

**POLITECHNIKA WARSZAWSKA
WYDZIAŁ ELEKTRYCZNY
INSTYTUT STEROWANIA
I ELEKTRONIKI PRZEMYSŁOWEJ**

**PRACA DYPLOMOWA INŻYNIERSKA
na kierunku AUTOMATYKA I ROBOTYKA**



Mateusz Staszków
Nr ind. 261006

Rok akad.: 2016/2017
Warszawa, 1 października 2016

Dwukołowy robot inspekcjny

Zakres pracy:

1. Przegląd istniejących rozwiązań
2. Projekty układu
3. Budowa robota
4. Implementacja programów sterujących
5. Testy i analiza

*(Podpis i pieczętka
Kierownika Zakładu
Dydaktycznego)*

Kierujący pracą: dr inż. Krzysztof Bieńkowski

Termin wykonania: 31 maja 2017
Praca wykonana i zaliczona pozostaje
własnością Instytutu i nie będzie
zwrócona wykonawcy

Warszawa, dnia 31 maja 2017r.

Politechnika Warszawska
Wydział Elektryczny

OŚWIADCZENIE

Świadom odpowiedzialności prawnej oświadczam, że niniejsza praca dyplomowa inżynierska pt. Dwukołowy robot inspekcyjny:

- została napisana przeze mnie samodzielnie,
- nie narusza niczyich praw autorskich,
- nie zawiera treści uzyskanych w sposób niezgodny z obowiązującymi przepisami.

Oświadczam, że przedłożona do obrony praca dyplomowa nie była wcześniej podstawą postępowania związanego z uzyskaniem dyplomu lub tytułu zawodowego w uczelni wyższej. Jestem świadom, że praca zawiera również wyniki stanowiące własności intelektualne Politechniki Warszawskiej, które nie mogą być udostępniane innym osobom i instytucjom bez zgody Władz Wydziału Elektrycznego.

Oświadczam ponadto, że niniejsza wersja pracy jest identyczna z załączoną wersją elektroniczną.

Mateusz Staszków.....

Spis treści

1 Wstęp	1
1.1 Postawienie zadania	2
2 Mechanika	4
2.1 Parametry ogólne	4
2.2 Model 3D	5
2.3 Obudowa	5
2.4 Napęd	7
2.5 Koła	9
2.6 Matematyczny model układu	10
2.7 Wyznaczenie równań liniowych układu	13
3 Elektronika	15
3.1 Zasilanie	15
3.1.1 Akumulator	15
3.1.2 Ładowarka	18
3.2 Zabezpieczenie elektroniki	19
3.3 Sterownik silników prądu stałego	19
3.3.1 Mikrokontroler ATMega168PA-PU	21
3.3.2 Dwukanałowy mostek H - układ L298N	25
3.3.3 Enkodery magnetyczne CPR 48	27
3.3.4 Wykonanie płytki PCB	28
3.4 Układ regulacji położenia i skanowania pomieszczeń	31
3.4.1 Mikrokontroler STM32F103RBT6	32
3.4.2 Żyroskop i akcelerometr - układ MPU6050	34
3.4.3 Czujniki ultradźwiękowe HC-SR04	37
3.4.4 Moduł WiFi ESP-01 8266	41
3.4.5 Wykonanie płytki PCB	43

4 Teoria Sterowania	46
4.1 Wyznaczenie parametrów układu	46
4.2 Projekt regulatora liniowo-kwadratowego (LQR)	47
5 Oprogramowanie	50
5.1 Projekt sterowania położenia w pionie (Matlab, Simulink)	50
5.2 Sterownik silników prądu stałego (C, AVR)	51
5.3 Obsługa czujników i sterowanie położenia w pionie (C, StdPeriph)	52
5.3.1 Plik HAL_MPU6050.h	53
5.3.2 Plik esp8266.c	53
5.3.3 Plik main.c	53
5.4 Obsługa modułu WiFi, serwer WWW (C++, HTML, CSS, JS, XML, Arduino)	56
5.4.1 Kod C++	56
5.4.2 Kod HTML, CSS, JavaScript i XML	57
6 Wnioski	59
Bibliografia	60

Podziękowania

Dziękuję bardzo serdecznie Rodzinie, Wykładowcom, Władzom Politechniki i Promotorowi - dr inż. Krzysztofowi Bieńkowskemu

Mateusz Staszków

Rozdział 1

Wstęp

Roboty inspekcyjne znajdują zastosowanie najczęściej w miejscach trudno dostępnych dla człowieka. Można również spotkać małe roboty przeszukujące korytarze w biurowcach w godzinach nocnych. Ich zadaniem jest odwiedzenie lub zeskanowanie każdego miejsca w budynku i upewnienie się czy wszystko jest zachowane w ustalonym porządku.

Problem stabilizacji położenia w pionie dotyczy wielu gałęzi przemysłu, na przykład takich jak: produkcja jednoosobowych transportowych pojazdów typu Segway na lotniskach i w centrach handlowych, układy balansujące na nierównych powierzchniach, czy nawet roboty humanoidalne.

Połączenie tych dwóch cech czyni robota pomocnym w trudno dostępnych miejscach zarówno jeżeli chodzi o toksyczność jak i bardzo nieregularną powierzchnię jezdnią.

Dwukołowy robot balansujący działa na zasadzie wahadła odwróconego. Jego zadaniem jest ciągłe utrzymywanie pionu z możliwie jak najmniejszymi odchyłami. Jest to układ niestabilny i wymaga automatycznej regulacji - zadanie to spełnia regulator liniowo-kwadratowy-Gaussa.

W układzie za sprzężenia zwrotne odpowiadają czujniki położenia, a dokładniej przyspieszenia - akcelerometr i prędkości kątowej - żyroskop. Do sterowania silnikami jest użyte sprzężenie zwrotne w postaci miernika położenia kół przez enkodery magnetyczne. Elektronika znajduje się na wytrawionych płytach PCB. Układ jest zasilany z akumulatora litowo-polimerowego. Robot jest zaprojektowany tak, aby każdy element był jak najlżejszy i można było go łatwo wymienić. Stąd duża ilość śrub i mocowań na wcisk. Przy każdym mikrokontrolerze znajduje się złącze programatora, co umożliwia bezproblemowe aktualizowanie oprogramowania. Do komunikacji robota z otoczeniem służą czujniki ultradźwiękowe. Moduł WiFi umożliwia komunikację z komputerem PC i odczyt wszystkich pomiarów.

1.1 Postawienie zadania

Zadaniem pracy jest przedstawienie konstrukcji autonomicznego i balansującego robota inspekcjnego oraz przedstawienie działania za pośrednictwem wykresów i tabel.

Budowa robota to złożone zadanie i wymaga podzielenia na mniejsze elementy konstrukcji:

1. Przeczytanie literatury na temat:
 - balansujących i inspekcyjnych robotów komercyjnych,
 - dokumentacji technicznych mikrokontrolerów serii AVR i ARM, czujników ultradźwiękowych, akcelerometrów i żyroskopów, modułów WiFi, enkoderów, silników,
 - dekodowania sygnałów z wyżej wymienionych czujników,
 - szeregowej komunikacji przewodowej i bezprzewodowej,
 - druku 3D,
 - technologii wytwarzania płyt ekologicznych,
2. Zaprojektowanie:
 - modelu matematycznego i modelu sterowania w środowisku Matlab i Simulink,
 - elektroniki w środowisku Cadsoft EAGLE,
 - modelu 3D w programie AutoCAD, podzielenie na części i wyeksportowanie do formatu, STL,
3. Zakup elementów elektronicznych,
4. Wydrukowanie części na drukarce 3D i sklejenie ich acetonem,
5. Wytrawienie płyt PCB w nadsiarczanie sodu,
6. Lutowanie elementów elektronicznych do płyt,
7. Skonfigurowanie środowisk programistycznych,
8. Wykonanie szkieletów kodu,
9. Wzajemne skomunikowanie mikrokontrolerów,
10. Oprogramowanie czujników,
11. Implementacja:
 - sterownika silników DC,
 - komunikacji mikrokontrolera z komputerem PC,
 - regulacji położenia w pionie (LQG),

- systemu skanowania pomieszczeń,
12. Testy i analiza,
 13. Wnioski.

Rozdział 2

Mechanika

Robot został zaprojektowany tak, aby był jak najlżejszy. Ułatwia to sterowanie jego położeniem przy odpowiednio dobranych silnikach. Środek ciężkości znajduje się jak najwyżej, co przekłada się na ułatwienie sterowania położenia w pionie - układ gwałtowniej odpowiada na zadane wartości.

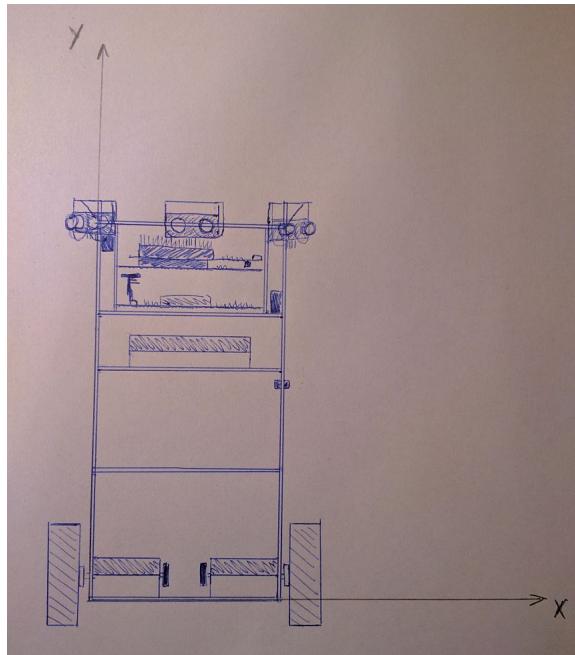
2.1 Parametry ogólne

Masa: 1 kg

Wymiary: 95 mm x 180 mm x 306 mm

Wysokość środka masy: 162.6 mm

Moment bezwładności dla osi położonej w środku masy: 0.01 kgm^2



Rysunek 2.1: Koncepcja obudowy robota [opracowanie własne]

2.2 Model 3D

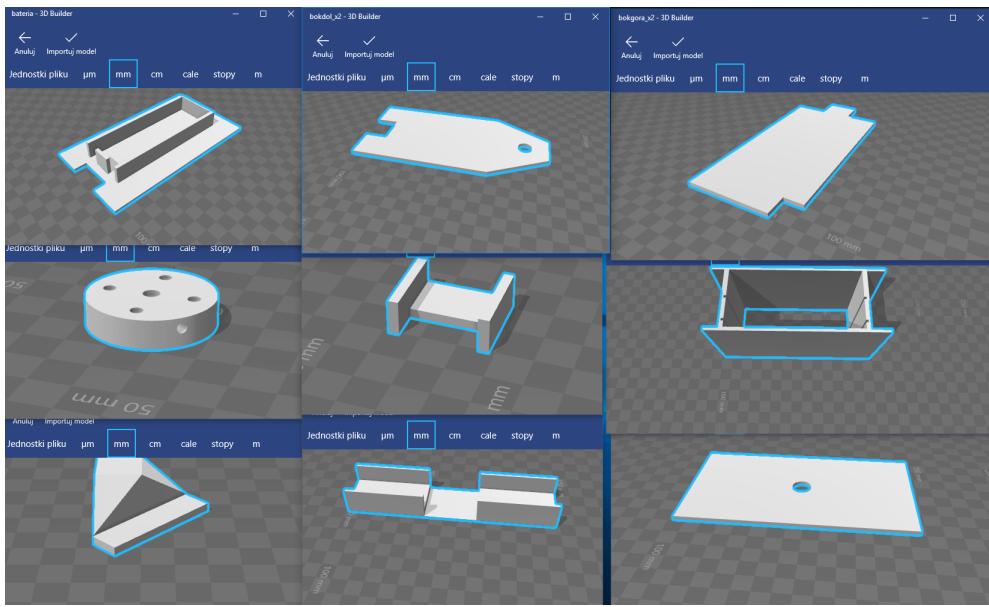
Model został zbudowany w programie Autodesk AutoCAD 2017. Następnie został podzielony na części w taki sposób, aby wydruk 3D był jak najmniej kłopotliwy. Każda część jest nie większa niż 100 mm x 180 mm x 80 mm. Następnie modele zostały wyeksportowane do formatu STL.

2.3 Obudowa

Masa: 402 g

Wymiary: 95 mm x 180 mm x 306 mm

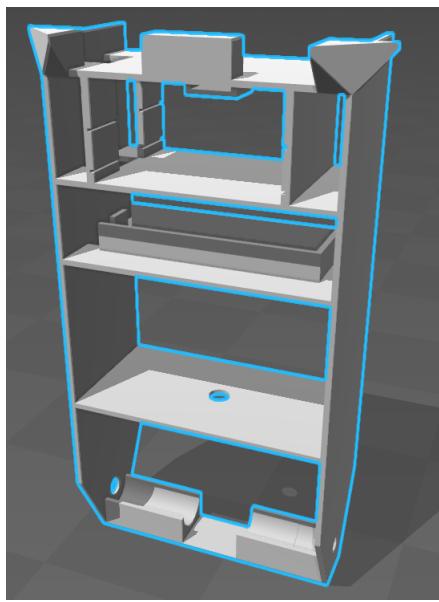
Obudowa została w całości wydrukowana na drukarce 3D, a poszczególne elementy są połączone acetonom. Proces adhezyjnego klejenia części jest możliwy poprzez użycie jako materiału do drukowania tworzywa ABS - kopolimeru akrylonitrylo-butadieno-styrenowego, który reagując z acetonom rozpuszcza się, umożliwiając, poprzez zbliżenie do niego innego materiału na przyklejenie się.



Rysunek 2.2: Model 3D podzielony na części i wyeksportowany do STL [opracowanie własne]

Obudowa składa się z czterech segmentów - kolejno od dołu:

- segment silników - odpowiadający za napęd robota, silniki są przykręcane do ścian bocznych za pomocą śrub i znajdują się w specjalnie zaprojektowanych miejscach, aby ograniczyć ich ruch. Takie zamontowanie umożliwia ich łatwą i szybką wymianę. W ścianach bocznych umieszczone zostały, w których znajdują się wały silników,
- część akumulatora - odpowiadająca za zasilanie, akumulator ma swój koszyk, co umożliwia łatwą wymianę i odpowiednie usztywnienie paku ogniw,
- część elektroniczna: ściany z otworami na wciśnięcie płytki PCB i z koszykiem na żyroskop/akcelerometr na spodzie najwyższej półki,
- część komunikacyjna: znajdują się tu miejsca na 3 czujniki ultradźwiękowe oraz na moduł Wi-Fi.



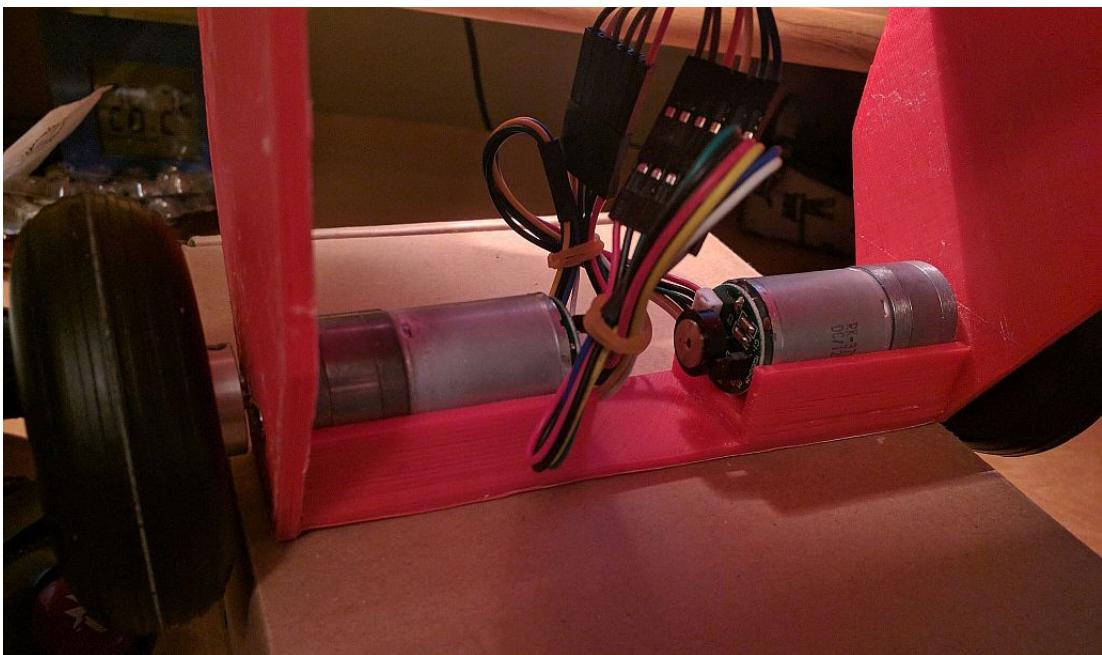
Rysunek 2.3: Cała obudowa zaprojektowana w programie Autodesk AutoCAD i wyeksportowana do pliku STL [opracowanie własne]

2.4 Napęd

Do napędu robota zostały użyte silniki prądu stałego firmy Pololu na napięcie 12 V z przekładnią 9.68 : 1 (25Dx48L) i enkoderem CPR 48, których wały opierają się na łożyskach kulkowych umieszczonych w ścianach bocznych.

Parametry silnika:

- masa: 0.095 kg,
- wymiary: front $\phi 25$ mm, długość korpusu 78.7 mm,
- moment obrotowy: 0.15 Nm,
- prędkość obrotowa: 770 obr/min,
- średni pobór prądu: 200 mA,
- maksymalny pobór prądu: 2100 mA,
- rozdzielczość enkodera: 48 impulsów na obrót (po przekładni 464).



Rysunek 2.4: Silnik DC Pololu 12 V [opracowanie własne]

Do wału silnika zostały dopasowane łożyska kulkowe o średnicach 4 mm i 12 mm oraz grubości 4 mm. Dzięki nim następuje bezpieczniejsze przełożenie napędu na koła z minimalnym tarciem. Obniżają nacisk na wał silnika.



Rysunek 2.5: Łożysko kulkowe [źródło: www.allegro.pl, aukcja sklepu TECH-POL-net]

2.5 Koła

W konstrukcji zostały użyte lekkie poliuretanowe koła z plastikową piastą, charakteryzują się niską ceną i dostateczną przyczepnością.
Parametry koła:

- średnica opony: 70 mm,
- szerokość opony: 24 mm ,
- średnica otworu: 4 mm ,
- średnica plastikowej piasty: 35 mm ,
- waga: 30g.

Do połączenia kół z krótkim wałem silnika zostały skonstruowane specjalne łączniki na śruby. Ich działanie polega na zaciśnięciu bocznej śruby o średnicy 2,5 mm do granic możliwości. Wymagane jest w tym momencie specjalne ułożenie, w środku łącznika, wału silnika, którego przekrój nie jest idealnym okręgiem, lecz zawiera delikatne zeszlifowanie (kształtem przypomina literę D). Po dociśnięciu śruby w miejsce tego szlifu jakikolwiek luz zostanie wyeliminowany.



Rysunek 2.6: Hub mocujący wał silnika z kołem [opracowanie własne]

Po testach wytrzymałościowych okazało się, że plastikowe tworzywo ABS nie wytrzymało dużych naprężeń śruby na wał silnika i uległo odkształceniemu, dlatego zostały zastosowane uniwersalne aluminiowe huby mocujące z otworem na 4 mm.

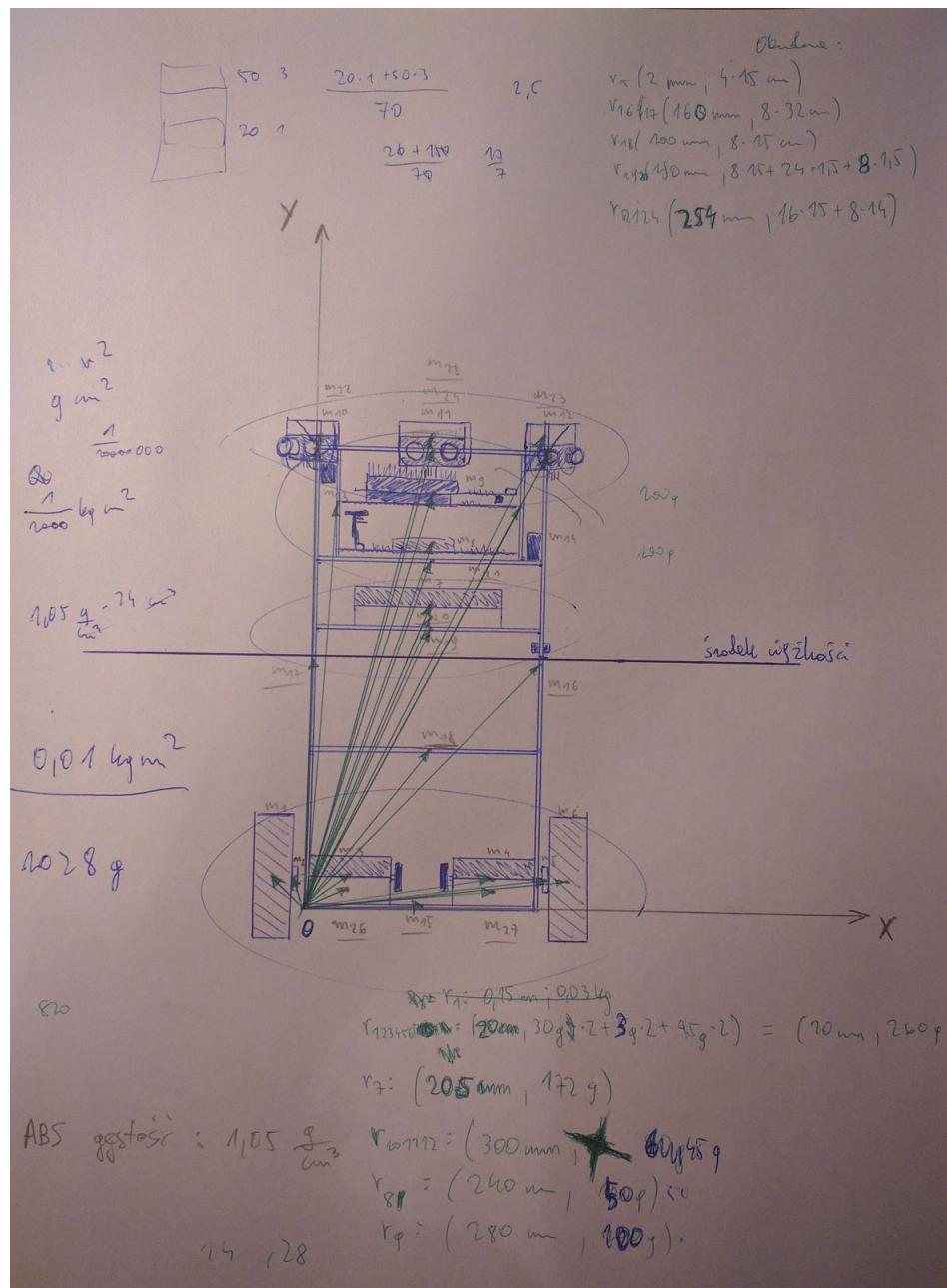


Rysunek 2.7: Aluminiowy hub mocujący [źródło: https://botland.com.pl/6039-thickbox_default/aluminiowy-hub-mocujacy-4mm-m3-2-szt.jpg]

2.6 Matematyczny model układu

Robot jest zbudowany symetrycznie w odniesieniu do szerokości i długości, więc po zastosowaniu uproszczenia środek masy jedynie w odniesieniu do wysokości wymaga przeprowadzenia dokładnych obliczeń. Reszta wartości będzie się pokrywała ze środkiem geometrycznym bryły. Przy obliczeniach położenie i masa przewodów zostały ujęte w dużym przybliżeniu.

Środek masy i moment bezwładności zostały obliczone poprzez podzielenie robota na obiekty i wyliczenie dla każdego środka masy i momentu bezwładności, a następnie zsumowanie wszystkich składowych w programie Microsoft Excel 2013.



Rysunek 2.8: Wektory średnich geometrycznych poszczególnych obiektów [opracowanie własne]

Obliczone dane bryły:

- wysokość środka masy obudowy: 162.6 mm,
 - moment bezwładności obudowy: $0.01 \text{ kg} * \text{m}^2$.

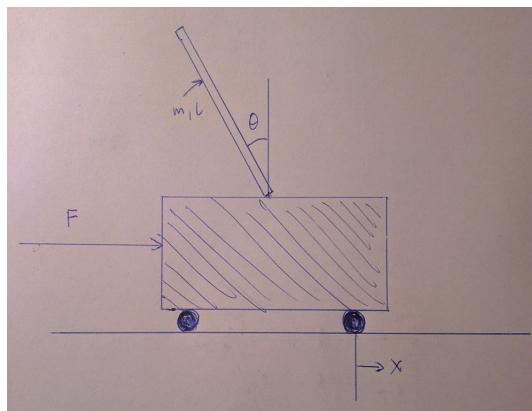
Obliczenia wykonane w programie Microsoft Excel 2013:

Pole fragmentu ABS [cm ²]	Wysokość [cm]	Objętość ABS [cm ³]	Masa [g]	Odgległość od środka ciężkości [cm]	Moment bezwładności [g*cm ²]	Środek ciężkości obudowy [cm]
60	16	24	25,2	16,05747287	6497,623964	18,24351464
256	16	102,4	107,52	0,257472874	7,127746017	
120	10	48	50,4	6,257472874	197,460725	
168	19	67,2	70,56	2,742527126	530,7138675	
352	25,4	140,8	147,84	9,142527126	12357,2501	
			401,52		21366,25671	
Masa elementu [g]						Wypadkowy środek ciężkości elementów [cm]
260	2	14,25747287	52651,63851			14,98564593
172	20,5	4,242527126	3095,834264			
45	30	13,74252713	8495,567332			
50	24	7,742527126	2997,336315			
100	28	11,74252713	13788,694333			
627			81232,07076			
1028,52						Środek ciężkości robota [cm]
						16,25747287
Moment bezwładności robota [kg*m ²]						
						0,010259833

Rysunek 2.9: Obliczenie środka ciężkości i momentu bezwładności robota w programie Microsoft Excel 2013 [opracowanie własne]

2.7 Wyznaczenie równań liniowych układu

Jest to układ odwróconego wahadła, który przypomina patyk znajdujący się na wózku, na który działa siła z zewnątrz. Układ taki wygląda następująco:



Rysunek 2.10: Układ wahadła odwróconego [opracowanie własne]

Równania opisujące model:

$$F = (M + m)\ddot{x} + b\dot{x} + ml\ddot{\theta}\cos\theta - ml\dot{\theta}^2\sin\theta$$

$$-ml\ddot{x}\cos\theta = (I + ml^2)\ddot{\theta} + mg\sin\theta$$

Dane układu:

- M - masa wózka: 0 kg (wózek służył tylko do zilustrowania sytuacji),
- m - masa odwróconego wahadła: 1 kg,
- I - moment bezwładności odwróconego wahadła: 0.01 kgm^2 ,
- F - siła działająca na wózek: nie większa niż 4 N (założenie),
- x - położenie wózka - zmienna stanu mierzona w metrach,
- θ - wychylenie wahadła od pionu - zmienna stanu mierzona w stopniach,
- b - współczynnik tarcia wózka o podłoże: ok. 0.1 N/m/s ,
- l - odległość od środka masy: 0.137 m,
- g - przyspieszenie ziemskie w Warszawie: 9.8123 m/s^2 .

W przypadku robota balansującego na dwóch kołach równanie upraszczacza się do postaci:

$$\begin{aligned} F &= m\ddot{x} + b\dot{x} + ml\ddot{\theta}\cos\theta - ml\dot{\theta}^2\sin\theta \\ -ml\ddot{x}\cos\theta &= (I + ml^2)\ddot{\theta} + mg\sin\theta \end{aligned}$$

Powysze równania są nieliniowe, więc po linearyzacji w otoczeniu punktu $\theta = \pi$ otrzymamy następujące równania ($\theta = \pi + \phi$, gdzie ϕ reprezentuje małe odchylenia wahadła, a u to wejście układu):

$$\begin{aligned} u &= m\ddot{x} + b\dot{x} - ml\ddot{\phi} \\ ml\ddot{x} &= (I + ml^2)\ddot{\phi} - mgl\phi \end{aligned}$$

Opis układu w przestrzeni zmiennych stanu:

$$\begin{aligned} \dot{x} &= Ax + Bu \\ y &= Cx + Du \end{aligned}$$

$$\begin{aligned} \begin{bmatrix} \dot{x} \\ \ddot{x} \\ \dot{\phi} \\ \ddot{\phi} \end{bmatrix} &= \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & \frac{-(I+ml^2)b}{Im} & \frac{mgl^2}{I} & 0 \\ 0 & 0 & 0 & 1 \\ 0 & \frac{-lb}{I} & \frac{mgl}{I} & 0 \end{bmatrix} \begin{bmatrix} x \\ \dot{x} \\ \phi \\ \dot{\phi} \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{I+ml^2}{Im} \\ 0 \\ \frac{l}{I} \end{bmatrix} u \\ y &= \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x \\ \dot{x} \\ \phi \\ \dot{\phi} \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \end{bmatrix} u \end{aligned}$$

Po wstawieniu liczb:

$$\begin{aligned} \begin{bmatrix} \dot{x} \\ \ddot{x} \\ \dot{\phi} \\ \ddot{\phi} \end{bmatrix} &= \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & -0.2877 & 18.4167 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & -1.37 & 134.4285 & 0 \end{bmatrix} \begin{bmatrix} x \\ \dot{x} \\ \phi \\ \dot{\phi} \end{bmatrix} + \begin{bmatrix} 0 \\ 2.8769 \\ 0 \\ 13.7 \end{bmatrix} u \\ y &= \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x \\ \dot{x} \\ \phi \\ \dot{\phi} \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \end{bmatrix} u \end{aligned}$$

Rozdział 3

Elektronika

3.1 Zasilanie

3.1.1 Akumulator

Do zasilania został użyty pakiet ogniw litowo-polimerowych firmy Redox. Akumulator został wybrany ze względu na to, że dostarcza dużo energii przy małych rozmiarach, a także może być stale obciążony dużym prądem. Dane techniczne:

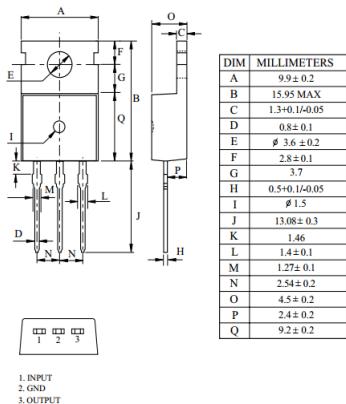
- napięcie nominalne: 11,1 V (pełne naładowanie - 12,6 V),
- pojemność: 2200 mAh,
- prąd rozładowania: 30 C (66 A),
- wymiary: 115 x 34 x 20 mm,
- masa: 172 g.

W najbardziej pesymistycznym przypadku robot może pobierać do 5 A ciągłego prądu. Przy pojemności 2200 mAh daje to minimalny czas pracy około 26 minut. Natomiast w przypadku zwykłej jazdy, bez przeszkód robot może pobrać ok. 1 A, co przekłada się na pięciokrotnie dłuższe działanie. W 130 minut powinno dać się zmapować kilka razy w ciągu nocy całe piętro budynku.

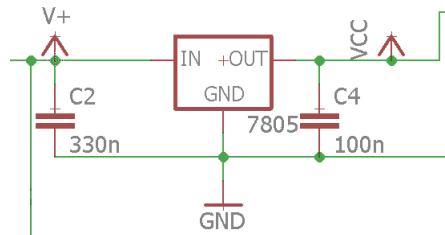


Rysunek 3.1: Akumulator litowo-polimerowy firmy Redox [opracowanie własne]

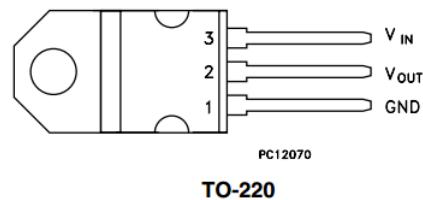
Napięcie z akumulatora jest bezpośrednio przekazywane tylko do silników (na dwukanałowy mostek), a na układy logiczne dostarczane jest poprzez stabilizatory liniowe 7805 na 5 V i LD1117VC33 na 3.3 V. Stabilizatory są w obudowach TO-220, co ułatwia ich chłodzenie. Zasilają części logiczne, więc wbudowana blaszka odprowadzająca ciepło wystarczy, żeby cały układ się nie przegrzał. Do stabilizatorów zostały dołączone kondensatory MKT 330 nF i ceramiczne 100 nF zgodnie z notą katalogową.



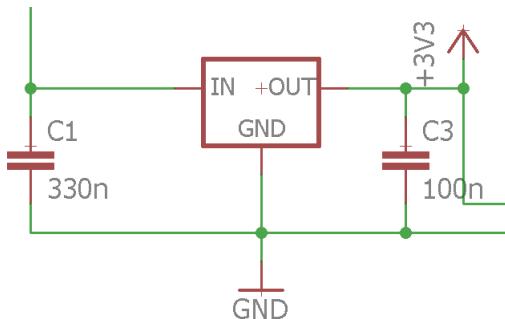
Rysunek 3.2: Układ 7805 [źródło: bibliografia pkt. 1, strona 1]



Rysunek 3.3: Schemat połączeń układu 7805, program EAGLE [opracowanie własne]



Rysunek 3.4: Układ LD1117VC33 [źródło: bibliografia pkt. 2, strona 3]



Rysunek 3.5: Schemat połączeń układu LD1117VC33, program EAGLE [opracowanie własne]

Napięcie 5 V jest kierowane do sterownika silników z mikrokontrolerem ATmega168PA-PU, a także do zasilania płytka NUCLEO z mikrokontrolerem STM32F103RBT6. Napięcie 3,3 V służy do zasilenia niektórych czujników i układów, na przykład modułu WiFi ESP8266.

3.1.2 Ładowarka

Ładowanie pakietów ogniw litowo-polimerowych wymaga stosowania ładowarek mikroprocesorowych z odpowiednim algorytmem ładowania, wyposażonych w układy balancerów, które równomiernie ładują każde ogniwo. W celu uniknięcia trwałych uszkodzeń ogniw powinno się nie rozładowywać go poniżej 3V, czyli w przypadku całego pakietu napięcie nie powinno osiągnąć wartości poniżej 9 V.

W celu zachowania powyższych zasad została zastosowana ładowarka sieciowa firmy Redox. Specyfikacja:

- Napięcie wejściowe: 100 - 240 V AC 50 / 60 Hz,
- Prąd ładowania: do 1000 mA,
- Obsługiwane pakiety: 2-3 ogniowe litowo-polimerowe (7,4 V i 11,1 V),
- Wbudowany balancer ogniw,
- Długość przewodów: 150 cm,
- Wymiary: 96 x 55 x 33 mm.



Rysunek 3.6: Ładowarka sieciowa Redox [opracowanie własne]

3.2 Zabezpieczenie elektroniki

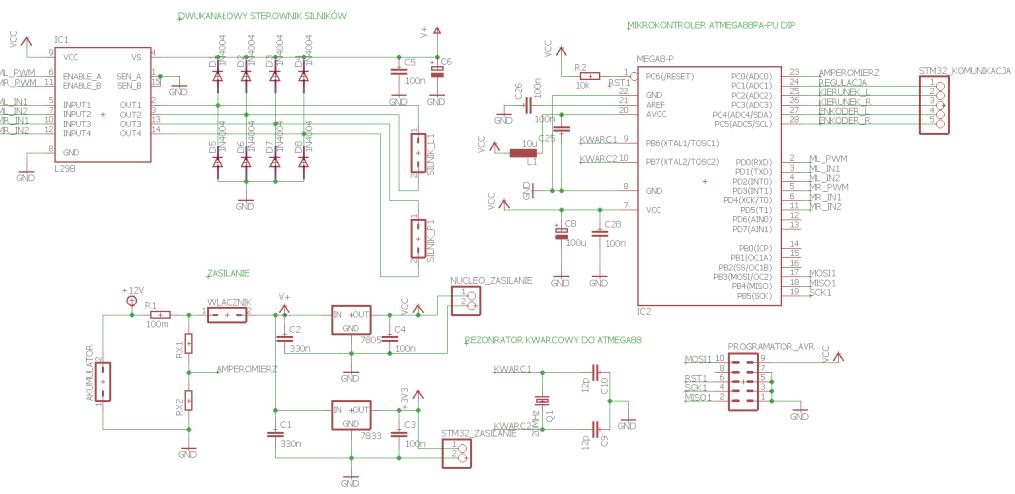
Elektronika może ulec uszkodzeniu ze względu na wiele czynników: silne pole magnetyczne, wysoka temperatura, za duży prąd lub napięcie. W układzie zostało zastosowanych kilka zabezpieczeń:

- Akumulator jest podłączony do przełącznika na prąd do 15 A, który służy jako główny włącznik robota,
- Dodatni sygnał z akumulatora przechodzi przez szeregowo wpięty rezystor $0,1 \Omega$ 5 W, który ogranicza przepływ prądu do 7 A, ze względu na swoją wytrzymałość. Służy on również do pomiaru prądu przez przetwornik analogowo-cyfrowy mikrokontrolera ATMega168PA-PU, jeżeli prąd osiągnie kosztownie wysoką wartość, to układ scalony ma możliwość obniżenia go do bezpiecznej normy,
- do dwukanałowego mostka w sterowniku silników, został dołączony radiator w formie dwóch lekkich blaszek aluminiowych (fragmentów ościeżnicy)

3.3 Sterownik silników prądu stałego

Do sterowania silników DC został wybrany mikrokontroler ATmega168PA-PU w obudowie DIP28. Sterowanie odbywa się za pomocą nadawania sygnału

PWM na piny dwukanałowego mostka L298N. Do układu mostków i mikrokontrolera zostały dołączone filtry poprzez umieszczenie kondensatorów i dławików zgodnie z notami katalogowymi. Dodatkowo do L298N zostały dołączone diody, które były wymagane do prawidłowego działania układu. Do łatwego przeprogramowywania mikrokontrolera na płytce znajduje się złącze programatora zgodnego z USBASP o 10 pinowym standardzie KANDA.



Rysunek 3.7: Sposób połączenia elementów elektronicznych sterownika silników prądu stałego wykonany w programie Cadsoft EAGLE [opracowanie własne]

3.3.1 Mikrokontroler ATMega168PA-PU

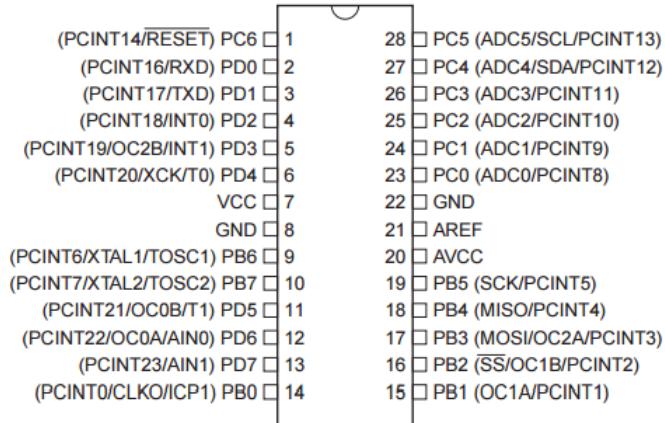
Mikrokontroler ATMega168PA-PU 8-bitowy z rodziny AVR w obudowie DIP28 został wybrany ze względu na niskie zapotrzebowanie mocy obliczeniowej w układzie sterownika silników, a co się z tym wiąże układ może być bardzo tani.

Dane techniczne:

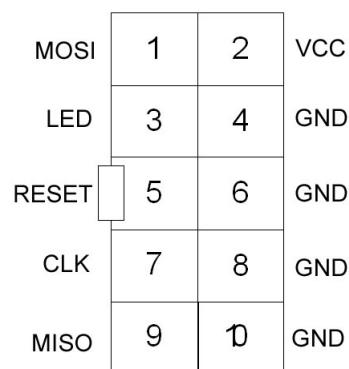
- taktowanie: do 20 MHz,
- pamięć Flash: 16 kB,
- pamięć RAM: 1 kB,
- 23 linie wyjścia/wejścia,
- 2 timer 8-bitowe,
- 1 timer 16-bitowy,
- 6 kanałów 10-bitowego przetwornika analogowo-cyfrowego,
- sprzętowe interfejsy komunikacyjne: USART, SPI, 2-wire (I2C).



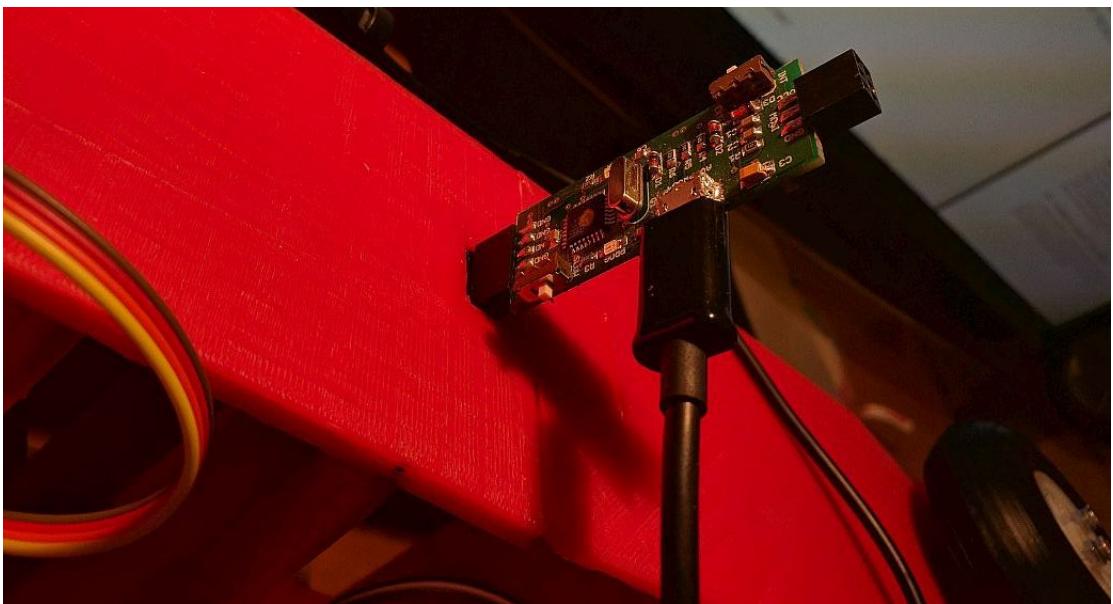
Rysunek 3.8: Mikrokontroler ATMega168PA-PU w obudowie DIP28 [opracowanie własne]



Rysunek 3.9: Rozkład pinów ATMega168PA-PU w obudowie DIP28 [źródło: bibliografia pkt. 3, strona 2]



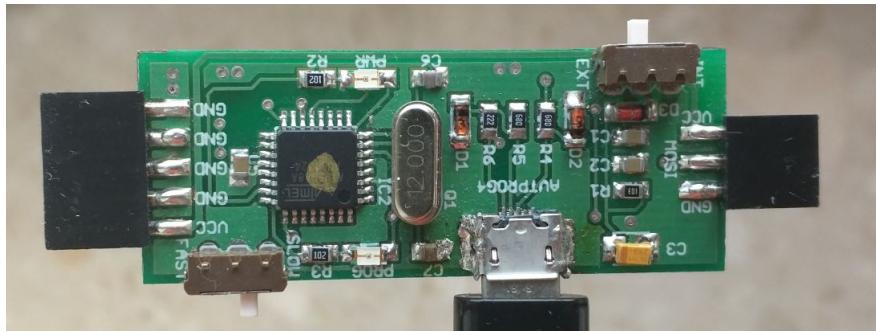
Rysunek 3.10: Schemat gniazda KANDA (złącze programatora mikrokontrolera ATMega168PA-PU) [opracowanie własne]



Rysunek 3.11: Gniazdo programatora AVR wbudowane w obudowę robota [opracowanie własne]

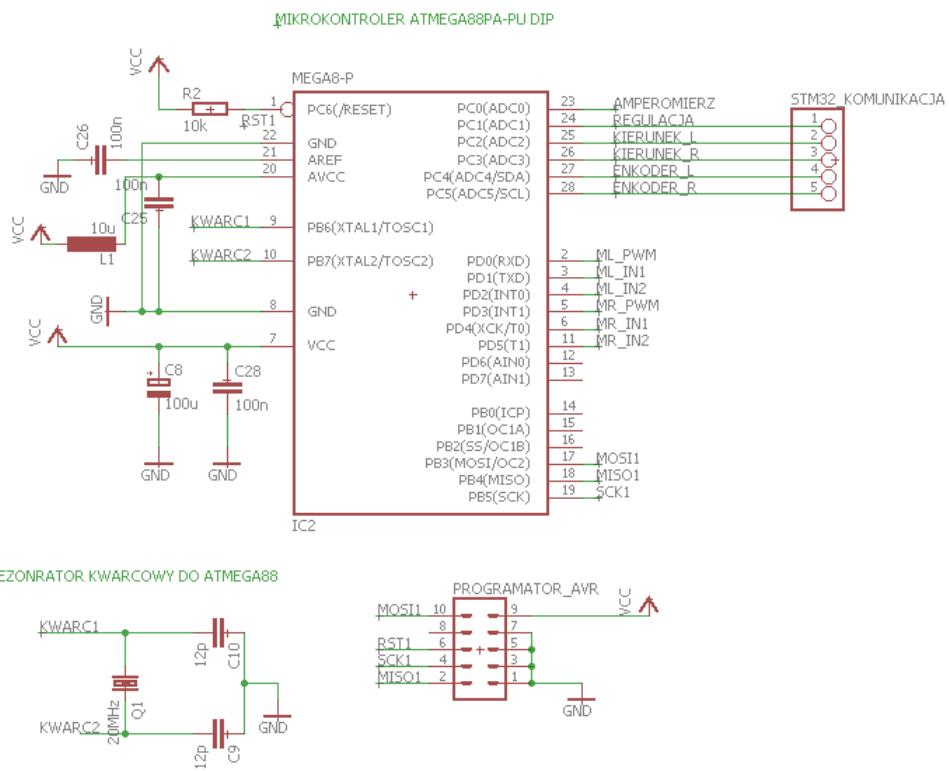
Mikrokontroler jest programowany poprzez dziesięciopinowe gniazdo KANDA połączone z programatorem zgodnym z USBasp. Programator ma wyjście Mini USB, co pozwala na bezproblemowe podłączenie do komputera PC. Do prawidłowego działania układu wymaganych jest kilka dodatkowych elementów, zgodnie z notą katalogową:

- kondensatory ceramiczne 100 nF między piny AREF a GND, VCC a GND i AVCC a GND,
- kondensator elektrolityczny 100 μ F 16 V między piny VCC a GND,
- dławik 10 μ H między piny AVCC i VCC,
- rezystor 10 $k\Omega$ 0,25 W podciągający pin RESET do VCC,
- rezonator kwarcowy 20 MHz do pinów XTAL1 i XTAL2 z dołączonymi kondensatorami ceramicznymi 12 pF.



Rysunek 3.12: Wykorzystany programator zgodny z USBasp [opracowanie własne]

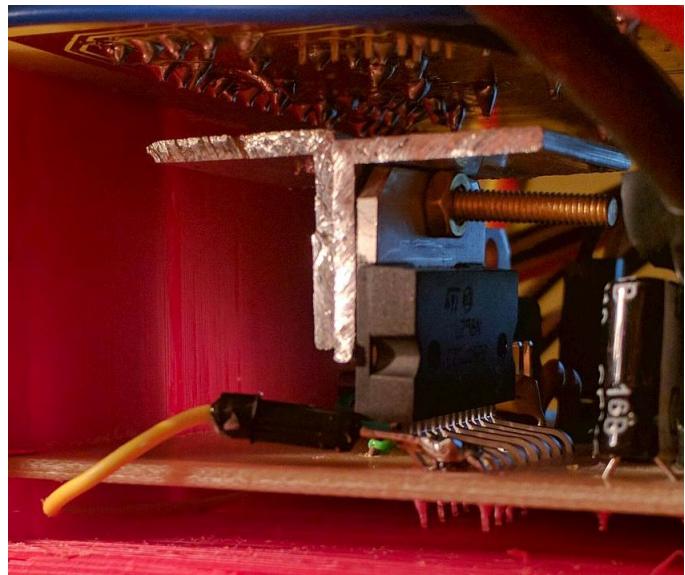
Układ ATMega168PA-PU jest wykorzystywany do sterowania prędkościami silników za pomocą regulatora PID i sprzężenia zwrotnego w postaci odczytu położenia kół z enkoderów magnetycznych znajdujących się przy silnikach. Wartość zadana skorygowana o uchyb jest przekazywana w postaci sygnału PWM na wejścia dwukanałowego mostka H w postaci układu scalonego L298N. Na przetwornik analogowo cyfrowy ADC0 jest podawane napięcie na rezystorze wejściowym $0,1\ \Omega$ wpiętym szeregowo przy dodatnim złączu akumulatora, co pozwala na pomiar całkowitego prądu płynącego w układzie. W przypadku osiągnięcia dużych wartości, pobór prądu zostanie programowo zmniejszony. Mikrokontroler zawiera również kilka połączeń z drugim mikrokontrolerem STM32F103RBT6 za pomocą pinów PC1-PC3, zawierających informacje o kierunku i szybkości jazdy.



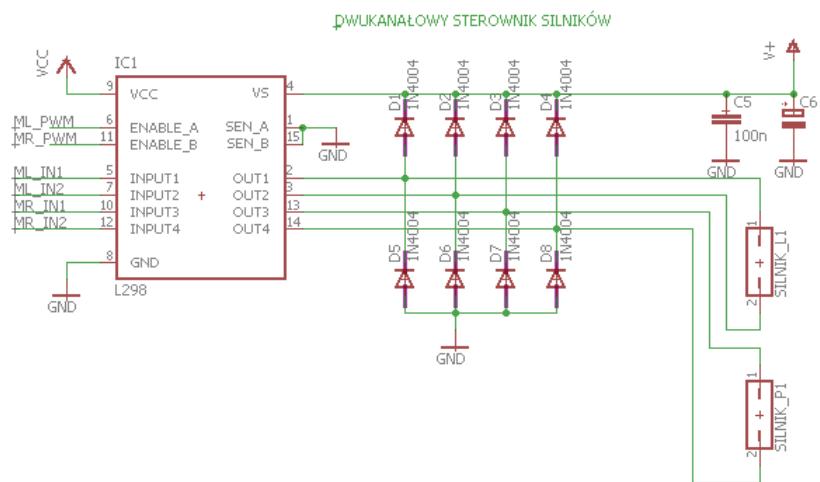
Rysunek 3.13: Sposób połączenia mikrokontrolera ATMeg168PA-PU wykonyany w programie EAGLE [opracowanie własne]

3.3.2 Dwukanałowy mostek H - układ L298N

Sam układ scalony L298N, nie w gotowym module, został zastosowany ze względu na niewielką cenę. Do prawidłowego działania należało jedynie dokupić, według noty katalogowej, 8 diod prostowniczych, 1 kondensator elektrolityczny 100 μ F i 1 kondensator ceramiczny 100 nF.



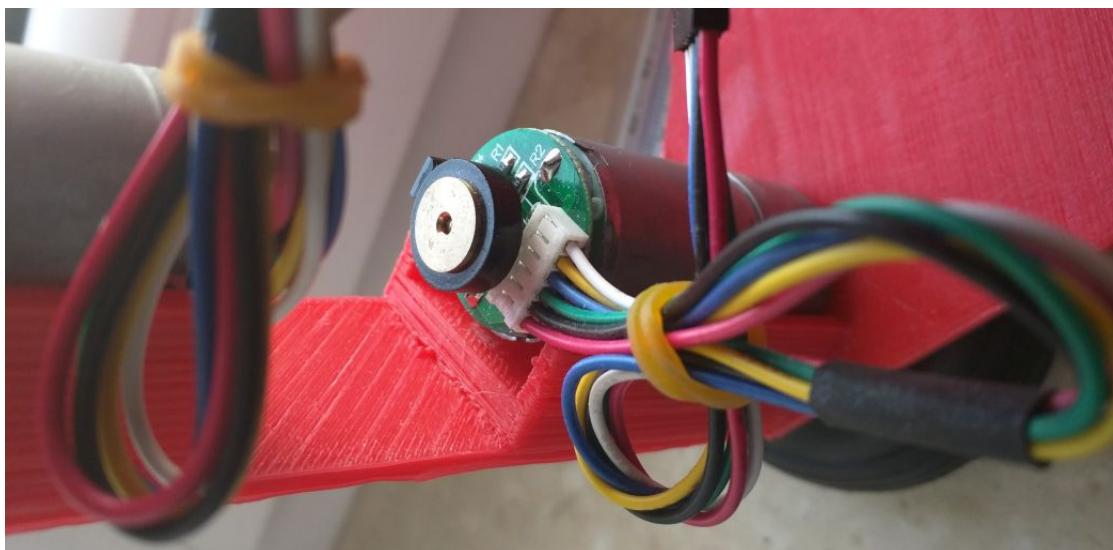
Rysunek 3.14: Układ scalony L298N z aluminiowymi radiatorami [opracowanie własne]



Rysunek 3.15: Schemat połączenia układu L298N wykonany w programie EAGLE [opracowanie własne]

3.3.3 Enkodery magnetyczne CPR 48

Zastosowane enkodery są zintegrowane z używanymi silnikami. Ich działanie jest oparte na zjawisku Halla - sensory wykrywają impulsy na obracającej się tarczy magnetycznej umieszczonej z tyłu silnika. Używane enkodery kwadraturowe posiadają rozdzielcość 48 impulsów na obrót, co po przemnożeniu przez wartość przekładni 9,7:1 - daje wynik prawie 466 impulsów na obrót.



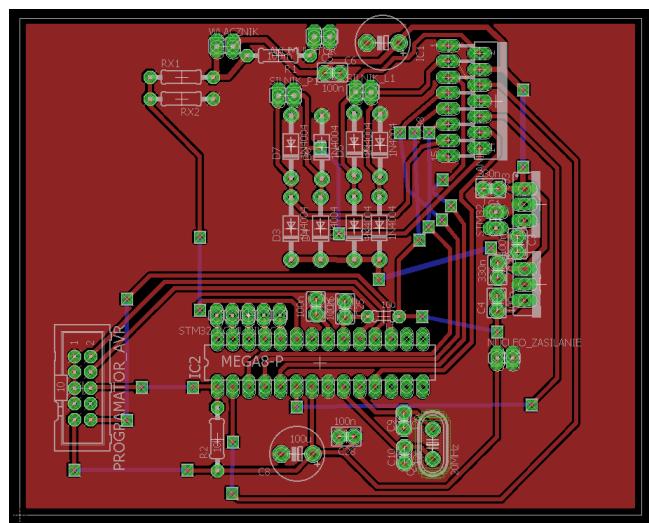
Rysunek 3.16: Enkodery magnetyczne CPR 48 i wyprowadzenia przewodów silnika [opracowanie własne]

Wyprowadzenia silników:

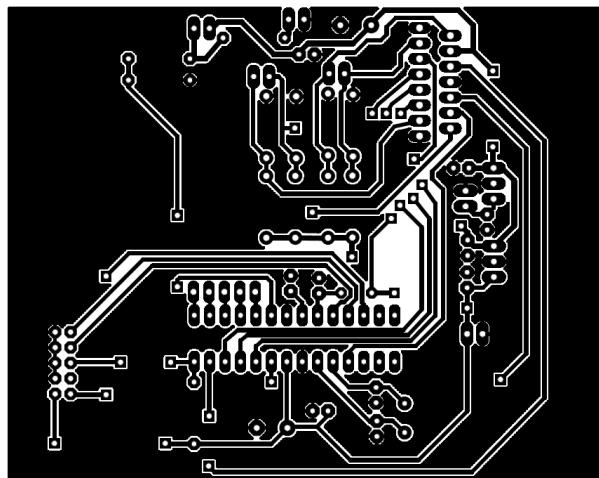
- czerwony - zasilanie silnika 1,
- czarny - zasilanie silnika 2,
- zielony - potencjał masy enkodera,
- niebieski - zasilanie enkodera, tolerancja od 3.5 V do 20 V,
- żółty - wyjście A enkodera,
- biały - wyjście B enkodera,

3.3.4 Wykonanie płytki PCB

Oprócz schematu połączeń elementów elektronicznych w programie EAGLE został również wykonany projekt płytki PCB:



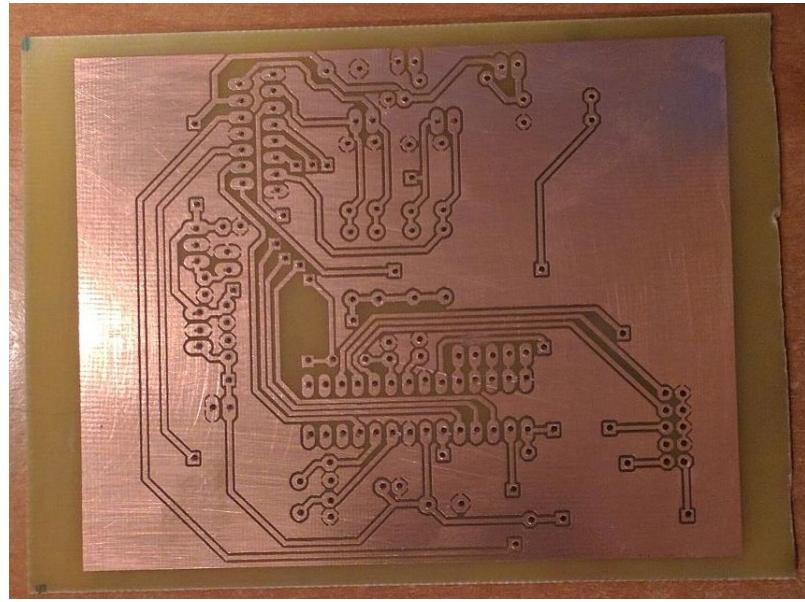
Rysunek 3.17: Projekt płytki PCB sterownika silników prądu stałego wykonyany w programie Cadsoft EAGLE [opracowanie własne]



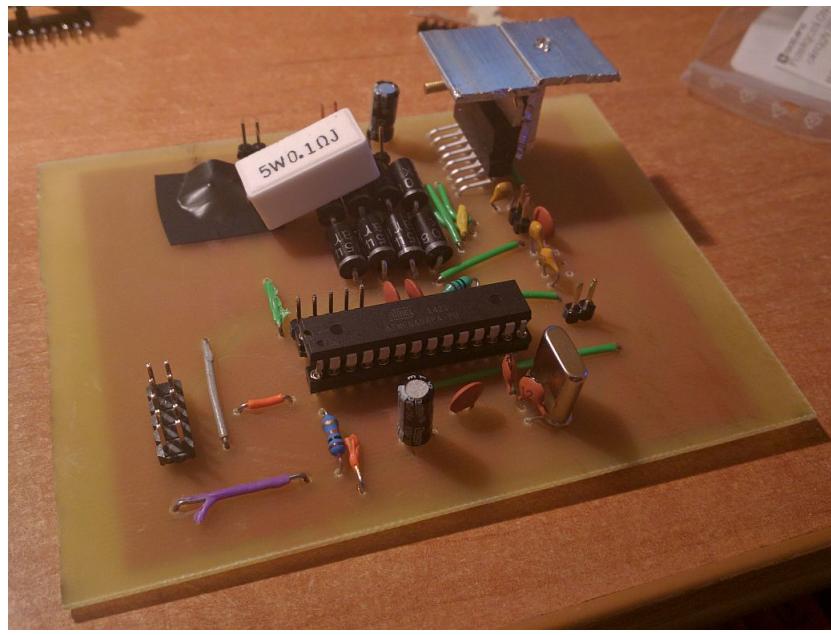
Rysunek 3.18: Obraz połączeń elementów elektronicznych sterownika silników prądu stałego przygotowany do wydrukowania [opracowanie własne]

Projekt płytki PCB został wyeksportowany do pliku PDF, aby uniknąć jakiegokolwiek przeskalowania elementów. Wydruk został przeprowadzony na drukarce laserowej na papierze kredowym. Wykorzystanie tego typu papieru umożliwia przeniesienie toneru na laminat pokryty miedzią za pomocą techniki termotransferu. Wydruk i papier zostały wyprasowane w temperaturze około 220 °C przez około 7 minut. Tak wysoka temperatura była konieczna ze względu na użytą płytę pokrytą dwustronnie miedzią - większa ilość metalu pochłania więcej ciepła. Następnie wyprasowane elementy zostały zanurzone na 15 minut w wodzie, a po rozmiękczeniu papieru został on delikatnie usunięty z płytki. Po powyższych czynnościach został usunięty osiadły kamień za pomocą zwykłego noża, a także zostały poprawione wszelkie niedoskonałości powstałe podczas termotransferu. W pełni przygotowana płytka została poddana trawieniu w roztworze nadsiarczanu sodu przez ponad 30 minut, pozwoliło to na usunięcie miedzi z miejsc niepokrytych tonerem. Na koniec zostały wywiercone otwory do montażu elementów elektronicznych przewlekanych przy użyciu wiertła o średnicy 1 mm.

Po ukończeniu płytki, przewody i uprzednio przetestowane elementy elektroniczne zostały przylutowane. Po skończonym montażu elementów, wszystkie ścieżki zostały sprawdzone za pomocą multimetru, czy ciągłość obwodu jest zachowana tam, gdzie powinna, a także czy nie zachodzą zwarcia. Następnie został przylutowany akumulator i nastąpiła konieczność sprawdzenia wszystkich poziomów napięć na płytce. Po upewnieniu się, że wszystko jest wykonane prawidłowo w podstawce precyzyjnej został umieszczony mikrokontroler, a sama płytka przymocowana do obudowy robota.



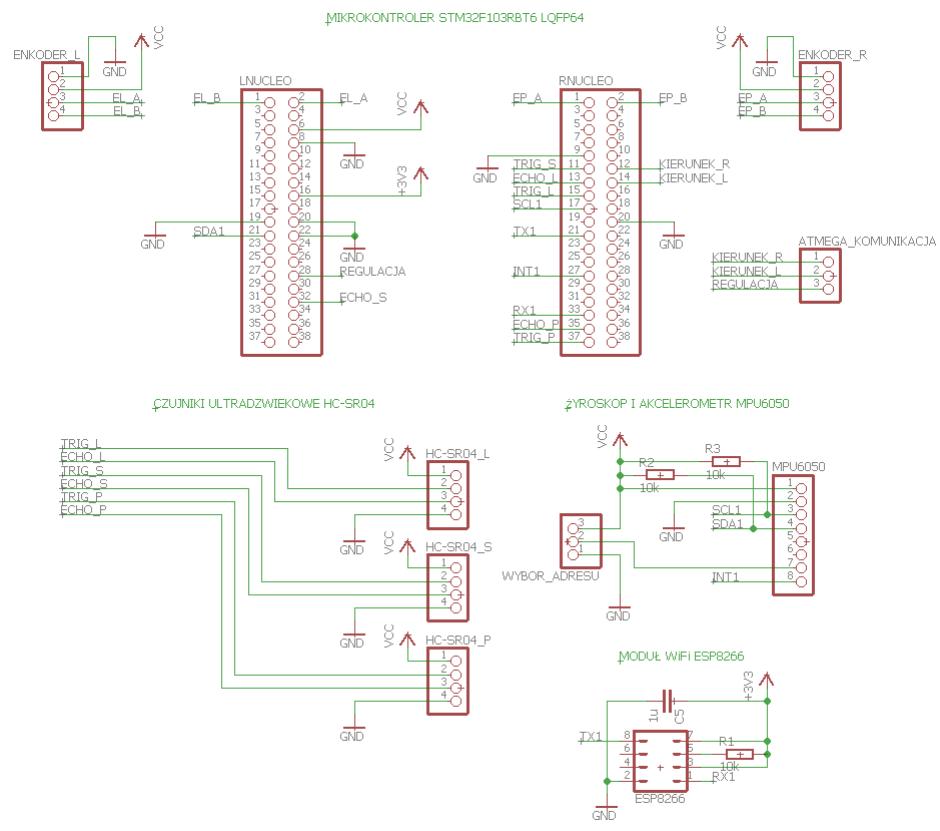
Rysunek 3.19: Płytki PCB sterownika silników wytrawiona w roztworze nad-siarczanu sodu [opracowanie własne]



Rysunek 3.20: Gotowa płytka sterownika silników [opracowanie własne]

3.4 Układ regulacji położenia i skanowania pomieszczeń

Do wykonywania większości obliczeń został wybrany mikrokontroler STM32F103RBTF na płytce NUCLEO. Układ scalony zbiera informacje ze wszystkich czujników robota: enkoderów magnetycznych, akcelerometru, żyroskopu, czujników ultradźwiękowych i modułu WiFi. Następnie wykorzystuje te dane do obliczenia aktualnej prędkości i kierunku obrotów silników, aby utrzymać pion i zeskanować całe pomieszczenie. Przeprogramowywanie mikrokontrolera jest ułatwione poprzez wbudowany programator na płytce NUCLEO.



Rysunek 3.21: Sposób połączenia elementów elektronicznych układu regulacji położenia i skanowania pomieszczeń wykonany w programie Cadsoft EAGLE [opracowanie własne]

3.4.1 Mikrokontroler STM32F103RBT6

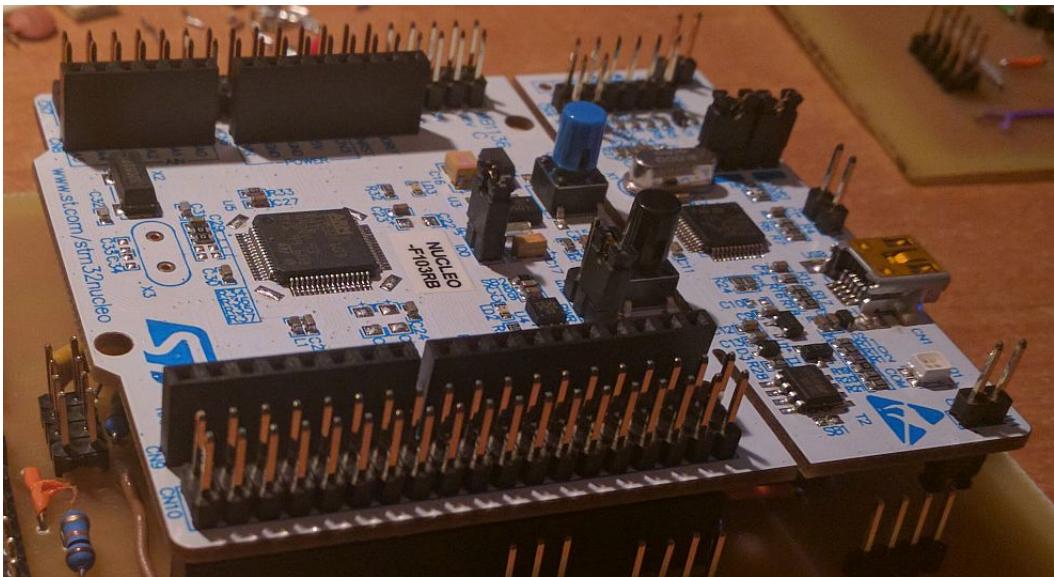
Mikrokontroler STM32F103RBT6 32-bitowej architektury ARM na płytce NUCLEO z wyprowadzeniami pinów układu scalonego na złącza goldpin i zintegrowany z programatorem został wybrany ze względu na łatwość montażu i programowania. Architektura używanego mikrokontrolera jest oprarta na rdzeniu Cortex M3, co oznacza dość dużą moc obliczeniową przy rozsądnej cenie.

Dane techniczne NUCLEO-F103RB:

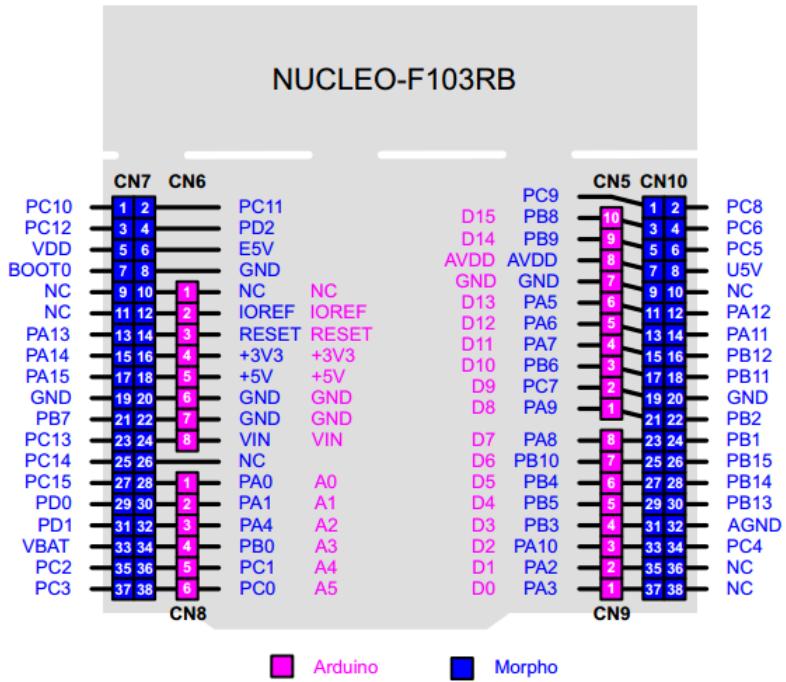
- częstotliwość taktowania: 72 MHz,
- pamięć programu Flash: 128 kB,
- pamięć SRAM: 20 kB,
- 2 przetworniki analogowo-cyfrowe: 12-bitowe, 16-kanałowe,
- 7 timerów,
- interfejsy: 3 USART, 2 SPI 18 Mbit/s, 2 I2C, USB Full Speed, CAN 2,0B,
- debugger ST-Link/V2 umieszczony na płytce,



Rysunek 3.22: Mikrokontroler STM32F103RBT6 w obudowie LQFP64 [opracowanie własne]



Rysunek 3.23: Płytkę NUCLEO-F103RB [opracowanie własne]



Rysunek 3.24: Wyprowadzenia NUCLEO-F103RB [źródło: bibliografia pkt. 4, strona 30]

Mikrokontroler jest programowany przez złącze Mini USB znajdujące się na płytce. Układ scalony STM32F103RBT6 jest wykorzystywany do zbierania sygnałów ze wszystkich czujników:

- enkodery magnetyczne - sprzężenie zwrotne w postaci informacji o położeniu kół do układu regulacji położenia w pionie,
- akcelerometr i żyroskop - sprzężenie zwrotne w postaci informacji o przyspieszeniu kątowym i kącie pochylenia robota do układu regulacji położenia w pionie,
- czujniki ultradźwiękowe - informacje o położeniu przeszkód na drodze - do układu skanowania pomieszczeń,
- moduł WiFi - odbieranie informacji o błędach z komputera PC.

Zadaniem mikrokontrolera jest przede wszystkim wykorzystać informacje o przebytej drodze i pochyleniu robota do utrzymania pozycji pionowej za pomocą regulatora liniowo-kwadratowego. Sygnały z czujników jako zmienne stanu dodatkowo przechodzą przez filtr Kalmana, co zwiększa ich dokładność. Drugim zadaniem układu scalonego jest zebranie informacji o aktualnej pozycji przeszkód znajdujących się przed robotem i wysłanie ich przez moduł WiFi do sieci lokalnej (co może być odebrane przez dowolne urządzenie obsługujące protokół HTTP). Po każdorazowym wykonaniu tych czynności obliczone wartości prędkości i kierunku obrotu kół są wysyłane do sterownika silników.

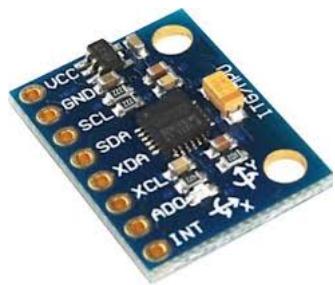
Połączenie płytki NUCLEO z resztą układu nie wymagało żadnych dodatkowych elementów pasywnych.

3.4.2 Żyroskop i akcelerometr - układ MPU6050

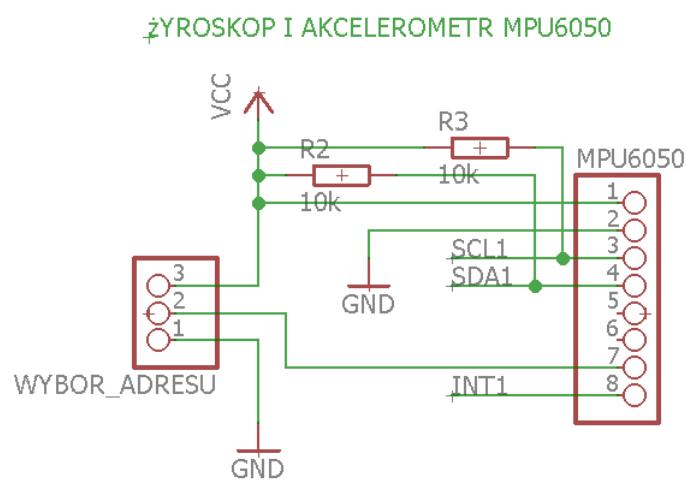
Użyty moduł firmy Invensense zawiera wbudowany cyfrowy żyroskop i akcelerometr. Jego zastosowanie wynika z niskiej ceny i dostatecznych parametrów pracy:

- napięcie zasilania: 3,3V - 5V,
- zakresy pracy żyroskopu: 250 dps, 500 dps, 1000 dps, 2500 dps (dps - degrees per second, stopni/sekundę),
- zakresy pracy akcelerometru: 2g, 4g, 8g, 16g,
- wymiary: 20mm x 16mm,
- waga: 0,9 g.

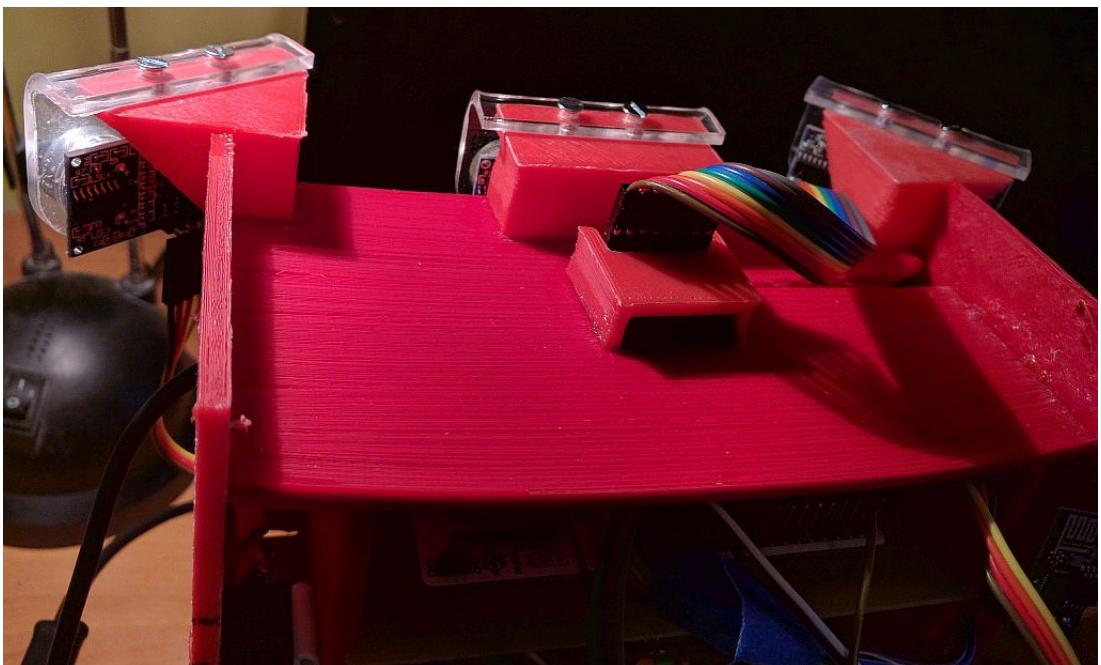
Komunikacja z modułem odbywa się za pośrednictwem magistrali szeregowej I2C. Ścieżki SDA i SCL zostały podciągnięte do napięcia zasilania, aby uniknąć stanów nieokreślonych na pinach.



Rysunek 3.25: Żyroskop i akcelerometr - układ MPU6050 [źródło: https://hobbytronics.com.pk/wp-content/uploads/MPU6050_1.png]



Rysunek 3.26: Sposób połączenia czujnika MPU6050 wykonany w programie EAGLE [opracowanie własne]



Rysunek 3.27: Umiejscowienie czujnika MPU6050 na obudowie robota [opracowanie własne]

3.4.3 Czujniki ultradźwiękowe HC-SR04

Użyte czujniki HC-SR04 zostały zakupione razem z uchwytymi montażowymi. Ich zastosowanie wynika z niskiej ceny i dostatecznych parametrów pracy:

- napięcie zasilania: 5 V,
- średni pobór prądu: 15 mA,
- zakres pomiarowy: od 2 cm do 200 cm,
- częstotliwość pracy: 40 kHz,
- wymiary: 45 x 20 x 15 mm.



Rysunek 3.28: Czujniki ultradźwiękowe HC-SR04 [opracowanie własne]

Komunikacja z czujnikami odbywa się za pośrednictwem dwóch pinów: ECHO i TRIG. Wynika to z działania układów. Po podaniu stanu wysokiego na pin TRIG w postaci impulsu trwającego 10 μ s moduł wykonuje pomiar odległości.

Aby rozpocząć pomiar należy podać na pin TRIG impuls napięciowy (stan wysoki 5V) przez 10 μ s. Moduł dokonuje pomiaru odległości przy pomocy fali dźwiękowej o częstotliwości 40 kHz. Na pinie ECHO otrzymywany jest sygnał, w którym odległość od przeszkody jest zależna od czasu trwania stanu wysokiego. Odległość w milimetrach od przeszkody wynosi:

$$d = 0,17t_{ECHO},$$

gdzie:

d - odległość mierzona,

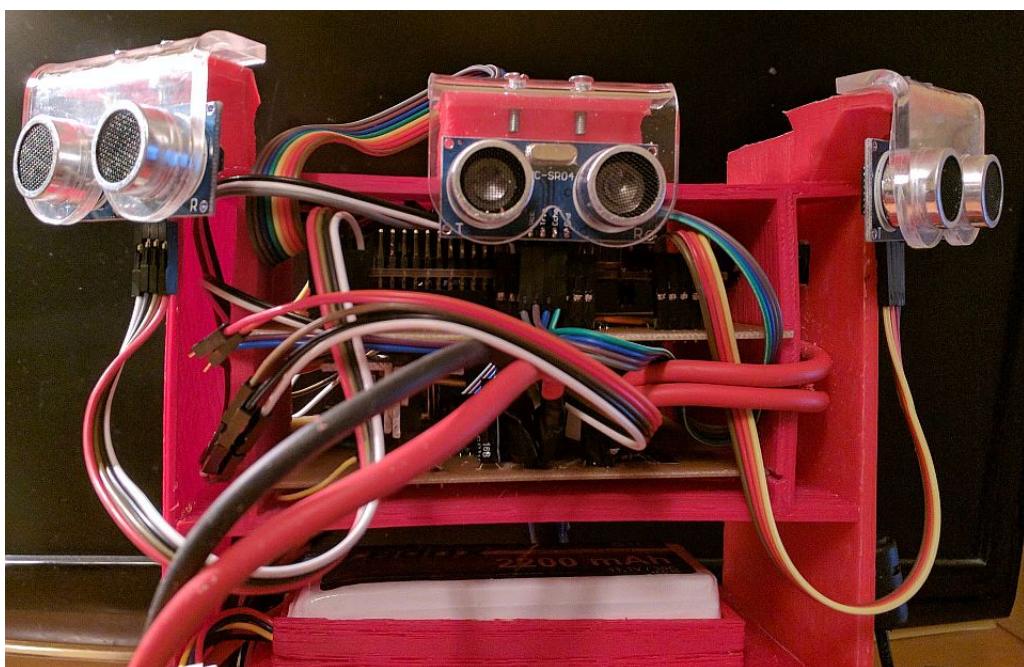
t_{echo} - czas trwania stanu wysokiego na pinie ECHO.

Wzór jest wyprowadzony z prostej zależności:

$$d = (t_{echo} V_{sound})/2,$$

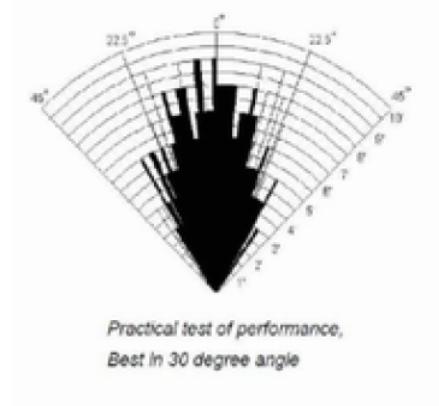
gdzie:

V_{sound} - prędkość rozchodzenia się dźwięku w powietrzu - 340 m/s.

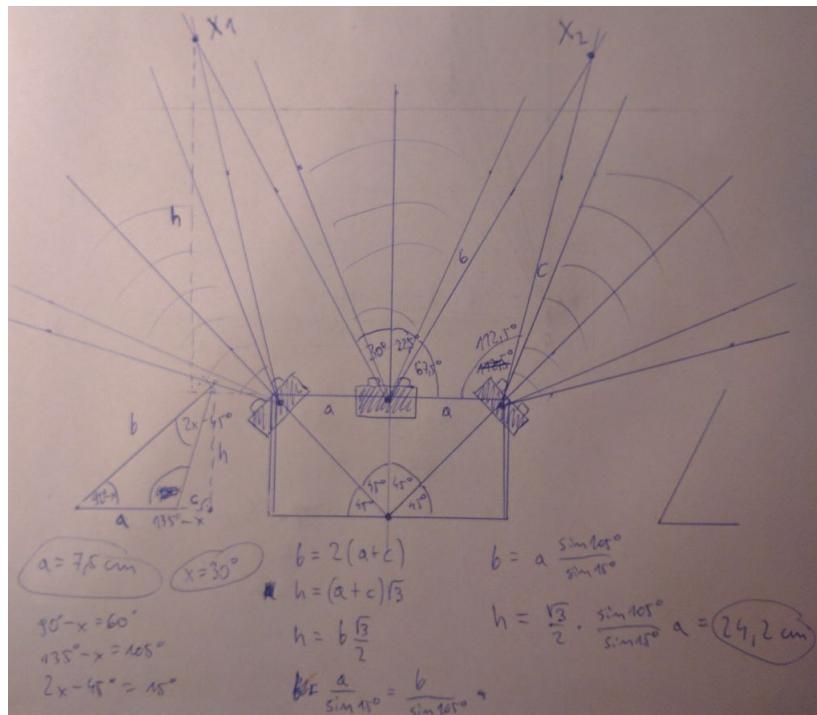


Rysunek 3.29: Umieszczenie czujników HC-SR04 na obudowie robota [opracowanie własne]

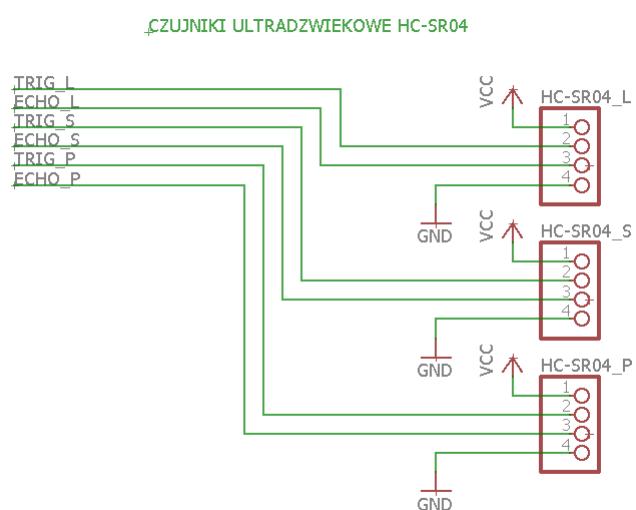
W zestawie znajdują się specjalne uchwyty montażowe, do których zostały zaprojektowane mocowania na szczycie konstrukcji robota. Model w 3D został zaprojektowany w ten sposób, aby czujniki ultradźwiękowe były ustawione: idealnie na wprost jazdy, pod kątem 45° w prawo i 45° w lewo. Czujnik najlepiej działa w polu do 30° od osi prostopadłej do modułu. Stąd od odległości ok. 24 cm od konstrukcji robot będzie miał widoczność w pełnym zakresie 180° .



Rysunek 3.30: Zasięg czujnika HC-SR04 [źródło: bibliografia pkt. 6, strona 4]



Rysunek 3.31: Obliczenie obszaru widoczności robota [opracowanie własne]



4

Rysunek 3.32: Sposób połączenia czujników HC-SR04 wykonany w programie EAGLE [opracowanie własne]

3.4.4 Moduł WiFi ESP-01 8266

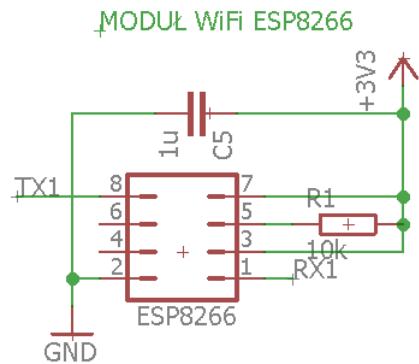
Moduł został wybrany ze względu na niską cenę, powszechnie użycie i dużą moc obliczeniową. Dane techniczne:

- zasilanie: 3,3 V,
- pamięć Flash: 1 MB,
- 2 GPIO - wyjścia/wejścia cyfrowe,
- 1 UART,
- wbudowana antena PCB,
- wymiary: 24,8 x 16 mm.

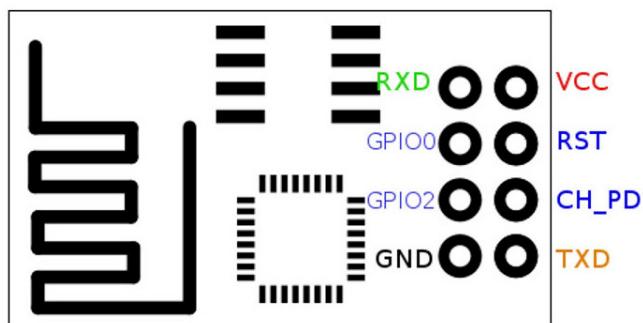


Rysunek 3.33: Moduł WiFi ESP-01 8266 [opracowanie własne]

Komunikacja z modułem odbywa się za pośrednictwem interfejsu UART - piny RX i TX. Do prawidłowego działania układu został dołączony kondensator ceramiczny $1\mu F$ między napięcie zasilania a masę. Piny Reset i CH_PD zostały podciagnięte do napięcia zasilania modułu, aby dodatkowo ustabilizować pracę urządzenia.



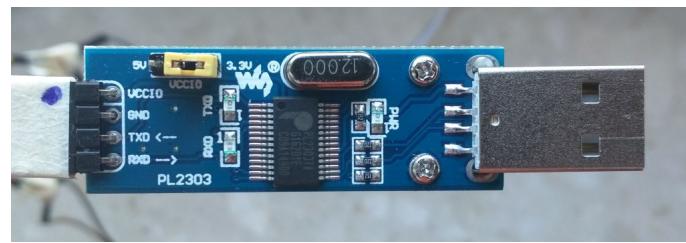
Rysunek 3.34: Sposób połączenia modułu ESP8266 wykonany w programie EAGLE [opracowanie własne]



Rysunek 3.35: Wyprowadzenia ESP-01 8266 [opracowanie własne]

Do zaprogramowania modułu należy użyć pinów UART i zewrzeć pino GPIO0 do masy. Z powodu napięcia zasilania ESP8266 3,3 V konieczne jest zastosowanie konwertera napięć, aby podłączyć moduł przez interfejs USB do komputera PC. Dla podwyższonej wygody w tym wypadku został użyty konwerter UART-USB oparty na układzie PL2303, który może pracować na dwóch napięciach 3,3 V i 5 V (wybór napięcia za pomocą zworki).

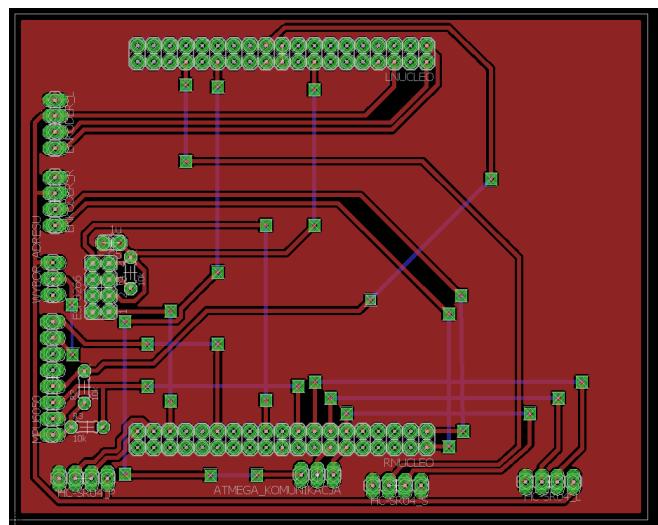
W ESP8266 został wgrany firmware NodeMCU w wersji: nodemcu_float_0.9.6-dev_20150704 za pomocą programu ESP8266Flasher. Nowy firmware umożliwia programowanie modułu za pomocą języka skryptowego Lua, a także w środowisku Arduino, z wykorzystaniem biblioteki esp8266 by ESP8266 Community w wersji 2.3.0-rc2



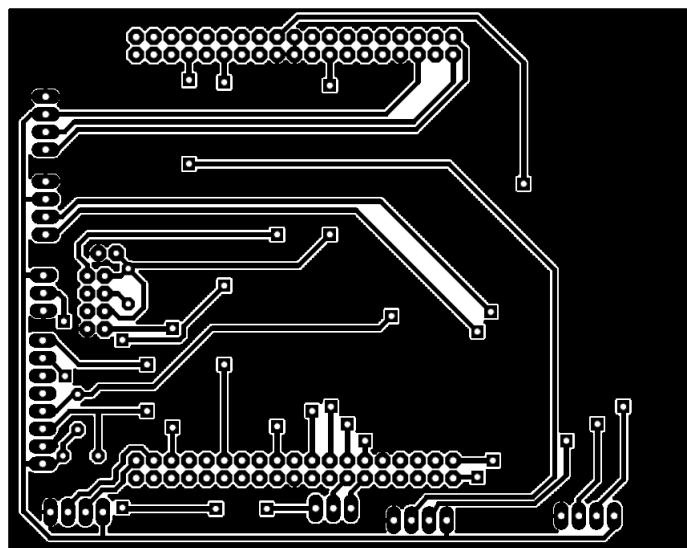
Rysunek 3.36: Konwerter UART-USB oparty na układzie PL2303 [opracowanie własne]

3.4.5 Wykonanie płytki PCB

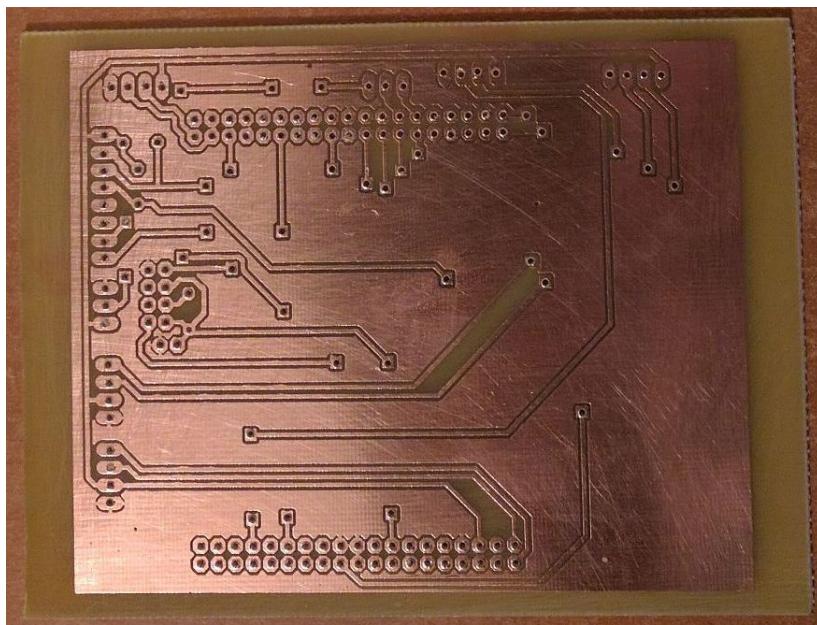
Płytkę PCB układu regulacji położenia i skanowania pomieszczeń została wykonana w analogiczny sposób jak płytka sterownika silników prądu stałego. Zamiast podstawki precyzyjnej pod mikrokontroler zostały zamontowane żeńskie gniazda goldpin, w których można umieścić płytke NUCLEO-F103RB.



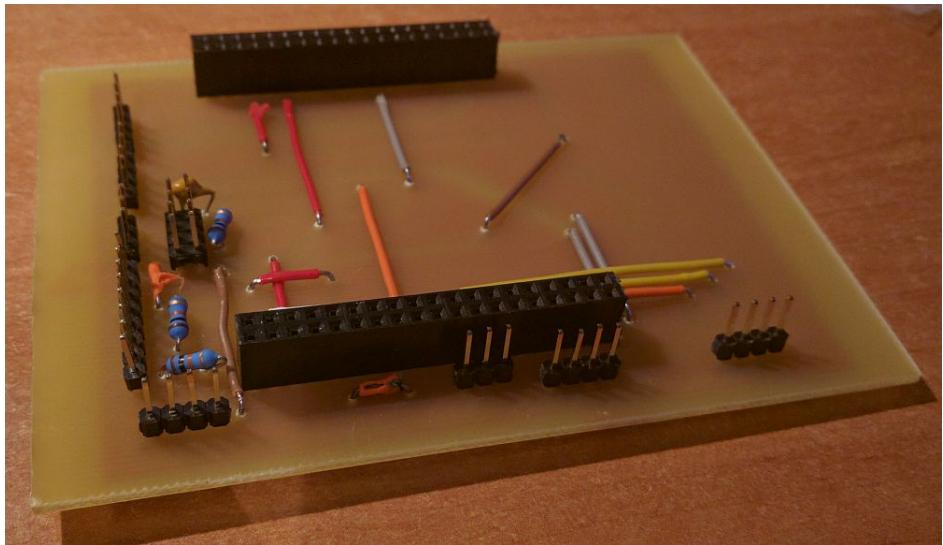
Rysunek 3.37: Projekt płytki PCB układu regulacji położenia i skanowania pomieszczeń wykonany w programie Cadsoft EAGLE [opracowanie własne]



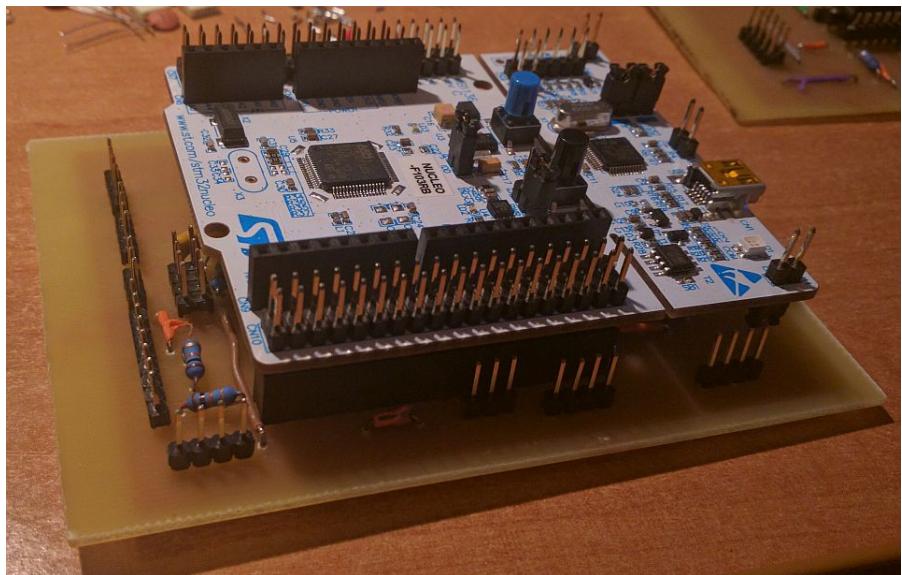
Rysunek 3.38: Obraz połączeń elementów elektronicznych układu regulacji położenia i skanowania pomieszczeń przygotowany do wydrukowania [opracowanie własne]



Rysunek 3.39: Płytki PCB układu regulacji położenia i skanowania pomieszczeń wytrawiona w roztworze nadsiarczanu sodu [opracowanie własne]



Rysunek 3.40: Gotowa płytka układu regulacji położenia i skanowania pojemności bez zamontowanej płytce NUCLEO-F103RB [opracowanie własne]



Rysunek 3.41: Gotowa płytka układu regulacji położenia i skanowania pojemności z zamontowaną płytą NUCLEO-F103RB [opracowanie własne]

Rozdział 4

Teoria Sterowania

Wykorzystując obliczony model matematyczny układu został zaprojektowany układ sterowania robota w programach Matlab i Simulink. Dobrze zamodelowany układ pozwoli znacznie szybciej dobrać prawidłowe wartości macierzy Q i R w regulatorze liniowo-kwadratowym LQR.

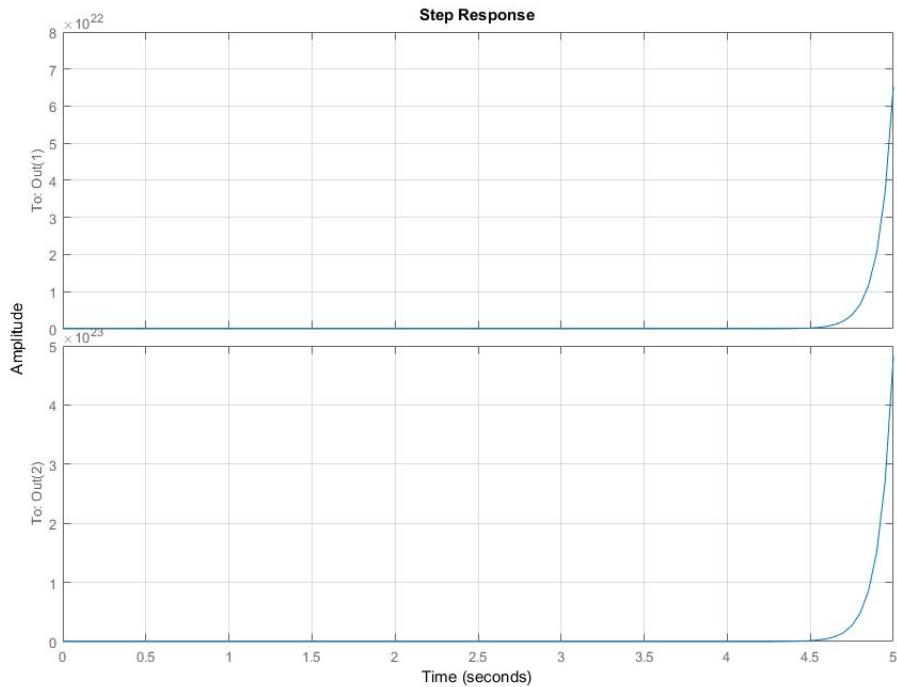
4.1 Wyznaczenie parametrów układu

Używając gotowych funkcji w Matlabie obliczyłem rzędy macierzy obserwowalności i sterowalności - wynoszą 4. Oznacza to, że skonstruowany układ jest sterowalny i obserwowałny.

Układ bez pętli sterowania jest niestabilny - posiada dodatni biegun:

$$\begin{bmatrix} 0 \\ -11.6894 \\ -0.1000 \\ 11.5017 \end{bmatrix}$$

Niestabilność układu można zilustrować zasymulowaną odpowiedzią skokową, po zadaniu impulsu dający do nieskończoności.

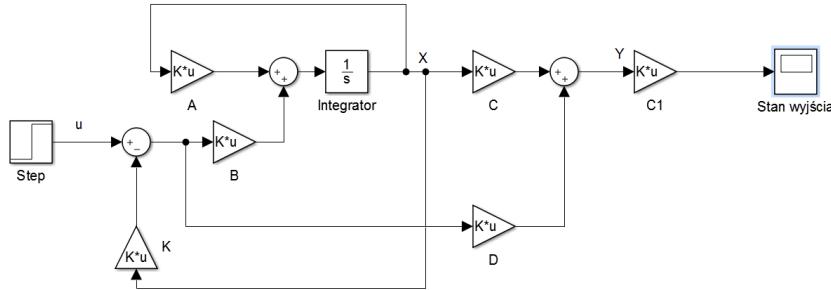


Rysunek 4.1: Odpowiedź skokowa układu otwartego [opracowanie własne]

4.2 Projekt regulatora liniowo-kwadratowego (LQR)

Zastosowanie regulatora LQR zapewnia optymalne sterowanie, poprzez zapewnienie dopasowanego wzmacnienia w pętli sprzężenia zwrotnego, ale wymaga dokładnie wyliczonego modelu układu. Założenie projektowe to czas ustalenia odpowiedzi poniżej 1 sekundy.

Regulator został zamodelowany w programie Simulink:



Rysunek 4.2: LQR w programie Simulink [opracowanie własne]

Macierz wzmocnień K sprzężenia zwrotnego jest obliczana rozwiązując równanie różniczkowe Riccatiego o postaci:

$$K = R^{-1}B^TP(t)$$

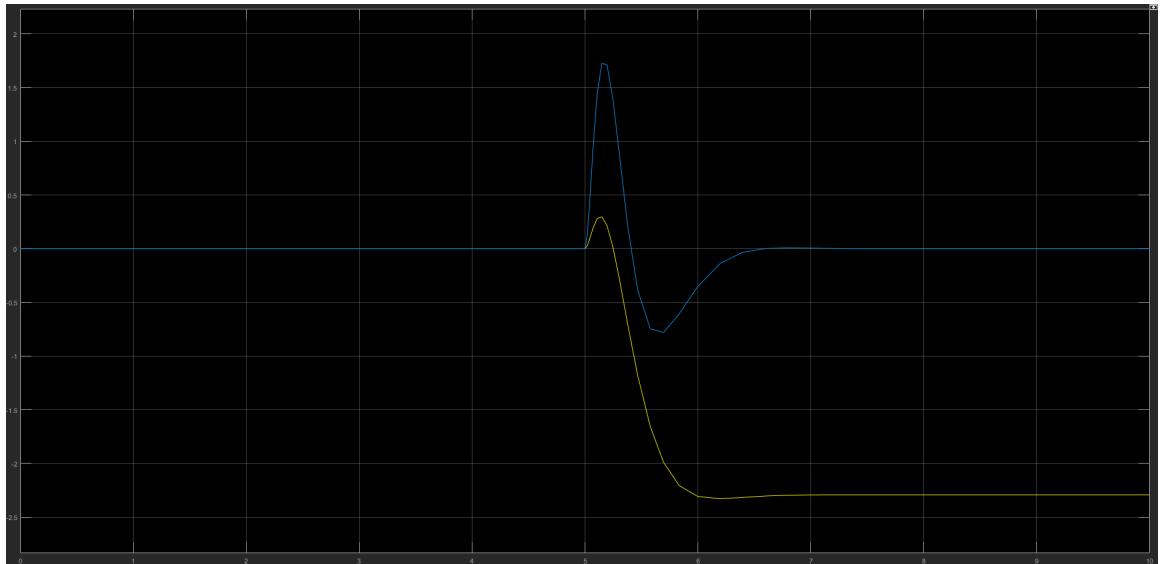
$$\dot{P}(t) = A^TP(t) + P(t)A - P(t)BR^{-1}B^TP(t) + Q$$

Gdzie Q i R to macierze wagowe wskaźnika J . Wartości macierzy Q i R zostały dobrane tak, aby spełnić założenie projektowe:

- Przesunięcie robota w przybliżeniu nie większe niż 8 cm:
 $Q_{11} = 1/0.08^2$,
- Prędkość robota nie większa niż 3 m/s: $Q_{22} = 1/3^2$,
- Mały kąt wychylenia wahadła nie większy niż 15°: $Q_{33} = 1/15^2$,
- Prędkość kątowa wychylenia wahadła nie większa niż 30°/s:
 $Q_{44} = 1/30^2$.

Wartości pozostałych elementów macierzy Q przyjąłem jako zerowe – używając w ten sposób macierz symetryczną. Podobnie dla macierzy R , która ma wymiary 1×1 , jest to kwadrat odwrotności największej siły zadziałającej na wahadło czyli ok. 2 N, więc $R = [1/2^2]$.

Odpowiedź układu na skok jednostkowy:



Rysunek 4.3: Odpowiedź skokowa układu z LQR w programie Simulink [opracowanie własne]

Układ jest stabilny i spełnia założenia projektowe.

Obliczone Macierze Q i R:

$$Q = \begin{bmatrix} 156.2500 & 0 & 0 & 0 \\ 0 & 0.1111 & 0 & 0 \\ 0 & 0 & 0.0044 & 0 \\ 0 & 0 & 0 & 0.0011 \end{bmatrix}, R = [0.2500]$$

Obliczone wzmacnienie LQR:

$$K = [-25.0000 \quad -12.6331 \quad 40.5490 \quad 4.9073]$$

Symulacja udowadnia, że teoretycznie jest możliwe ustabilizowanie takiego obiektu w pionie.

Rozdział 5

Oprogramowanie

Robot został podzielony na kilka modułów, stąd konieczność napisania wielu programów. Każdą część można traktować jako osobny element, co umożliwia wykonywanie testów pojedynczych modułów, bez konieczności podłączania ich do reszty podzespołów. Takie podejście ułatwia wykrycie błędu w całym systemie i zapewnia lepszą diagnostykę.

Łączna objętość kodu to 2538 linii.

5.1 Projekt sterowania położenia w pionie (Matlab, Simulink)

Projekt sterowania został napisany w Matlabie i Simulinku, ponieważ umożliwia to łatwe zaprojektowanie systemu za pomocą bloków oraz testowanie różnych wariantów i parametrów sterowania.

Cały program został zapisany w jednym pliku (62 linie). Składowe programu lqr.m:

- wyliczenia macierzy układu przedstawionych w rozdziale 2.7,
- obliczenia sterowalności, obserwonalności, biegunów i zasymulowania odpowiedzi skokowej układu, przedstawionych w rozdziale 4.1,
- obliczenia wzmacnienia LQR i zasymulowania działania na zamodelowanym układzie robota (rozdział 4.2).

Do obliczenia sterowalności, obserwonalności, biegunów, wzmacnienia LQR i zasymulowania odpowiedzi skokowej zostały wykorzystane funkcje z biblioteki Matlaba, odpowiednio:

- ctrb() - sterowalność, zwraca rząd macierzy sterowalności,

- obser() - obserwonalność, zwraca rząd macierzy obserwonalności,
- pole() - oblicza bieguny układu,
- lqr() - oblicza wzmacnienie LQR,
- step() - symulacja odpowiedzi skokowej układu.

5.2 Sterownik silników prądu stałego (C, AVR)

Program został napisany w środowisku Eclipse Mars, z pluginem AVR, w języku ANSI C. Zegar mikrokontrolera ATMega168PA-PU został ustawiony na 20 MHz, generowany z zewnętrznego rezonatora kwarcowego.

Użyte biblioteki:

- avr/interrupt.h - biblioteka AVR do obsługi przerwań,
- avr/iom168.h - biblioteka AVR do pinów wejścia/wyjścia dla mikrokontrolera ATmega168,
- util/delay.h - dla funkcji _delay_ms(), która pozwala wprowadzić opóźnienia do programu.

Cały program został zapisany w jednym pliku - main.c (157 linii). Składowe programu:

1. Dyrektywy preprocesora dla użytych pinów i pomocniczych makr.
Wpisanie wartości sprzętowych, takich jak, użyte piny w dyrektywach preprocesora umożliwia łatwe przeprowadzenie zmian w programie, w przypadku zmiany któregoś pinu i sprawia, że program staje się bardziej uniwersalny.
2. Deklaracje zmiennych globalnych, słowo kluczowe volatile zapobiega optymalizacji zmiennych przez kompilator, co jest konieczne, w przypadku, gdy zmienne są używane w przerwaniach.
3. Funkcja uint16_t measure(uint8_t channel) umożliwiająca pomiar napięcia dla dowolnego pinu ADC mikrokontrolera (zakres od 0 do 5), za pomocą działań na bitach.
4. Funkcja void hardware_setup() ustawiająca porty jako wejście/wyjście, konfigurująca ADC, TIMER2, i wyjścia na mostek L298N oraz uruchamia globalną flagę przerwania.
Konfiguracja ADC: ADEN - włączenie ADC, ADPSX - prescaler (w tym przypadku ustawiony na 8), REFS0 - ustawienie napięcia odniesienia jako AVCC z dołączonym kondensatorem.

Konfiguracja TIMER2: WGM21 - ustawienie trybu CTC, CS21 - prescaler 8, OCR2A - dodatkowy podział częstotliwości przez 200, OCIE2A - ustawienie porównania do rejestru OCR2A
sei() - umożliwia używanie przerwań w programie.

5. Funkcja void gather_data(), która odbiera dane od mikrokontrolera STM32F103RBT6,
6. Funkcja void set_direction() - ustala kierunek obrotu silników na podstawie danych odebranych funkcją gather_data(),
7. Procedura obsługi przerwania ISR(TIMER2_COMPA_vect) do obsługi programowego PWM,
8. Główna funkcja programu - int main(void), która zajmuje się ustawianiem odpowiedniego kierunku obrotów silnika i utrzymywaniem prędkości silników za pomocą sygnału PWM. Konieczne było zastosowanie opóźnienia 2 ms, aby umożliwić stabilną pracę mikrokontrolera.

5.3 Obsługa czujników i sterowanie położenia w pionie (C, StdPeriph)

Główny program sterujący został zaimplementowany w środowisku System Workbench for STM32 opartym na programie Eclipse. Kod został napisany w języku ANSI C.

Struktura plików programu:

- stm32f1xx_it.h - plik wygenerowany przez środowisko System Workbench do komplikacji projektu (61 linii),
- syscalls.c - plik wygenerowany przez środowisko System Workbench do minimalnej obsługi zwołań (calls), 204 linie kodu,
- system_stm32f10x.c - plik wygenerowany przez środowisko System Workbench, główny kod konfiguracyjny dla mikrokontrolera (częstotliwość taktowania, początkowe ustawienie głównych rejestrów), 379 linii,
- HAL_MPU6050.h - plik nagłówkowy biblioteki do obsługi I2C (MIT license Copyright (c) 2012 Harinadha Reddy Chintalapalli) zawierający konfigurację sprzętową - wybór pinów i kanału I2C mikrokontrolera (65 linii),
- MPU6050.h - plik nagłówkowy biblioteki do obsługi I2C i modułu akcelerometra i żyroskopu MPU6050 (MIT license Copyright (c) 2012 Harinadha Reddy Chintalapalli), zawiera definicję wszystkich rejestrów MPU6050 (428 linii),

- MPU6050.c - zawiera kod źródłowy dla biblioteki obsługującej I2C i moduł MPU6050 (MIT license Copyright (c) 2012 Harinadha Reddy Chintalapalli), obsługuje całą komunikację z MPU6050 i udostępnia funkcję void MPU6050_GetRawAccelGyro(s16* AccelGyro), która pobiera dane akcelerometru i żyroskopu (465 linii),
- esp8266.h - plik nagłówkowy do obsługi modułu Wifi ESP8266 (21 linii),
- esp8266.c - zawiera kod źródłowy części obsługującej moduł WiFi ESP8266, czyli konfigurację UART (53 linie),
- main.c - plik z programem głównym, zawiera obsługę enkoderów, komunikację z ATmega168PA-PU i MPU6050 oraz sterowanie położenia w pionie (408 linie).

Program składa się z 482 linii autorskiego kodu, 644 linii kodu wygenerowanego i 958 linii kodu bibliotecznego, co przekłada się na łączną sumę 2084 linii. Po skompilowaniu program zużywa 35056 bajtów miejsca w pamięci flash mikrokontrolera.

5.3.1 Plik HAL_MPU6050.h

Komunikacja z modułem MPU6050 została skonfigurowana dla kanału pierwszego magistrali I2C mikrokontrolera STM32F103RBT6. Prędkość transmisji danych to 100 kHz.

5.3.2 Plik esp8266.c

Komunikacja z modułem WiFi ESP8266 została skonfigurowana dla kanału trzeciego USART, baud rate 115200, 8 bitów danych, brak kontroli parzystości i 1 bit stopu. Transmisja jest inicjalizowana przez wysłanie pakietu zawierającego same zera.

Zaimplementowanie funkcji int __io_putchar(int c) pozwala na używanie funkcji formatowanego wydruku danych printf() dla transmisji USART.

5.3.3 Plik main.c

Plik główny programu zbiera dane ze wszystkich czujników co wiąże się z użyciem kilku bibliotek:

- math.h - biblioteka matematyczna, wykorzystywana do obliczania dwuargumentowego arcusa tangensa,

- stdbool.h - umożliwia używanie w języku C logicznego typu danych bool,
- stm32f10x.h - biblioteka środowiska System Workbench for STM32 zawierająca definicje rejestrów i obsługę peryferiów mikrokontrolera STM32F103RBT6,
- MPU6050.h - biblioteka do obsługi modułu żyroskopu i akcelerometra MPU6050,
- esp8266.h - biblioteka do modułu WiFi opisana w części 5.3.2.

Kod składa się z kilku części:

1. Deklaracja stałych mechanicznych, matematycznych i elektronicznych:
 - ROTATION_CONSTANT wyliczone doświadczalnie na podstawie pomiarów odległości z enkoderów, tak aby dać jak najdokładniejszy wynik,
 - WHEEL_DIAMETER - średnica kół w centymetrach,
 - PI - liczba PI z dokładnością do 6 miejsc po przecinku,
 - MPU_CONSTANT - stała, przez którą należy podzielić wyniki z MPU6050, aby otrzymać jednostki w układzie SI,
 - ACCELEROMETER_SENSITIVITY - czułość akcelerometru,
 - GYROSCOPE_SENSITIVITY - czułość żyroskopu,
 - dt - czas pomiędzy odczytami pomiarów z MPU6050.
2. Deklaracje struktur, zmiennych i flag globalnych.
3. Funkcja void delay_ms(int time) - wprowadza opóźnienia do programu.
4. Funkcja void SysTick_Handler() - obsługa przerwania od zegara, częstotliwość przerwania zdefiniowana w funkcji atmega_communication_init() przy wywołaniu funkcji SysTick_Config(). W programie funkcja SysTick_Handler() wykonuje się co 1 ms. Wystawia flagi do obliczeń w pętli głównej programu.
5. Funkcja void global_variables_init() - inicjalizuje wszystkie zmienne globalne.
6. Funkcja void set_pwm(TIM_OCInitTypeDef channel, int pwm_nr, int duty_cycle) - ustawia wypełnienie sprzętowego PWM dla konkretnego pinu,
7. Funkcja float check_velocity(int encoder_counter) do dokładnego pomiaru prędkości chwilowej,

8. Funkcja void encoder_init() - inicjalizuje porty, zegary i przerwania reagujące na zbocza narastające do obsługi enkoderów.
9. Funkcja void atmega_communication_init() - uruchamia przerwanie obsługiwane przez funkcję SysTick_Handler(), inicjalizuje porty, zegary, timery (tryb sprzętowej modulacji szerokości impulsu - PWM) do komunikacji z mikrokontrolerem ATmega168.
10. Funkcja void mpu_read_data() - odczytuje surowe dane z modułu MPU6050 i zapisuje je do globalnej struktury measuredData.
11. Funkcje void EXTI9_5_IRQHandler() i void EXTI15_10_IRQHandler() to procedury obsługi przerwania dla enkoderów, wystawiają flagi do obliczeń w pętli głównej programu.
12. Funkcja void hardware_setup() - uruchamia wszystkie funkcje konfiguracyjne.
13. Funkcja void export_data_wifi() - wysyła dane ze struktury measuredData przez USART do modułu WiFi ESP8266.
14. Funkcja void export_data_atmega() - wysyła dane do sterownika silników (mikrokontroler ATmega168) przy pomocy 5 linii, na których jest wysyłany sygnał prostokątny ze zmiennym wypełnieniem. Sygnały są odbierane za pomocą przetwornika analogowo-cyfrowego.
15. Funkcja void ComplementaryFilter(float *pitch) - implementacja filtra komplementarnego do filtrowania odczytu kąta pochylenia robota,
16. Funkcja główna int main(void) - zawiera definicje zmiennych lokalnych (głównie do regulatora PID) i pętlę główną programu, w której odbywa się sprawdzanie poszczególnych flag wystawianych w przerwaniach i wykonywanie kodu przeznaczonego do poszczególnych zadań:
 - flagi l_cnt_flag i r_cnt_flag - obliczenie pozycji kół i kierunku poruszania się,
 - flaga velocity_flag - obliczenie drogi i prędkości średniej robota,
 - flaga mpu_flag - odczyt danych z MPU6050, obliczenie wyjścia regulatora PID i przekształcenie go na wypełnienie sygnału PWM oraz wysłanie wyników do sterownika silników,
 - flaga uart_flag - wysłanie danych przez USART do modułu ESP8266

W kodzie ze szczególną uwagą zadbane o wydajność:

- zastosowanie sprzętowej modulacji szerokości impulsu (PWM) praktycznie w ogóle nie obciąża mikrokontrolera,

- stosowanie zmiennych całkowitych ograniczając działania na liczbach zmiennoprzecinkowych,
- stosowanie zmiennych typu unsigned, co przyspiesza obliczenia dwukrotnie,
- wystawianie flag w przerwaniach i wykonywanie zadań w pętli głównej programu.

5.4 Obsługa modułu WiFi, serwer WWW (C++, HTML, CSS, JS, XML, Arduino)

5.4.1 Kod C++

Program został napisany w środowisku Arduino 1.8.2, z wykorzystaniem dodatkowego zestawu płytek esp8266 by ESP8266 Community w wersji 2.3.0-rc2, a dokładniej płytka Generic ESP8266 Module. Cały kod został zapisany w jednym pliku - ajax.ino (325 linii) w języku C++. Wykorzystane biblioteki:

- ESP8266WiFi.h - umożliwia połączenie się do konkretnej sieci WiFi,
- ESP8266WebServer.h - umożliwia utworzenie serwera.

Składowe programu:

1. Deklaracja zmiennych i stałych, a także serwera na porcie 80 - co oznacza konieczność obsługi protokołu HTTP przez klienta.
2. Funkcja void parseData(MeasuredData &m, char* buffer), która przetwarza wartości odebrane z mikrokontrolera STM32F103RBT6 - umieszcza odpowiednie dane w odpowiednich miejscach struktury MeasuredData.
3. Funkcja void read_uart() - odczytuje wartości wysłane z mikrokontrolera STM32F103RBT6. Format wysyłanych danych: "x0,x1,x2,x3,x4,x5,x6,x7,x8,x9,x10,;" , w przypadku napotkania średnika odczytywanie się zakończy. Następnie jest wywoływana funkcja parseData().
4. Funkcja void buildWebsite() - buduje stronę za pomocą kodu HTML i CSS.
5. Funkcja void buildJavascript() - buduje stronę za pomocą kodu JavaScript.

6. Funkcja void buildXML() - zbiera dane odczytane z interfejsu UART oraz sprawdza czas wykonywania programu i napięcie zasilania modułu za pomocą ADC - w Arduino jest to gotowa funkcja ESP.getVcc(). Zebrane pomiary są zapisywane do dokumentu XML.
7. Funkcja String millis2time() - oblicza czas wykonywania programu za pomocą gotowej funkcji w Arduino - millis() oraz zamienia go na godziny, minuty i sekundy.
8. Funkcje void handleWebsite() i void handleXML() wysyłają dokumenty HTML i XML na server.
9. Funkcja void setup() konfiguruje UART (baud rate 115200) do komunikacji z mikrokontrolerem STM32F103RBT6, podejmuje próbę połączenia z określona siecią Wifi i wyświetla komunikaty. Najważniejszym komunikatem jest wyświetlenie WiFi.localIP(), czyli dynamicznie przydzielonego adresu IP w sieci lokalnej, dzięki któremu można się połączyć z serwerem. Aby sprawdzić IP konieczne jest podłączenie modułu do jakiegokolwiek terminala (baud rate 115200), zresetowanie ESP8266 i sprawdzenie komunikatu.
10. Funkcja void loop() - główna pętla programu, zajmuje się odczytywaniem danych z UART i obsługą klienta (wyświetleniem zawartości strony).

5.4.2 Kod HTML, CSS, JavaScript i XML

Dokument HTML został napisany w oparciu o języki HTML, CSS, JavaScript. Dane są aktualizowane asynchronicznie przy pomocy dokumentu XML, dzięki technice AJAX - Asynchronous JavaScript and XML. Dzięki takiemu rozwiązaniu w polach wyświetlają się aktualne pomiary bez konieczności odświeżania całej strony.

Struktura dokumentu HTML:

1. Część HEAD napisana w HTML i CSS. Został zastosowany system kodowania Unicode UTF-8 i ustawiony główny styl strony (font, tło).
2. Część SCRIPT zawierająca kod JavaScript.

Funkcja createXmlHttpRequest() - tworzy nowy obiekt XMLHttpRequest do obsługi zapytań asynchronicznych.

Funkcja process() - wykonywana co 200 ms, sprawdza czy obiekt XMLHttpRequest jest otwarty, jeżeli nie, to go otwiera przy pomocy metody PUT i przypisuje funkcję callback handleServerResponse(), gdy obiekt XMLHttpRequest zmieni stan na gotowy.

Funkcja handleServerResponse() - otwiera otrzymany dokument XML od obiektu XMLHttpRequest. Przegląda cały dokument i przypisuje odpowiednie wartości do pól w części BODY.

3. część BODY napisana w HTML i CSS. Przy załadowaniu strony jest wywoływana funkcja JavaScript process(), która rozpoczyna proces przetwarzania otrzymywanych dokumentów XML. Dalej jest zapisana struktura strony podzielona na części DIV, w których znajdują się pola <A> o unikalnych identyfikatorach, pozwalających skryptom JavaScript na lokalizację odpowiednich pól.

Wykorzystywana wersja XML to 1.0. Dokument XML ma prostą strukturę - wszystkie znaczniki znajdują się w głównym znaczniku <Donnees></Donnees>, a każdy znacznik wewnętrz jednoznacznie identyfikuje pomiar.

Rozdział 6

Wnioski

TODO

Bibliografia

- [1] KEC „SEMICONDUCTOR TECHNICAL DATA, KIA7805AP-KIA7824AP BIPOLAR LINEAR INTEGRATED CIRCUIT”, 2010.5.19, Revision No: 1.
- [2] STMicroelectronics „LD1117 series, Low drop fixed and adjustable positive voltage regulators”, December 2006, Rev. 20.
- [3] Atmel Corporation „8-bit Atmel Microcontroller with 4/8/16K Bytes In-System Programmable Flash, ATmega48/V, ATmega88/V, ATmega168/V”, © 2011 Atmel Corporation, Rev. 2545T-AVR-05/11
- [4] STMicroelectronics „UM1724 User manual, STM32 Nucleo-64 board”, November 2016, DocID025833 Rev 11
- [5] STMicroelectronics „RM0008 Reference manual STM32F101xx, STM32F102xx, STM32F103xx, STM32F105xx and STM32F107xx advanced ARM®-based 32-bit MCUs”, November 2015, DocID13902 Rev 16
- [6] Cytron Technologies „Product User’s Manual – HC-SR04 Ultrasonic Sensor”, September 2012, V1.0
- [7] Segway Inc. „User Manual Segway Personal Transporter (PT) i2 SE, x2 SE, x2 SE Turf”, 2014, 24010-00001 aa
- [8] Espressif Systems IOT Team „ESP8266EX Datasheet”, 2015, Version 4.3
- [9] Prolific „PL-2303 USB to RS-232 Bridge Controller Product Datasheet”, August 2002, Document Revision 1.4
- [10] Strona internetowa producenta Pololu „9.7:1 Metal Gearmotor 25Dx48L mm MP 12V with 48 CPR Encoder”, 27.04.2017, <https://www.pololu.com/product/3238>

- [11] STMicroelectronics „L298 DUAL FULL-BRIDGE DRIVER”, January 2000
- [12] Strona internetowa NodeMcu „NodeMcu Connect Things EASY”, 27.04.2017, http://nodemcu.com/index_en.html
- [13] Strona internetowa Arduino „Language Reference”, 27.04.2017, <https://www.arduino.cc/en/Reference/HomePage>
- [14] Projekt esp8266/Arduino na Github.com „Arduino core for ESP8266 WiFi chip”, 27.04.2017, <https://github.com/esp8266/Arduino>
- [15] Projekt Harinadha/STM32_MPU6050lib na Github.com „MPU6050 I2C Library for STM32f103xx family of microcontrollers”, 27.04.2017, https://github.com/Harinadha/STM32_MPU6050lib
- [16] Strona internetowa Eclipse Foundation, Inc. „Eclipse - The Eclipse Foundation open source community website.”, 27.04.2017, <https://eclipse.org>
- [17] CadSoft „EAGLE - EASILY APPLICABLE GRAPHICAL LAYOUT EDITOR Manual”, 2008, Version 5 2nd Edition
- [18] Strona internetowa OpenSTM32 „OpenSTM32 CommunityThe STM32 Systems Resource”, 27.04.2017, <http://www.openstm32.org>
- [19] Strona internetowa MathWorks „MATLAB Documentation”, 27.04.2017, <https://www.mathworks.com/help/matlab/>
- [20] Strona internetowa w3schools „THE WORLD’S LARGEST WEB DEVELOPER SITE”, 27.04.2017, <https://www.w3schools.com/>

