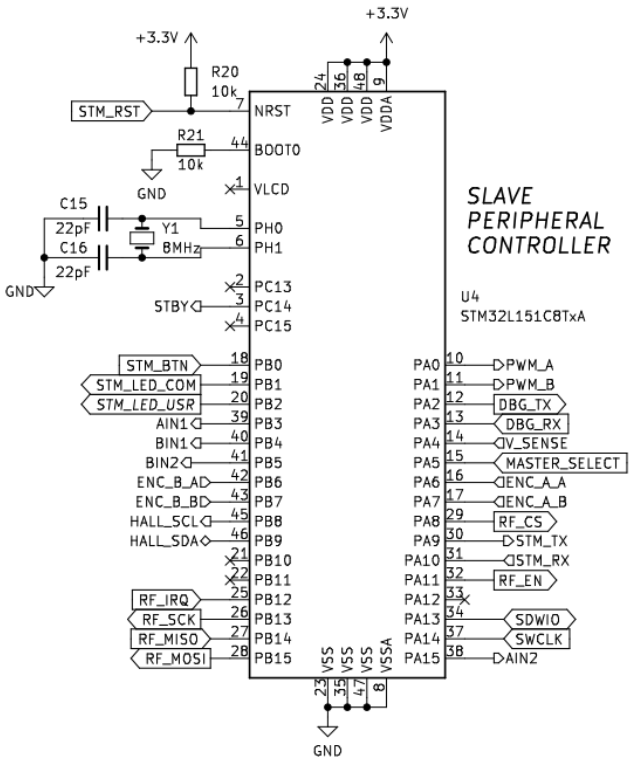
Rysunek 4.2 Diagram podziału modułu na sekcje

Sekcja sterująca realizuje zadanie sterowania sekcją wykonawczą oraz wymiany danych z sekcjami komunikacyjnymi. W skład zadania sterowania wchodzi przetwarzanie danych z czujników oraz regulacja prędkości obrotowej kół. Sekcja komunikacji zewnętrznej umożliwia zdalne zadawanie sterowania poprzez komunikację WiFi. Dodatkowo wyposażona została w wyświetlacz w celu informowania o stanie robota, w tym napotkanych błędach. Sekcja komunikacji wewnętrznej pozwala na dwukierunkową komunikację pomiędzy modułami robota. Zastosowanie łączności bezprzewodowej pozwala na nieograniczony obrót modułów robota wokół własnej osi. Sekcja wykonawcza modułu robota poprzez mostek H pozwala na sterowanie kierunkiem oraz prędkością obrotową kół. W celu przekazania do sekcji sterującej informacji zwrotnej o prędkości obrotowej kół oraz aktualnej rotacji modułu względem korpusu zastosowano enkodery magnetyczne względne oraz absolutne. Sekcja zasilania jest zasadniczym elementem każdego układu elektronicznego. W opisywanym robocie zastosowano zasilanie akumulatorowe wraz z układem ładowania, zabezpieczeniem nadprądowym oraz układem monitorowania stanu naładowania. Zaprojektowana przetwornica oraz stabilizator liniowy dostarczają odpowiednich napięć dla pozostałych sekcji układu elektronicznego robota. Dokładny opis sekcji zamieszczono w kolejnych podrozdziałach.

4.2 Sekcja sterująca

Głównym układem sekcji sterującej jest 32-bitowy mikrokontroler STM32[8] z energooszczędnej serii L1. Schemat ideowy jego podłączenia widoczny jest na rysunku 4.3. Mikrokontroler taktowany jest zegarem o częstotliwości 32 MHz, a ta częstotliwość uzyskiwana jest z powielenia częstotliwości rezonatora kwarcowego Y1 (8 MHz), za pomącą pętli PLL. Wykorzystano następujące moduły peryferyjne mikrokontrolera:

- SWD – programowanie w układzie,
- UART – debugowanie poprzez konwerter USB-UART,
- UART – komunikacja z modułem ESP32,
- SPI – komunikacja z modułem radiowym,
- I2C – odczyt wartości z enkodera absolutnego,
- ADC – pomiar napięcia akumulatora,
- timer w trybie enkodera – pomiar prędkości z enkodera kwadraturowego,
- timer w trybie PWM – generowanie sygnału prostokątnego,
- wyjścia/wejścia cyfrowe – między innymi interfejs użytkownika.



Rysunek 4.3 Schemat podłączenia mikrokontrolera STM32

4.3 Sekcja komunikacji zewnętrznej

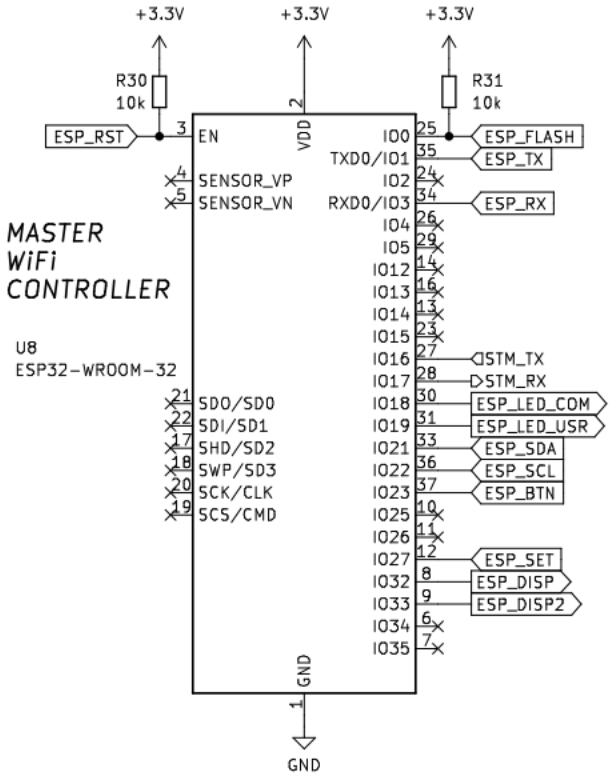
Łączność WiFi zrealizowana została za pomocą mikrokontrolera ESP32 [9]. Schemat połączeń zaprezentowany jest na rysunku 4.4. Układ ten wyposażony został w dwa rdzenie taktowane zegarem o częstotliwości 240 MHz, co pozwala na jednoczesną komunikację bezprzewodową oraz wykonywanie skomplikowanych obliczeń. Wymiana danych z mikrokontrolerem STM32 została zrealizowana za pomocą komunikacji szeregowej UART.

4.4 Sekcja komunikacji wewnętrznej

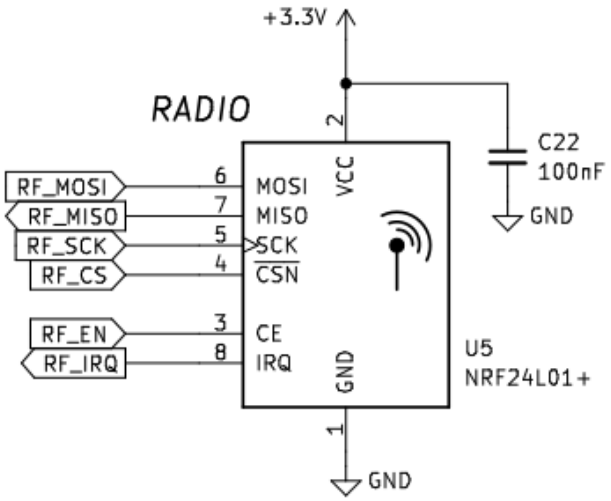
Komunikację radiową między modułami robota umożliwia moduł NRF24L01+ [10], którego podłączenie zostało przedstawione na rysunku 4.5. Wymiana danych z mikrokontrolerem odbywa się za pomocą interfejsu SPI. Układ ten pozwala na dwustronne przesyłanie wiadomości. Dodatkowo potwierdzanie odbioru czy obliczanie sum kontrolnych realizowane jest sprzętowo. Odebranie wiadomości sygnalizowane jest poprzez wystawienie stanu wysokiego na wyjściu IRQ.

4.5 Sekcja wykonawcza

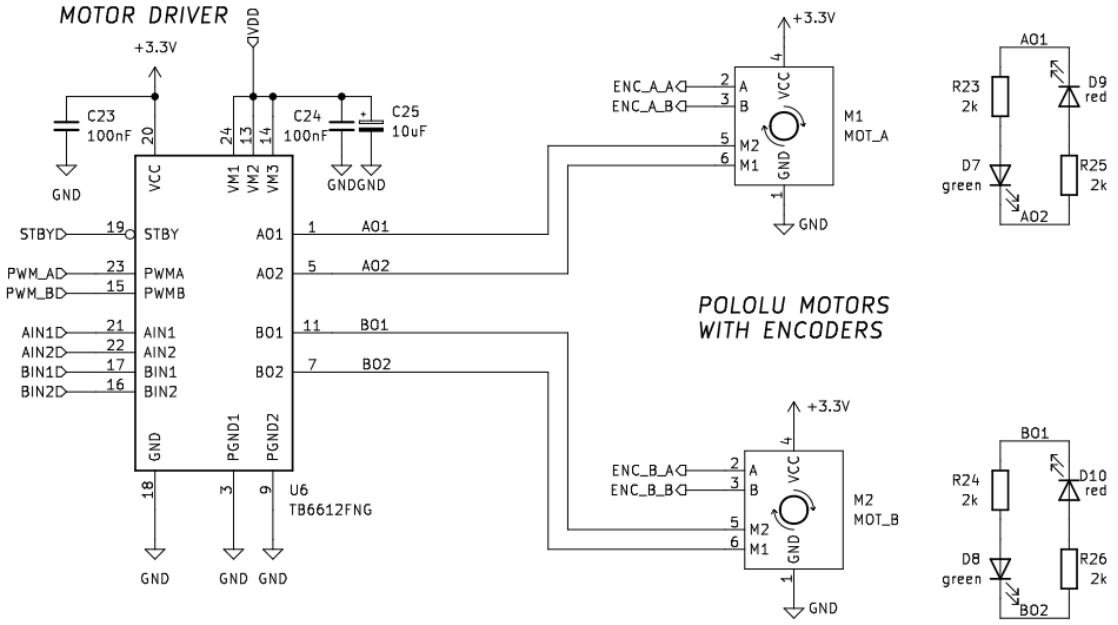
Sekcja wykonawcza realizuje zadanie sterowania silnikami, pomiaru prędkości obrotowych kół oraz pomiaru kąta skrętu przegubu.



Rysunek 4.4 Schemat podłączenia mikrokontrolera ESP32



Rysunek 4.5 Schemat podłączenia modułu radiowego



Rysunek 4.6 Schemat podłączenia mostka H oraz silników z endkoderami

4.5.1 Mostek H

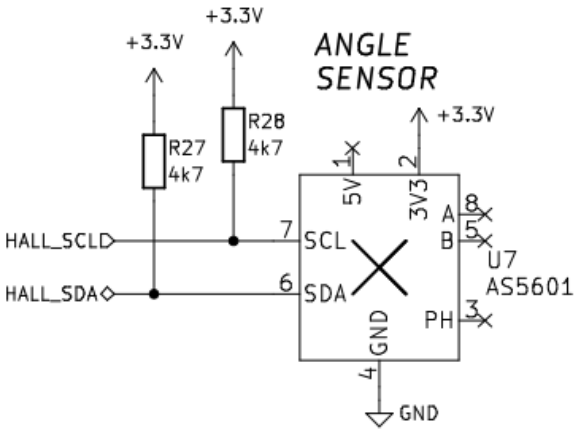
Moduł robota posiada dwa silniki szczotkowe prądu stałego Pololu 2203 [6]. Do sterowania silnikami został zastosowany podwójny scalony mostek-H TB6612FNG [11]. Mostek ten składa się z 4 kluczy tranzystorowych pozwalających na obrót wirnika silnika w obu kierunkach poprzez odwracanie polaryzacji zasilania. Regulowanie prędkości obrotowej zostało uzyskane za pomocą modulacji szerokości impulsu PWM. Poziom wypełnienia sygnału prostokątnego oraz ustawienia kluczy zadawane są przez mikrokontroler. Dodatkowo kierunek obrotu silnika sygnalizowany jest poprzez diody D7–D10. Schemat ideowy podłączenia wyżej wymienionych układów znajduje się na rysunku 4.6.

4.5.2 Pomiar prędkości obrotowej kół

Do odczytu prędkości obrotowej użyto enkoderów magnetycznych z wyjściem kwadraturowym Pololu 3081 [12]. Silniki posiadają przedłużony wał, na którym zamontowany został sześciobiegunowy dysk magnetyczny, co pozwala na uzyskanie rozdzielczości 12 impulsów na obrót wału silnika. Enkodery te wykorzystują czujniki efektu Halla TLE4946 [13]. Ich podłączenie zostało zaprezentowane na rysunku 4.6.

4.5.3 Pomiar kąta skrętu przegubu

Pomiar kąta skrętu przegubu został zrealizowany za pomocą magnetycznego enkodera absolutnego AS5601 [14]. Schemat jego podłączenia widoczny jest na rysunku 4.7. Enkoder za pomocą pomiaru pola magnetycznego tworzonego przez



Rysunek 4.7 Schemat podłączenia enkodra absolutnego

obracający się nad nim magnes neodymowy spolaryzowany wzdłuż średnicy pozwala na określenie bezwzględnego położenia kątowego przegubu z rozdzielczością 12 bitów. Komunikacja z mikroprocesorem odbywa się za pomocą magistrali I2C. Linie SDA i SCL sygnału zostały podciągnięte do zasilania poprzez rezystory R27, R28 o wartości 4,7kΩ, z powodu charakterystyki wyjść cyfrowych typu otwarty kolektor.

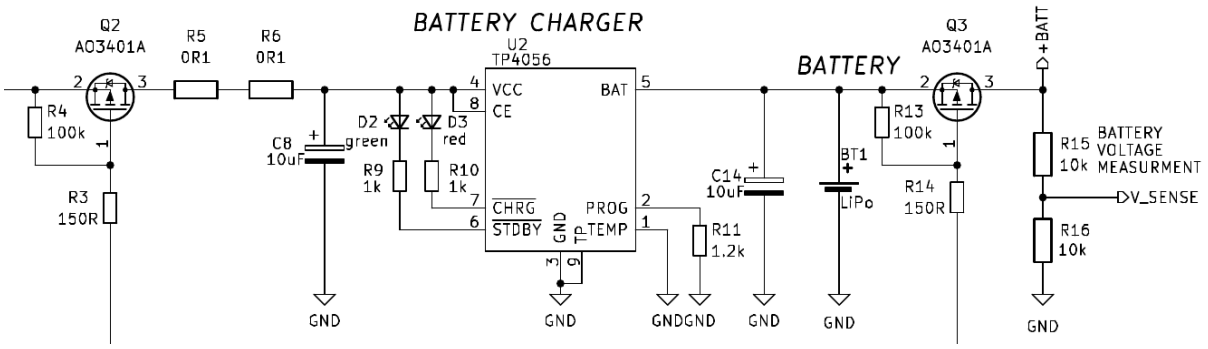
4.6 Sekcja zasilania

Sekcja zasilania robota dostarcza napięć 9V dla silników oraz 3,3V dla części logicznej, a także pozwala na ładowanie i nadzorowanie pracy akumulatora. Zastosowano akumulator litowo-polimerowy o pojemności 2400 mA oraz napięciu znamionowym 3,7 V.

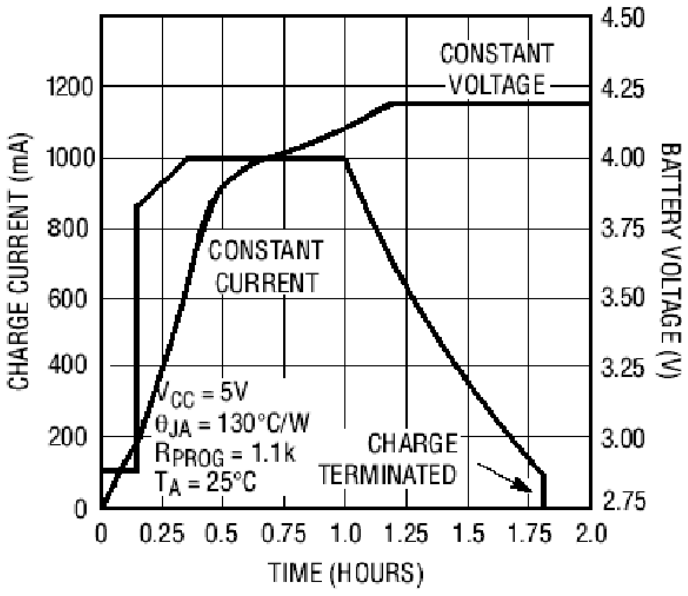
4.6.1 Układ ładowania akumulatora

Ładowanie akumulatora realizowane jest poprzez układ scalony U2 – TP4056 [15], którego aplikacja została zaprezentowana na rysunku 4.8. Jest to stabilizator stałoprądowo–stałonapięciowy (*ang. CC/CV*). Oznacza to, że w pierwszej fazie ogniwo ładowane jest prądem o wartości 1 A wyznaczonej przez rezystor R11, a po osiągnięciu napięcia granicznego – 4,2 V, następuje faza stałonapięciowa, w której prąd ładowania zaczyna spadać. Gdy spadnie on do poziomu 100 mA, następuje zakończenie ładowania. Cykl ten został zaprezentowany na rysunku 4.9. Stan ładowania wskazywany jest poprzez diody świecące D2 i D3. Diado czerwona oznacza, że ładowanie trwa, dioda zielona osiągnięcie stanu pełnego naładowania – 4,2 V.

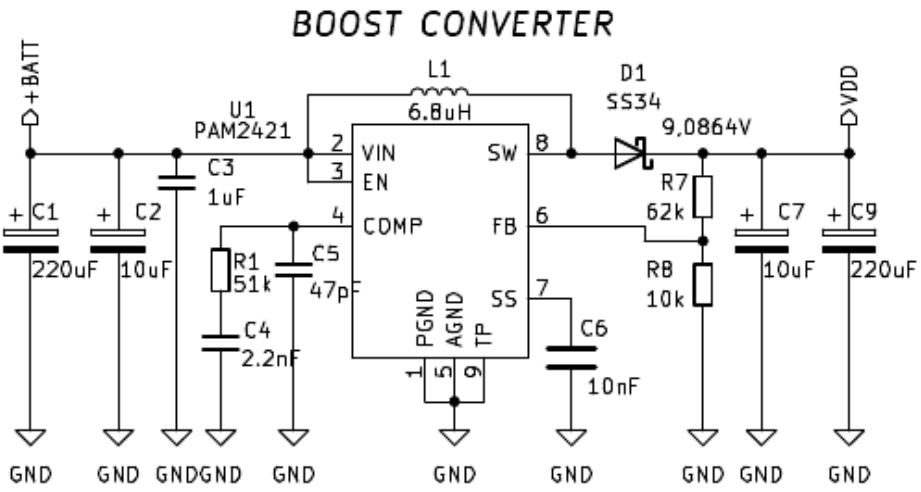
Załączanie zasilania układów elektroniki robota oraz ładowania akumulatora zostało zrealizowane za pomocą tranzystorów Q2, Q3 – P-MOSFET AO3401A [16]. Potencjał bramek tranzystorów polowych jest zrównany z potencjałem źródeł (*pull-up*) poprzez rezystory R4, R13, a sterowanie nimi odbywa się za pomocą dwupozy-



Rysunek 4.8 Schemat układu ładowania akumulatora



Rysunek 4.9 Cykl ładowania akumulatora (na podstawie [15])



Rysunek 4.10 Schemat przetwornicy

cyjnego przełącznikiem SW1. Rezystory R3 i R14 zastosowano w celu ograniczenia prądu płynącego w czasie rozładowywania bramek. W pozycji OFF zasilanie robota jest odłączone, lecz możliwe jest jego ładowanie. W pozycji ON odcięty zostaje port ładowania, a napięcie zostaje podane do pozostałych sekcji robota.

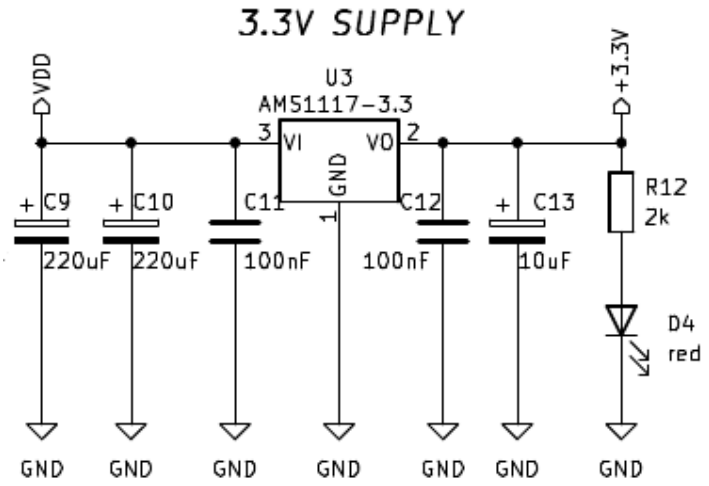
Pomiar wartości napięcia akumulatora wykonywany jest przy użyciu przetwornika analogowo–cyfrowego mikrokontrolera STM32. W celu dostosowania wartości mierzonej do maksymalnego napięcia przetwornika zasotoswano dzielnik rezystorowy R15, R16. Pozawala to na wskazanie na wyświetlaczu stanu baterii obu modułów oraz poinformowanie użytkownika o konieczności ładowania.

4.6.2 Przetwornica DC/DC

W celu dostarczenia napięcia wymaganego przez silniki (9 V), napięcie akumulatora (nominalnie 3,7 V) jest podwyższane przez przetwornicę impulsową w topologii BOOST. Przetwornica została zbudowana w oparciu o scalony sterownik impulsowy PAM2421 [17]. Jej schemat ideowy widoczny jest na rysunku 4.10. Układ zapewnia maksymalny ciągły prąd wyjściowy o wartości 3 A, co w zupełności zaspokaja zapotrzebowanie robota. Częstotliwość przełączania przetwornicy to 520 kHz. Cewka L1, dioda D1, rezystor R1 oraz kondensatory C1–C9 zostały wybrane zgodnie z notą aplikacją producenta. Wartości rezystorów R7, R8 określające sprzężenie zwrotne dla ustalenia napięcia wyjściowego zostały wyznaczone ze wzrou

$$R7 = R8 \cdot (\frac{V_{OUT}}{V_{FB}} - 1), \tag{4.1}$$

gdzie $V_{FB} = 1.1262, V_{OUT} \approx 9V$. Wspomniany układ scalony posiada zabezpieczenie nadprądowe oraz układ łagodnego startu (*ang. soft-start*), którego stała czasowa wyznaczana jest za pomocą kondensatora C6, co pozwala na zabezpieczenie akumulatora przed udarem prądowym przy załączaniu zasilania.



Rysunek 4.11 Schemat stabilizatora

4.6.3 Stabilizator liniowy

Zasilanie układów logicznych jest zapewniane przez stabilizator linowy AMS-1117 [18], którego schemat apliakacyjny zaprezentowano na rysunku 4.11. Jego uzupełnienie stanowią kondensatory stabilizujące C10, C13 oraz kondensatory odsprzęgające C11, C12. Obecność napięcia 3,3V w obwodzie sygnalizuje dioda D4.

Rozdział 5

Oprogramowanie robota

Oprogramowanie robota składa się z dwóch części (zobacz rysunek 5.1). Podział ten wynika z zastosowania dwóch współpracujących ze sobą platform programowalnych. Pierwszą z nich jest program mikrokontrolera STM32 będącego centralnym elementem sekcji sterującej modułu robota, a druga to program mikrokontrolera ESP32 realizującego zadanie komunikacji z użytkownikiem poprzez wyświetlacz oraz łączność WiFi.

W rozdziale opisano strukturę oprogramowania, protokoły komunikacyjne, struktury danych, algorytmy oraz wykorzystane biblioteki. Kody źródłowe programów zostały zamieszczone na dostarczonej płycie CD.

5.1 Oprogramowanie mikrokontrolera STM32

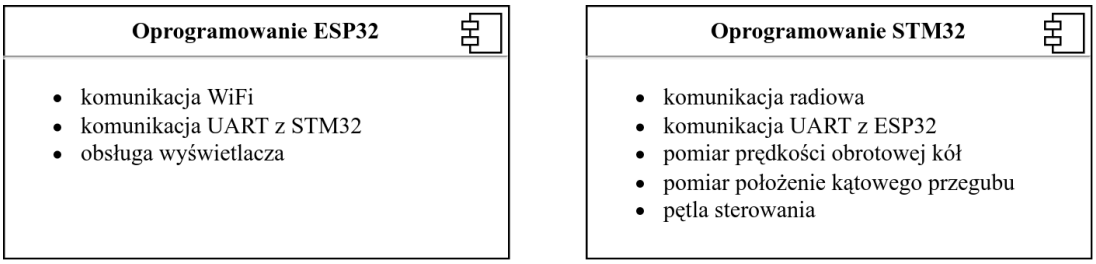
Oprogramowanie mikrokontrolera STM32 zostało napisane w języku `C++`, przy użyciu oficjalnej sprzętowej warstwy abstrakcji (HAL) oraz dedykowanego konfiguratora graficznego `STM32CubeMx` [19]. Do edycji kodu wykorzystano środowisko programistyczne `PlatformIO` [20]. Zamiast standardowej biblioteki szablonów (STL), wykorzystano jej wersję ze statyczną alokacją pamięci – Embedded Template Library (ETL) [21]. Takie podejście zapobiega fragmentacji sterty, dzięki czemu można uniknąć związanych z nią nieoczekiwanych zachowań programu, przy jednoczesnym zachowaniu wygody stosowania standardowych kontenerów.

5.1.1 Konfiguracja mikrokontrolera

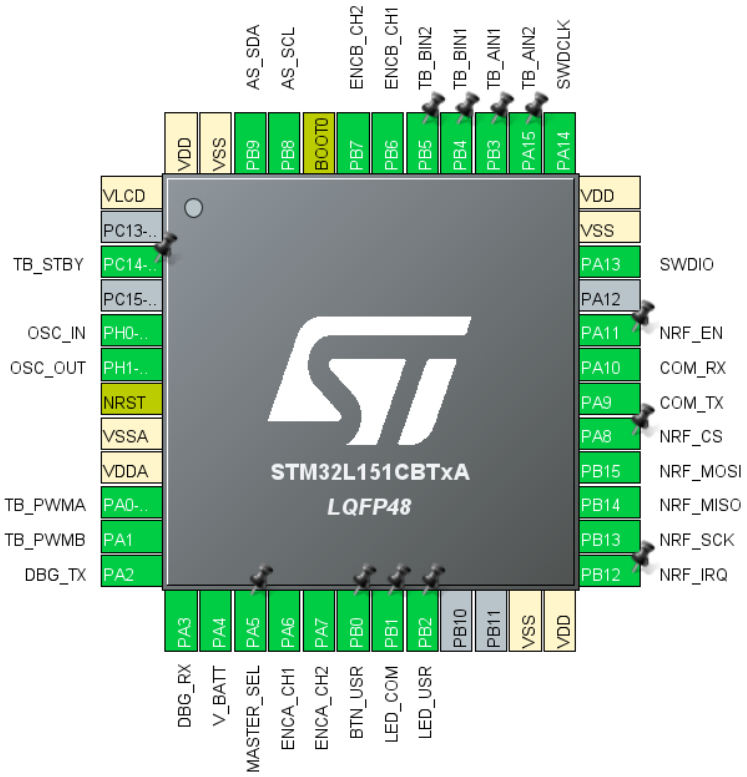
Peryferia mikrokontrolera zostały skonfigurowane w graficznym interfejsie aplikacji `STM32CubeMx`. Na rysunku 5.2 przedstawiono konfigurację wyjść i wejść układu wraz ze stosownymi opisami, dostępnymi w programie jako makra preprocesora.

Zegar procesora

Jako źródło częstotliwości wzorcowej zegarów procesora został wybrany zewnętrzny oscylator w postaci rezonatora kwarcowego o częstotliwości 8 MHz. Kolejno sygnał ten został podany na wejście pętli PLL, z której uzyskano taktowanie 32 MHz. Ostatecznie ta częstotliwość została podana na szyny zegarowe rdzenia (FCLK) i wszystkich peryferiów (APB).



Rysunek 5.1 Diagram podziału oprogramowania



Rysunek 5.2 Konfiguracja mikrokontrolera STM32 w programie STM32CubeMx

Peryferia

Skonfigurowano następujące peryferia mikrokontrolera.

- Przetwornik analogowo-cyfrowy `ADC`:
  - rozdzielczość: 12 bitów,
  - tryb oczytu: zlecenie DMA.
- Licznik `TIM2`:
  - tryb pracy: genracja sygnału PWM,
  - preskaler: 1,
  - przepełnienie: 4095.

- Liczniki TIM3 i TIM4:
  - tryb pracy: wejście enkoder,
  - przepełnienie: 65535.
- Liczniki TIM6 i TIM7:
  - tryb pracy: podstawowy.
- Interfejs komunikacyjny I2C1:
  - częstotliwość lini zegarowej: 400 MHz.
- Interfejs komunikacyjny SPI2:
  - tryb pracy: nadajnik full-duplex,
  - szybkość transmisji: 8 Mbits/s.
- Interfejsy komunikacyjne USART1 i USART2:
  - tryb pracy: asynchroniczny,
  - szybkość transmisji: 115200 bits/s.

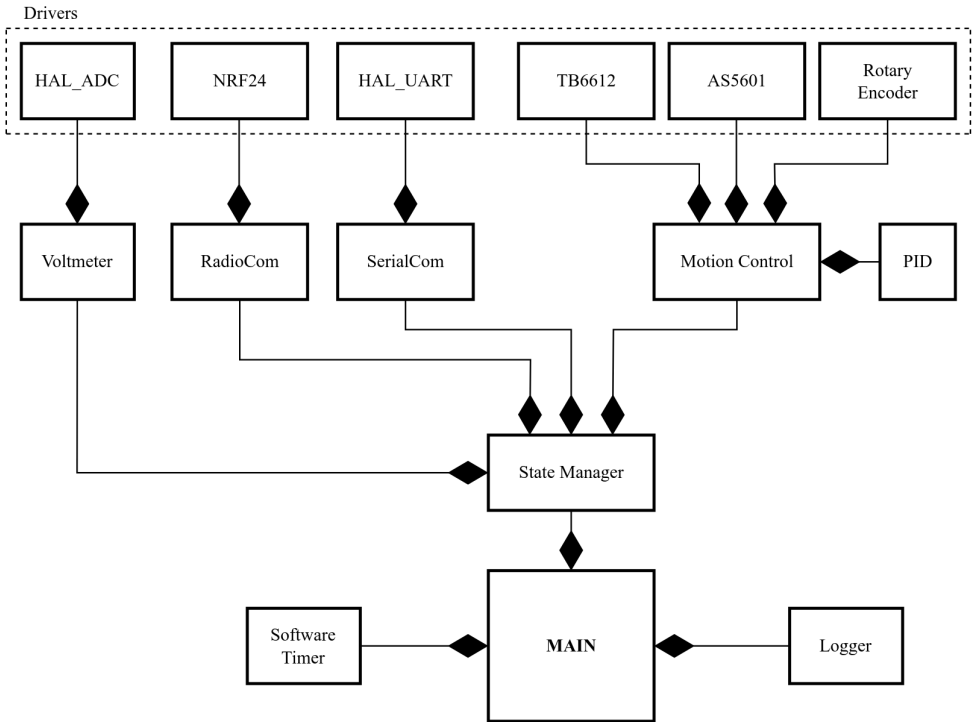
5.1.2 Struktura programu

W celu obsługi układów podłączonych do mikroprocesora napisano sterowniki pozwalające na nałożenie warstwy abstrakcji nad komunikacją sprzętową i innymi operacjami niskiego poziomu. Owe sterowniki zostały wykorzystane w klasach zarządzających nieprzerwaną realizacją poszczególnych zadań. Metody obsługujące interfejsy sterowników wywoływane są cyklicznie w pętli głównej programu, gdzie zajmują się obróbką danych przychodzących i wychodzących. Klasa menadżera stanów, w której zawierają się wszystkie powyższe interfejsy, zajmuje się przekazywaniem danych między nimi. Opisana struktura klas programu została przedstawiona na rysunku 5.3.

5.1.3 Sterowniki

Sterowanie silnikami

Sterowanie silnikami zostało zrealizowane za pomocą układu scalonego mostka-H. Do jego obsługi wykorzystano wyjścia cyfrowe mikrokontrolera oraz kanały PWM licznika TIM2. Wyjścia mocy mostka-H załączane są poprzez wystawienie stanu wysokiego na wejście STBY. Kierunek obrotu wybierany jest poprzez odpowiednią konfigurację sygnałów na wejściach AIN1, AIN2, BIN1, BIN2, a regulacja prędkości odbywa się poprzez podanie sygnału PWM na wejścia PWMA i PWMB.



Rysunek 5.3 Diagram klas oprogramowania mikrokontrolera STM32

Pomiar prędkości obrotowej kół

Pomiar prędkości obrotowej kół realizowany jest z wykorzystaniem enkoderów magnetycznych. Ich obsługa została zrealizowana za pomocą liczników sprzętowych `TIM3` i `TIM4` w trybie enkodera. Każdy impuls generowany przez enkoder powoduje inkrementację lub dekrementację rejestru `CNT` licznika, w zależności od kierunku obrotu. Określanie prędkości obrotowej polega na zliczaniu impulsów enkodera w stałym oknie czasowym.

Pomiar położenia kątowego

Bezwzględny pomiar kąta przegubu został zrealizowany za pomocą układu scalonego AS5601, z którym komunikacja odbywa się za pomocą interfejsu I2C. Posiada on rozdzielczość odczytu położenia wynoszącą 12 bitów. W celu odczytania wartości zmierzonej układ o adresie `0x36` odpytywany jest o wartość rejestru `0x0C`.

Komunikacja szeregowa UART

Komunikacja szeregowa miedzy mikrokontrolerami sekcji sterującej i sekcji komunikacji z użytkownikiem (tylko w module nadrzędnym) odbywa się z wykorzystaniem interfejsu sprzętowego `UART`. Odbiór pakietów danych został zrealizowany z użyciem zlecenia bezpośredniego dostępu do pamięci DMA z dodatkową funkcją `ReceiveToIdle`. W tym trybie dane odbierane są bez udziału głównego rdzenia, aż do przejścia układu UART w tryb `IDLE` lub odebrania maksymalnej liczby bajtów. Wysyłanie odbywa się w trybie blokującym.

Tabela 5.1 Ramka danych komunikacji szeregowej

0x7E	TOPIC	LEN	DATA...	CRC	0x81

Ramka danych składa się z jednego bajtu startu, bajtu informującego o typie wiadomości, bajtu ilości danych, bajtów danych, bajtu sumy kontrolnej oraz jednego bajtu stopu (zobacz tabela 5.1). Po odebraniu każdej ramki sprawdzana jest jej poprawność poprzez sprawdzenie obecności bajtów startu i stopu, zgodności długości odebranych danych oraz zgodności sumy kontrolnej. Poprawne ramki dodawane są do bufora kołowego w celu przetworzenia w obiegu pętli menadżera stanu.

Komunikacja z układem radiowym NRF24

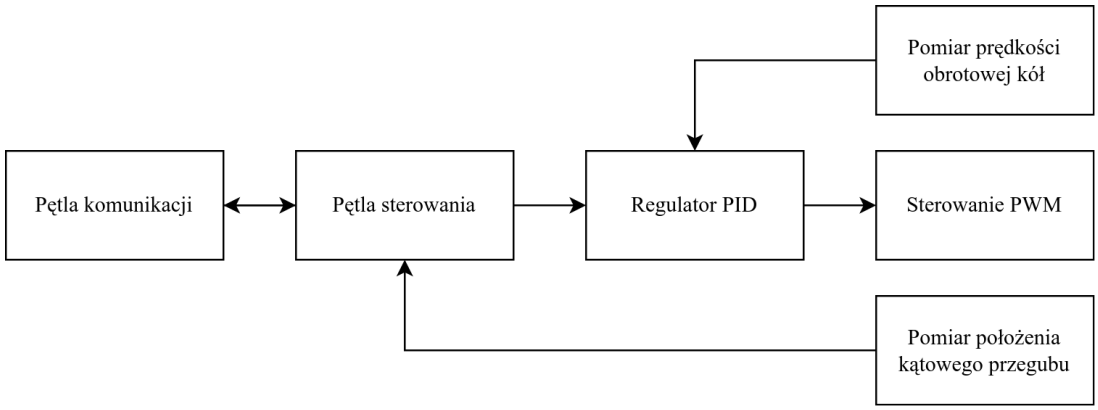
Komunikacja radiowa między modułami robota została zrealizowana z użyciem układu NRF24. Wymiana danych z układem odbywa się poprzez interfejs `SPI`. Kontrola zgodności przesyłanych pakietów, automatyczna retransmisja oraz potwierdzanie odbioru realizowane są w pełni sprzętowo. Do układu przesyłana jest ramka złożona jedynie typu z wiadomości oraz danych właściwych. Odebranie wiadomości oraz zakończenie nadawania przez układ sygnalizowane jest pojawieniem się stanu niskiego na wyjściu `IRQ`. Zdarzenie to powoduje wyzwolenie przerwania, po którym odbywa się odczytanie rejestrów układu. Odebrane dane przechowywane są w buforze kołowym w celu późniejszego obsłużenia w pętli głównej programu.

5.1.4 Pętla sterowania

Pętla sterowania robotem (zobacz rysunek 5.4) zrealizowana jest w postaci klasy `MotionControl`. Po odebraniu pakietu zawierającego zadane sterowanie wyzwalana jest zmiana stanu pętli sterującej na odpowiedni rodzaj sterowania. Kolejno w każdym obiegu pętli sterowania pomiary z enkoderów podawane są na wejścia regulatorów PID, za pomocą których wyliczane są wartości wypełnienia sygnału PWM silników. Zaimplementowano tryb sterowania bezpośredniego, w którym ustalane są prędkości każdego z silników z osobna oraz tryb sterowania różnicowego, gdzie ustalane są wartości prędkości liniowej i kątowej modułu. Do innych dostępnych poleceń można zaliczyć zatrzymanie robota oraz automatyczne ustalenie pozycji początkowej przegubów.

5.2 Oprogramowanie mikrokontrolera ESP32

Oprogramowanie mikrokontrolera ESP32 realizuje zadanie komunikacji z użytkownikiem poprzez wyświetlacz graficzny oraz łączność WiFi. Zostało ono napisane w języku `C++` przy użyciu popularnej biblioteki Arduino [22] oraz wspomnaiej wcześniej biblioteki ETL [21].



Rysunek 5.4 Diagram pętli sterowania

5.2.1 Wymiana danych

Głównym zadaniem mikrokontrolera ESP32 jest wymiana danych między interfejsem programowania robota uruchomionym na komputerze osobistym przy użyciu łączności bezprzewodowej WiFi. Z pomocą biblioteki `AsyncWebServer` [23], na mikrokontrolerze uruchomiony został serwer sieciowy udostępniający gniazdo sieciowe w protokole WebSocket. Poprzez gniazdo przesyłane są polecenia sterowania, zapytania o stan robota oraz informacje zwrotne w formie pliku JSON. Odebrany strumień danych jest przetwarzany do struktury danych typu mapa za pomocą biblioteki ArduinoJSON [24]. Kolejno w pętli głównej odbierane struktury są wysyłane do mikrokontrolera STM32 za pomocą komunikacji szeregowej UART opisanej w poprzednim podrozdziale. W przypadku odbierania informacji zwrotnej z mikrokontrolera STM32 dane o stanie robota, są wysyłane gniazdem z powrotem do interfejsu programowania robota.

5.2.2 Obsługa wyświetlacza

Obsługa wyświetlacza OLED odbywa się poprzez interfejs komunikacyjny `I2C` przy użyciu biblioteki AdafruitGFX [25]. Biblioteka ta dostarcza również zestaw podstawowych funkcji do wyświetlania grafiki np. wypisywanie tekstu, rysowanie podstawowych figur geometrycznych lub wyświetlania monochromatycznych obrazów rastrowych.

W czasie pracy robota na ekranie wyświetlane są informacje o stanie robota, w tym o pojawiających się błędach. Informacje podzielone zostały na strony, które można przewijać za pomocą umieszczonych pod wyświetlaczem przycisków. Wyświetlane są następujące dane:

- stan naładowania baterii,
- prędkości obrotowe kół,
- położenia kątowe przegubów,
- tryb sterowania,
- informacje o błędach.

5.3 Protokoły komunikacji

Komunikacja między interfejsem programowania a robotem odbywa się według następującego protokołu. Pojedynczy pakiet danych składa się zawsze z dwóch części – typu wiadomości oraz danych właściwych. Zasada ta jest zachowana bez względu na media transportu, do których zaliczają się – gniazdo sieciowe, komunikacja szeregowa i komunikacja radiowa. Dla każdego typu wiadomości zdefiniowane zostały struktury danych, w których przesyłane są informacje. W przypadku poleceń sterowania oraz informacji zwrotnej o stanie robota w skład struktury danych właściwych wchodzą dwie podstruktury – jedna na każdy moduł robota. Podstruktura pojedynczego polecenia dla jednego modułu robota posiada zdefiniowane pola danych ze ściśle określonym typem.

5.3.1 Zestawienie wiadomości

Poniżej przedstawiono zestawienie zdefiniowanych wiadomości.

1. **CTL_STOP** – komenda do zatrzymania ruchu robota
2. **CTL_POS** – komenda do nadania wstępnej rotacji modułów
   - md1
     - rotation: *float*
   - md2
     - rotation: *float*
3. **CTL_DIRECT** – komenda do nadania prędkości każdego koła
   - md1
     - speedA: *float*
     - speedB: *float*
   - md2
     - speedA: *float*
     - speedB: *float*
4. **CTL_DIFF** – komenda do nadania prędkości liniowej i kątowej modułu
   - md1
     - speedLin: *float*
     - speedAng: *float*
   - md2
     - speedLin: *float*
     - speedAng: *float*
5. **STATE_REQUEST** – zapytanie o stan robota

6. **STATE_RESPONSE** – informacja zwrotna o stanie robota
    - md1
        - volatage: *float*
        - rotation: *float*
        - speedA: *float*
        - speedB: *float*
    - md2
        - volatage: *float*
        - rotation: *float*
        - speedA: *float*
        - speedB: *float*
7. **ACK** – potwierdzenie odebrania wiadomości
8. **CORRUPTED** – prośba o retransmisję

Rozdział 6

Interfejs sterowania robota

Interfejs sterowania robota został zrealizowany w formie biblioteki języka Python [26]. Język ten jest szeroko stosowany w różnego rodzaju badaniach ze względu na mnogość dostępnych bibliotek pozwalających na wykonywanie obliczeń numerycznych. Do takich bibliotek można zaliczyć bibliotekę `numpy` [27] oferującą między innymi interfejs obliczeń macierzowych, bibliotekę `scipy` [28] oferującą numeryczne całkowanie równań różniczkowych, czy bibliotekę `sympy` pozwalającą na symboliczne rozwiązywanie równań. Dzięki temu zarówno algorytmy planowania ruchu, jak i sterowania robotem mogą zostać zaimplementowane w postaci jednego skrytpu w językau Python.

W rozdziale opisano interfejs użytkownika przygotowanej biblioteki oraz przykłady jej użycia. Biblioteka wraz z przykładami została zamieszczona na dołączonej płycie CD.

6.1 Opis biblioteki

Przygotowana biblioteka `robot_lib.py` zawiera klasę `Robot` pozwalającą na zadawanie parametrów jazdy robota. Po utworzeniu obiektu tej klasy inicjowane są dwa dodatkowe wątki. Pierwszy z nich obsługuje komunikację z robotem przez gniazdo sieciowe, a drugi cykliczne odpytuje o stan robota. Klasa zawiera prywatną metodę `send()`, która pozwala na wysłanie do robota wiadomości w formacie JSON [29]. Wiadomości te odpowiadają wcześniej zdefiniowanym strukturom danych poleceń. W czasie wykonywania programu odbierany stan robota przechowywany jest w dostępnym dla użytkownika polu `STATE`.

W celu rozpoczęcia jazdy robota należy wywołać metodę `RUN` jako argument podając funkcję z zaplanowanym ruchem. Dla użytkownika udostępnione zostały następujące metody zadawania ruchu:

- `Robot.CTL_DIRECT(speed1A, speed1B, speed2A, speed2B)` – zadanie prędkości obrotowej dla każdego koła robota,
- `Robot.CTL_DIFF(speed1Lin, speed1Ang, speed2Lin, speed2Ang)` – zadanie prędkości liniowej i kątowej modułów robota,
- `Robot.CTL_POS(pos1, pos2)` – zadanie automatycznego ustawienia modułów w pozycji początkowej,

```
from robot_lib import Robot
r = Robot()
def cmd():
    r.CTL_POS(0, ANGLE)
    while r.STATE.md1.rotation != 0 or r.STATE.md2.rotation != ANGLE:
      r.CTL_WAIT(1)
r.RUN(cmd)
```

Wydruk 6.1 Skrypt zadający pozycję początkową robota

```
from robot_lib import Robot
r = Robot()
def cmd():
    forward = np.arange(0, MAX_SPEED, STEP_SZIE, dtype=float)
    for i in forward:
        r.CTL_DIRECT(i, i, i, i)
        r.CTL_WAIT(STEP_TIME)
r.RUN(cmd)
```

Wydruk 6.2 Skrypt realizujący ruch przyspieszony

- `Robot.CTL_STOP()` – zatrzymanie robota,
- `Robot.CTL_WAIT(time)` – odczekanie zadanego czasu.

6.2 Przykłady użycia

Poniżej przedstawiono przykładowe skrypty sterowania robotem w różnych trybach.

6.2.1 Zadawanie pozycji początkowej

Na wydruku 6.1 zaprezentowano skrypt pozwalający na ustawienie robota w pozycji, w której tylny moduł robota jest ustawiony prostopadle do korpusu, a przedni skręcony pod wybranym kątem. Po zadaniu pozycji następuje oczekiwanie na uzyskanie zadanych wartości w zmiennych stanu robota.

6.2.2 Sterowanie w trybie bezpośrednim

Na wydruku 6.2 zaprezentowano skrypt realizujący ruch prostoliniowy jednostajnie przyspieszony. Z wykorzystaniem biblioteki numpy [27] wyliczona zostaje skwantowana tablica prędkości robota, które w pętli są zadawane kolejno na jego koła.

```
from robot_lib import Robot
r = Robot()
def cmd():
    while True:
        r.CTL_DIFF(V_LIN, V_ANG1, V_LIN, V_ANG2)
        r.CTL_WAIT(CIRCLE_TIME)
        r.CTL_DIFF(V_LIN, -V_ANG1, V_LIN, -V_ANG2)
        r.CTL_WAIT(CIRCLE_TIME)
r.RUN(cmd)
```

Wydruk 6.3 Skrypt realizujący "ósemkę"

6.2.3 Sterowanie w trybie różnicowym

Na wydruku 6.2 zaprezentowano skrypt realizujący wykonywanie przez robota "ósemki". Zadawane są prędkości liniowe i kątowe modułów robota z odpowiednim odstępem czasowym, dzięki czemu robot wykonuje 2 styczne ze sobą okręgi.

Rozdział 7

Podsumowanie

Celem pracy było skonstruowanie małego robota laboratoryjnego klasy (1,2). Zaprojektowano i wykonano konstrukcję mechaniczną robota, zaprojektowano i wykonano dedykowane układów elektroniczne, napisano program dla zastosowanych mikrokontrolerów oraz opracowano interfejs sterowania robotem.

Można stwierdzić, że cel pracy został osiągnięty. Wykonany został robot klasy (1,2) składający się z dwóch modułów będących platformami mobilnymi klasy (2,0). Zaprojektowana konstrukcja mechaniczna zapewnia odpowiednią sztywność i umożliwia stabilną jazdę robota po płaskiej powierzchni. Uruchomienie układu elektronicznego robota przebiegło pomyślnie, nie były wymagane żadne poprawki. Oprogramowanie wbudowane robota oraz zastosowany sposób komunikacji bezprzewodowej pozwala na realizację zadanego sterowania z akceptowalnymi opóźnieniami. Dzięki opracowanemu interfejsowi sterowania użytkownik ma możliwość zadawania parametrów jazdy za pomocą skryptu w języku Python. Zaimplementowany mechanizm zmian klasy pozwala na badanie algorytmów sterowania nie tylko dla tytułowej klasy (1,2), ale także dla konfiguracji samochodu kinematycznego czy dwukółki.

W przyszłości robot powinien posłużyć do przeprowadzania badań w obszarze algorytmów sterowania robotów mobilnych. Może także zostać wykorzystany jako stanowisko do przeprowadzania ćwiczeń laboratoryjnych dla studentów.

Możliwy jest dalszy rozwój projektu np. poprzez dodanie do oprogramowania robota możliwości współpracy ze środowiskiem ROS2 [30] za pomocą interfejsu microROS.

Literatura

[1] Szczepan Mejer. Budowa i sterowanie robota mobilnego klasy (1,2). https://kcir.pwr.edu.pl/~mucha/Pracki/Szczepan_Mejer_praca_inzynierska.pdf.

[2] Damian Góral. Konstrukcja robota mobilnego napędzanego dwiema półsferami. https://kcir.pwr.edu.pl/~mucha/Pracki/Damian_Goral_praca_inzynierska.pdf.

[3] K. Tchoń, R. Muszyński. *Mechanika analityczna. Notatki do wykładów z dziedziny automatyki i robotyki*. Katedra Cybernetyki i Robotyki, Wydział Elektroniki, Politechnika Wrocławska, https://kcir.pwr.edu.pl/~mucha/#skrypty, 2018.

[4] Autodesk. Strona internetowa programu Fusion360. https://fusion360.pl/.

[5] Ultimaker. Strona internetowa programu Cura. https://ultimaker.com/software/ultimaker-cura.

[6] Pololu. Nota katalogowa silnika Pololu 2203. https://www.pololu.com/product/2203/specs.

[7] Strona internetowa programu KiCad. https://www.kicad.org/.

[8] ST Microelectronics. Nota katalogowa mikroprocesora STM32L151CBT6A. https://www.st.com/en/microcontrollers-microprocessors/stm32l151cb.html.

[9] Espressif. Nota katalogowa mikroprocesora ESP32. https://www.espressif.com/sites/default/files/documentation/esp32-wroom-32_datasheet_en.pdf.

[10] Nordic Semiconductors. Nota katalogowa układu NRF24L01+. https://www.nordicsemi.com/products/nrf24-series.

[11] Toshiba. Nota katalogowa układu TB6612FNG. https://toshiba.semicon-storage.com/info/TB6612FNG_datasheet_en_20141001.pdf?did=10660&prodName=TB6612FNG.

[12] Polou. Nota katalogowa enkdoera Pololu 3081. https://www.pololu.com/product/3081/specs.

[13] Infineon. Nota katalogowa układu TLE4946. https://www.infineon.com/dgdl/Infineon-TLE4946_2K-DS-v01_00-en.pdf?fileId=db3a30431f848401011fbc925c48637f.

[14] Osram Group. Nota katalogowa układu AS5601. https://ams.com/documents/20143/36005/AS5601_DS000395_3-00.pdf/9a58e74f-f6d8-53eb-1fa2-ad62d9911ca4.

[15] Top Power ASIC Corp. Nota katalogowa układu TP4056. https://dlnmh9ip6v2uc.cloudfront.net/datasheets/Prototyping/TP4056.pdf.

[16] Alpha Omega Semiconductor Ltd. Nota katalogowa tranzystora AO3401. https://www.alldatasheet.com/datasheet-pdf/pdf/136085/AOSMD/AO3401.html.

[17] Diodes Incorporated. Nota katalogowa układu PAM2421. https://www.mouser.com/datasheet/2/115/PAM242x-247314.pdf.

[18] Advanced monolithic Systems. Nota katalogowa układu AMS1117. http://www.advanced-monolithic.com/pdf/ds1117.pdf.

[19] ST Microelectronics. Strona internetowa oprogramowania STM32CubeMx. https://www.st.com/en/development-tools/stm32cubemx.html.

[20] Strona internetowa oprogramowania PlatformIO. https://platformio.org/.

[21] Strona internetowa biblioteki ETL. https://www.etlcpp.com/.

[22] ST Microelectronics. Strona internetowa biblioteki STM32CubeMx. https://www.arduino.cc/.

[23] me-no dev. Repozytorium biblioteki AsyncWebServer. https://github.com/me-no-dev/ESPAsyncWebServer.

[24] Strona internetowa biblioteki ArduinoJSON. https://arduinojson.org/.

[25] Strona internetowa biblioteki AdafruitGFX. https://learn.adafruit.com/adafruit-gfx-graphics-library?view=all.

[26] Guido Van Rossum, Fred L. Drake. *Python 3 Reference Manual*. CreateSpace, Scotts Valley, CA, 2009.

[27] Charles R. Harris, K. Jarrod Millman i in. Array programming with NumPy. *Nature*, 585(7825):357–362, September 2020.

[28] Pauli Virtanen, SciPy 1.0 Contributors. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, 17:261–272, 2020.

[29] Specyfikacja formatu JSON. https://json-spec.readthedocs.io/reference.html.

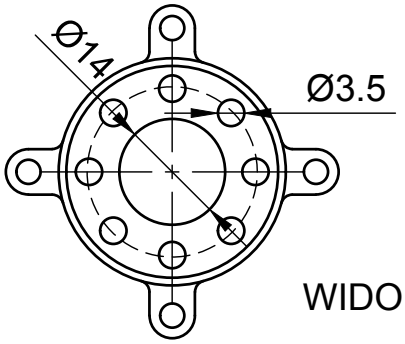
[30] Strona internetowa środowiska ROS2. https://www.ros.org/.

Spis rysunków

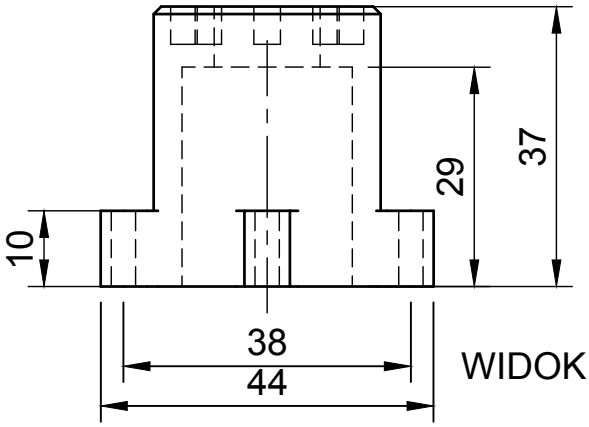
Dodatek A

Rysunki techniczne

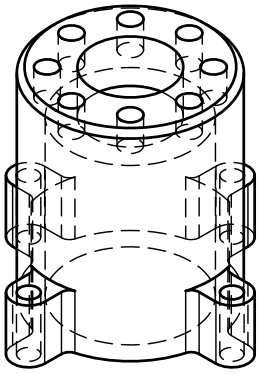
Dodatek zawiera rysunki techniczne części mechanicznych wytworzonych na potrzeby projektu wykonane w programie Fusion360.

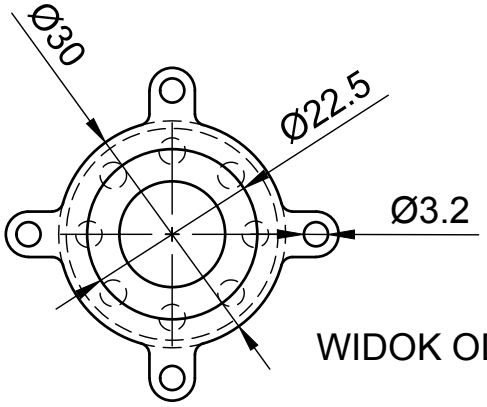


WIDOK Z GÓRY



WIDOK Z BOKU

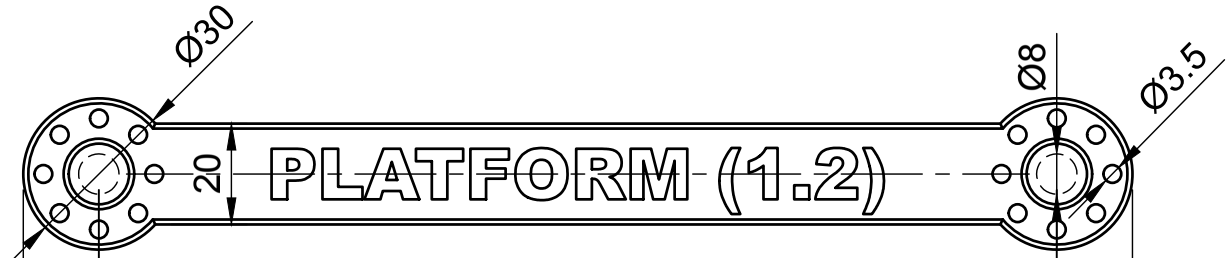




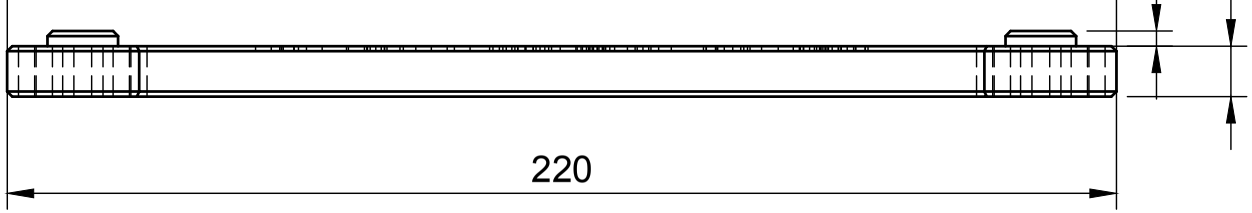
WIDOK OD SPODU

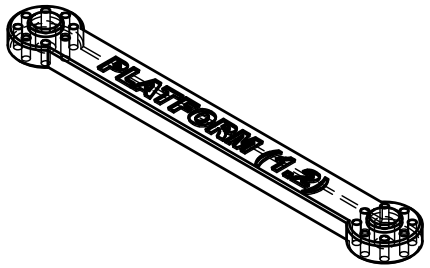
Dept.	Technical reference	Created by	Approved by	
		Tomasz Lubelski	dr inż. Robert Muszyński	
		Document type	Document status	
		Title	DWG No.	
		Tuleja przegubu		
			Rev.	Date of issue: 19.12.2022 / Sheet: 1/1

WIDOK Z GÓRY



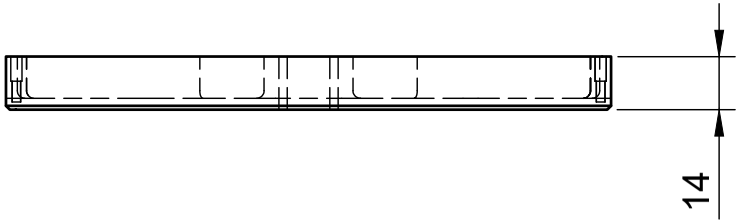
WIDOK Z BOKU



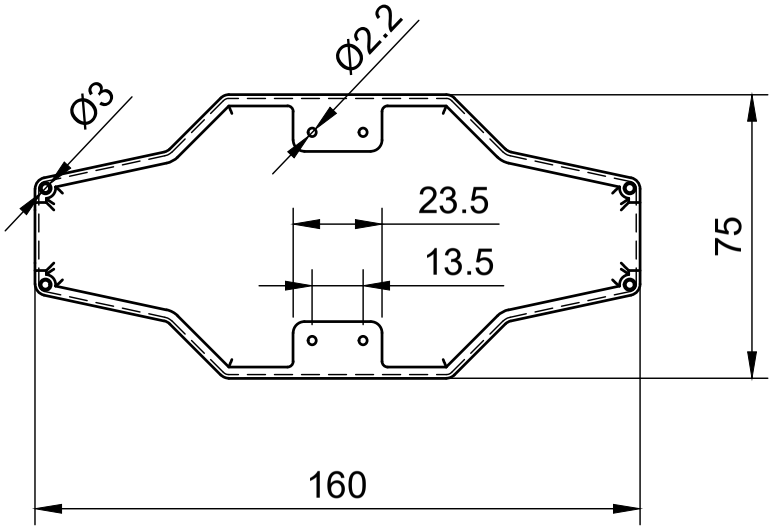


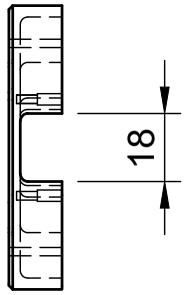
Dept.	Technical reference	Created by	Approved by
		Tomasz Lubelski	dr inż. Robert Muszyński
		Document type	Document status
		Title: Korpus robota	DWG No.
			Rev. / Date of issue: 19.12.2022 / Sheet: 1/1

WIDOK Z PRZODU

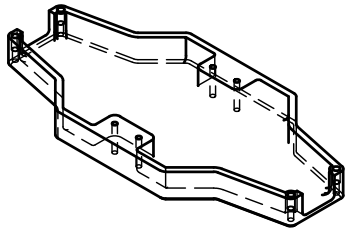


WIDOK Z GÓRY





WIDOK Z BOKU

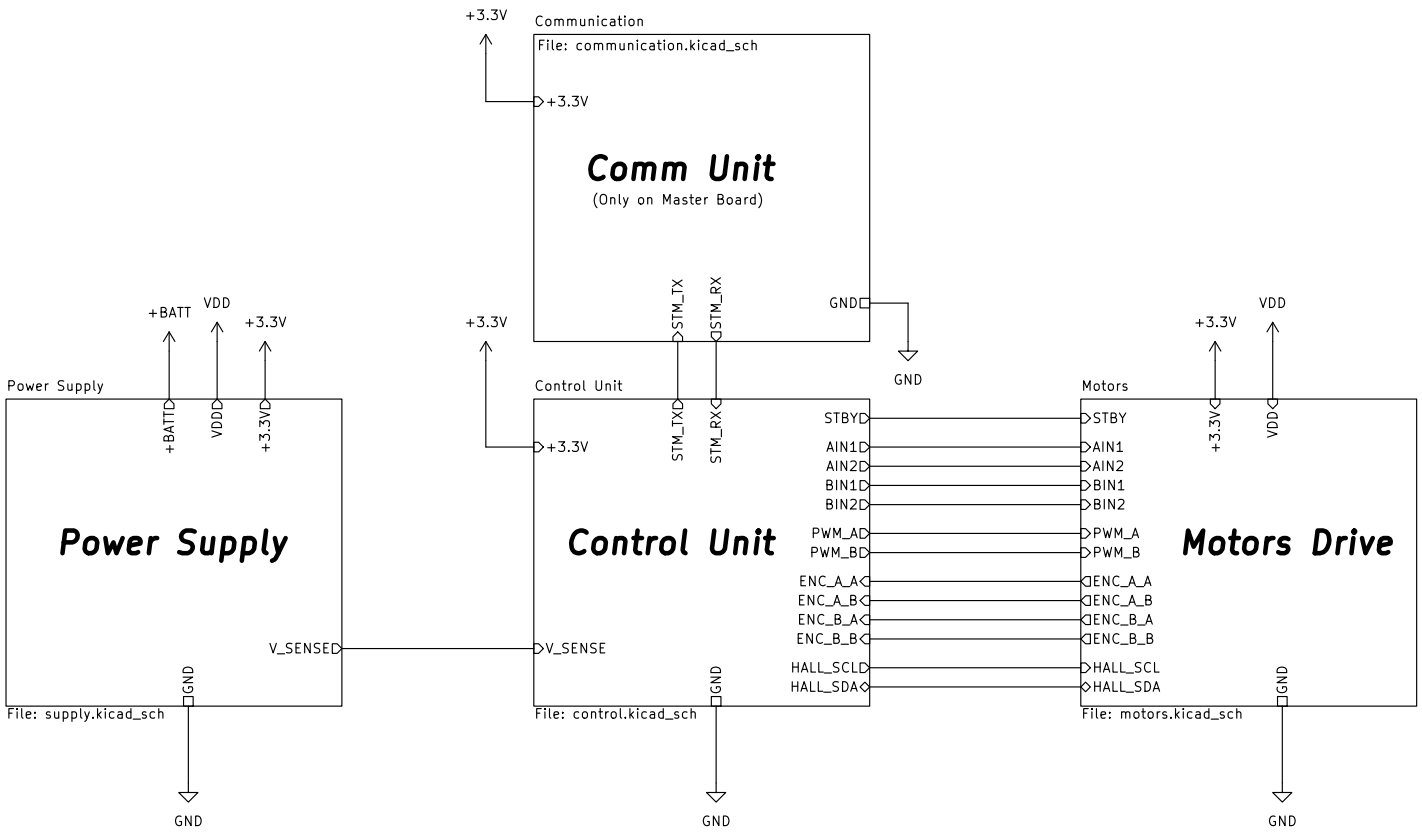


Dept.	Technical reference	Created by	Approved by
		Tomasz Lubelski	dr inż. Robert Muszyński
		Document type	Document status
		Title: Pokrywa modułu robota	DWG No.
			Rev. / Date of issue: 19.12.2022 / Sheet: 1/1

Dodatek B

Schematy elektroniczne

Dodatek zawiera schematy ideowe oraz montażowe układów elektroniki robota wykonane w programie KiCad.



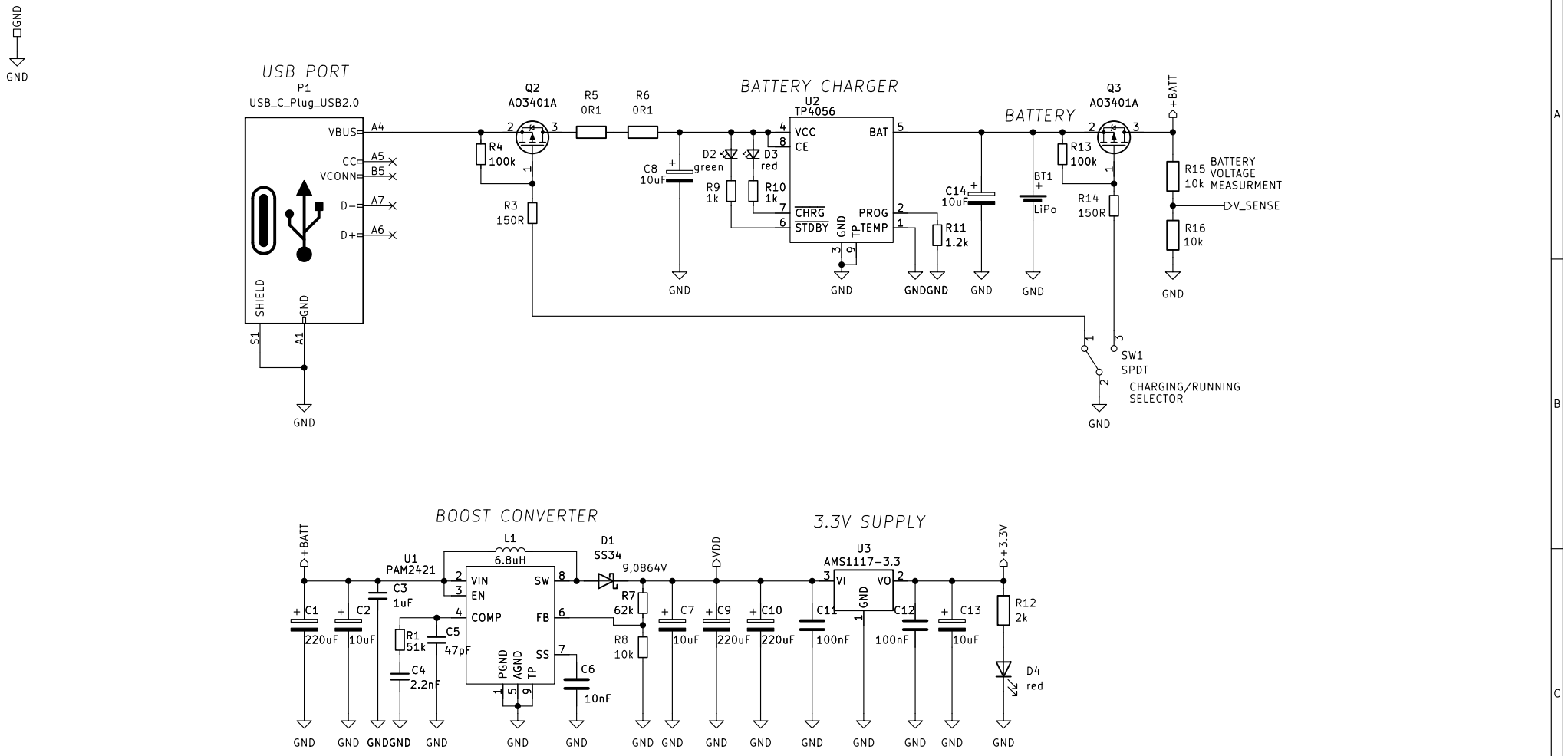
H1 MountingHole
H2 MountingHole
H3 MountingHole
H4 MountingHole
H5 Caster
H6 Caster
H7 Holder

Tomasz Lubelski 253985
ARR – Robotyka
Politechnika Wrocławska

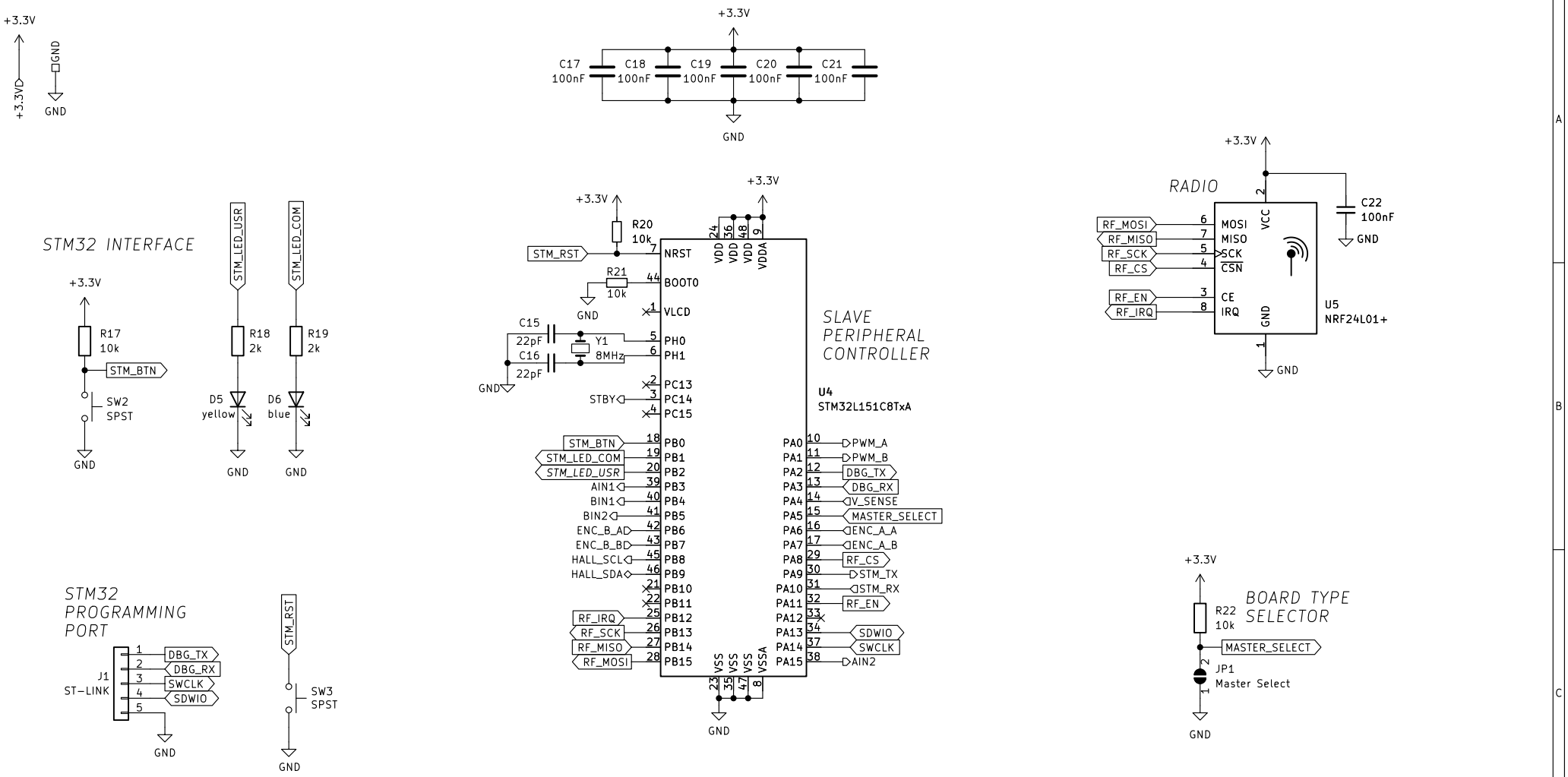
Sheet: /
File: platform1.2-electronics.kicad_sch

Title: Praca Inżynierska – Robot Klasy 1.2

Size: A4	Date: 2022-12-22	**Rev: 1.0**
KiCad E.D.A. kicad (6.0.9)		Id: 1/5



Tomasz Lubelski 253985		
ARR – Robotyka		
Politechnika Wrocławska		
Sheet: /Power Supply/		
File: supply.kicad_sch		
Title: Robot Klasy 1.2 – Sekcja zasilania		
Size: A4	Date: 2022-12-22	**Rev: 1.0**
KiCad E.D.A. kicad (6.0.9)		Id: 2/5

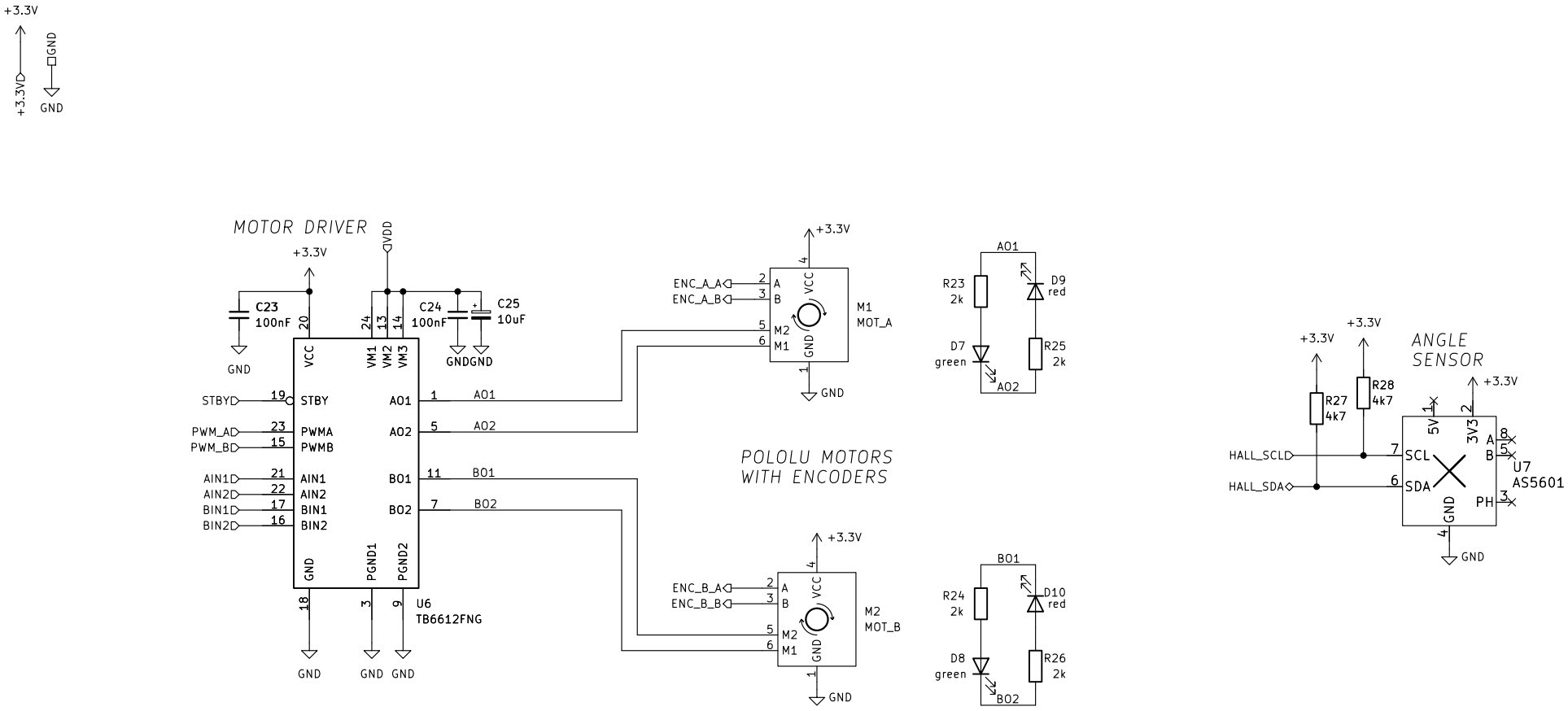


Tomasz Lubelski 253985
ARR – Robotyka
Politechnika Wrocławska

Sheet: /Control Unit/
File: control.kicad_sch

Title: Robot Klasy 1.2 – Sekcja sterująca i komunikacji wewnętrznej

Size: A4	Date: 2022-12-22	**Rev: 1.0**
KiCad E.D.A. kicad (6.0.9)		Id: 3/5

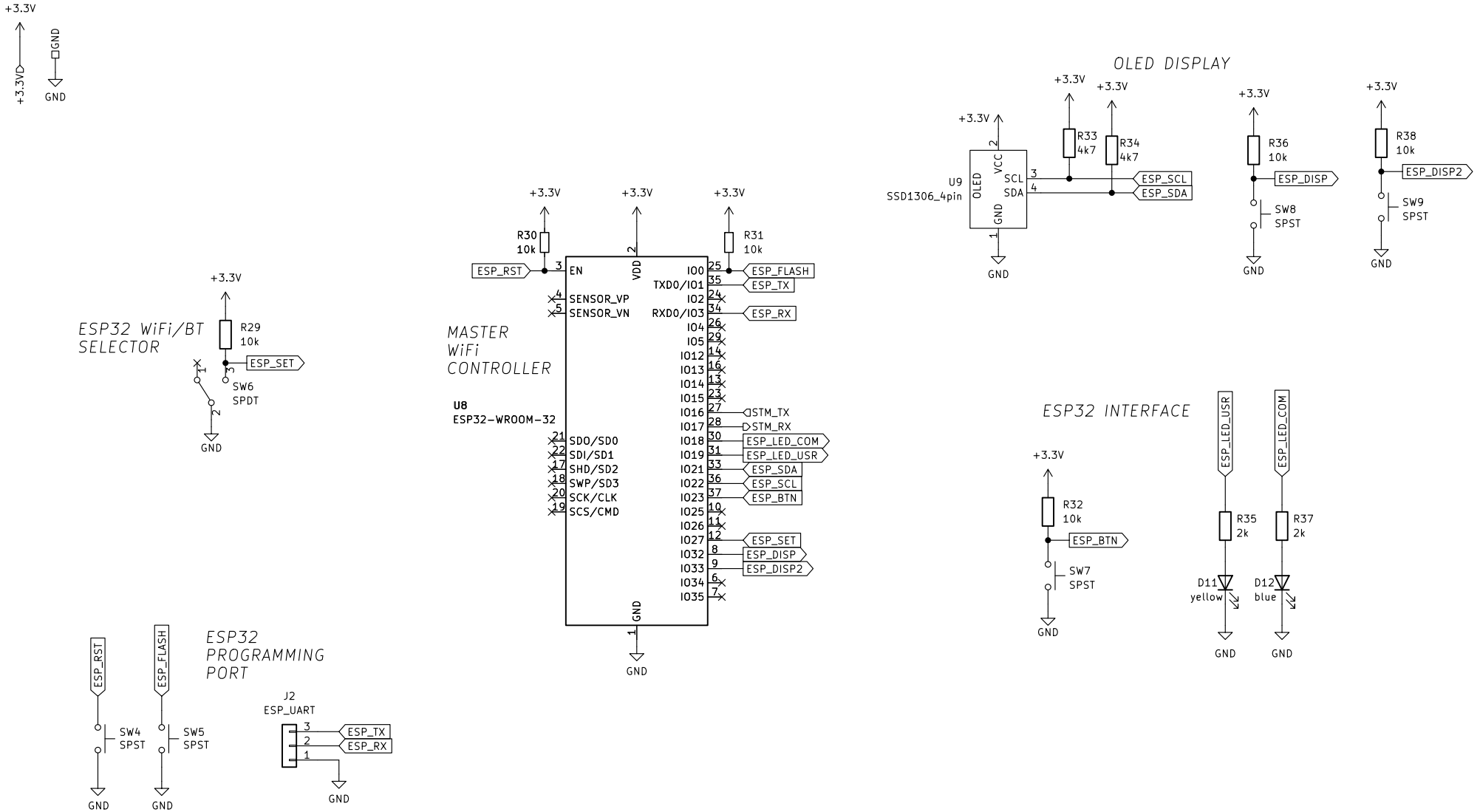


Tomasz Lubelski 253985
ARR – Robotyka
Politechnika Wrocławska

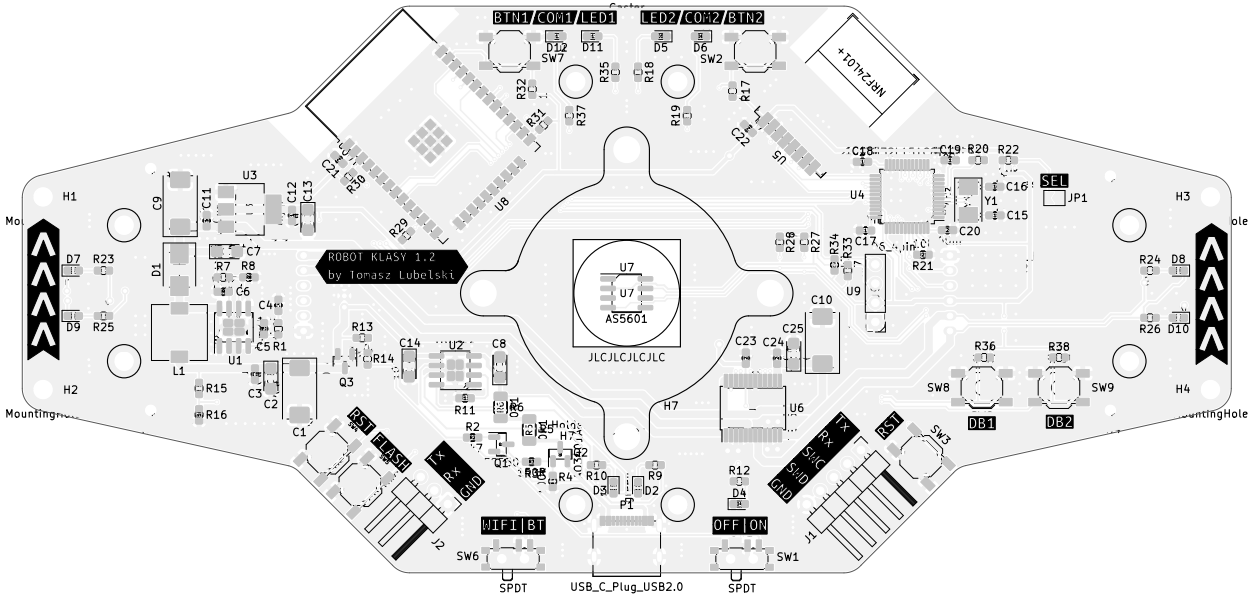
Sheet: /Motors/
File: motors.kicad_sch

Title: Robot Klasy 1.2 – Sekcja wykonawcza

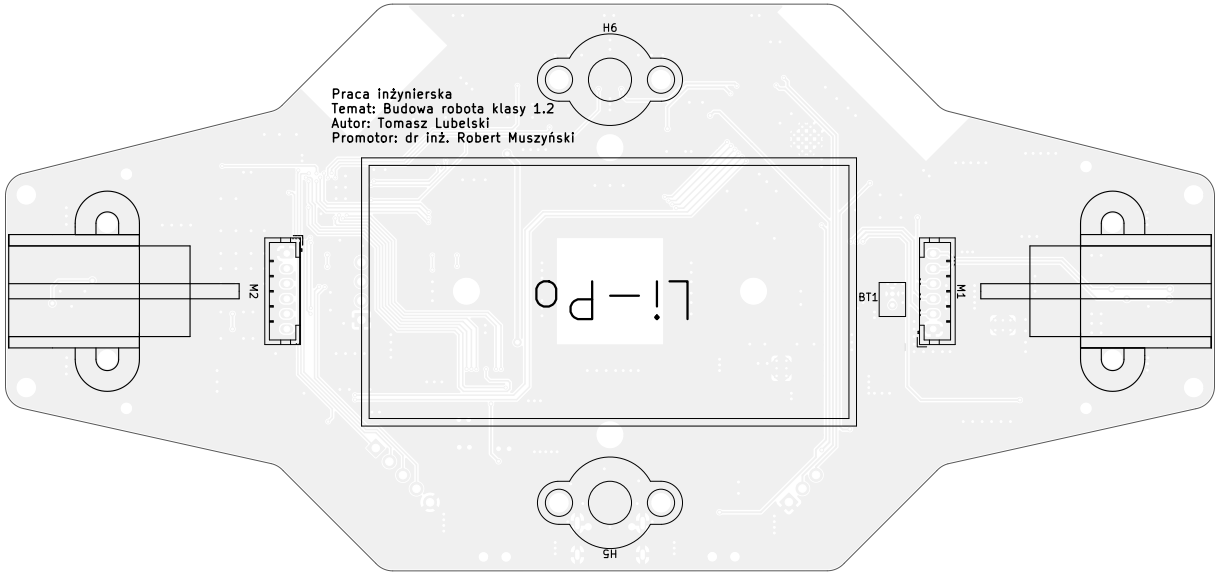
Size: A4	Date: 2022-12-22	**Rev: 1.0**
KiCad E.D.A. kicad (6.0.9)		Id: 4/5



Tomasz Lubelski 253985		
ARR – Robotyka		
Politechnika Wrocławska		
Sheet: /Communication/		
File: communication.kicad_sch		
Title: Robot Klasy 1.2 – Sekcje komunikacji zewnętrznej		
Size: A4	Date: 2022-12-22	**Rev: 1.0**
KiCad E.D.A. kicad (6.0.9)		Id: 5/5



Sheet: BlackAndWhiteTop		
File: platform1.2-electronics.kicad_pcb		
Title:		
Size: A4	Date:	**Rev:**
KiCad E.D.A. kicad (6.0.9)		Id: 1/1



Sheet: BlackAndWhiteBottom File: platform1.2-electronics.kicad_pcb		
Title:		
Size: A4	Date:	**Rev:**
KiCad E.D.A. kicad (6.0.9)		Id: 1/1

Dodatek C

Galeria

Rysunek C.1 Robot zakrzywia czasoprzestrzeń swoją prędkością (koloryzowane)