Politechnika Wrocławska

Wydział Elektroniki, Fotoniki i Mikrosystemów

KIERUNEK: Automatyka i Robotyka (AIR)

PRACA DYPLOMOWA INŻYNIERSKA

TYTUŁ PRACY:

System kontroli orientacji rakiety eksperymentalnej

AUTOR:

Łukasz Walczak

PROMOTOR:

dr inż. Wojciech Domski

WROCŁAW 2023

Pracę dedykuję mamie i dziadkom za nieustające wsparcie i wiarę. Dedykuję ją również przyjaciołom, w szczególności Hubertowi, Oli i Filipowi za to, że byli, gdy ich najbardziej potrzebowałem.

Streszczenie

Praca przedstawia projekt systemu stabilizującego obrót rakiety w zadanym kierunku jej ruchu. W toku pracy przeprowadzono symulację takiego systemu z wykorzystaniem silnika MuJoCo na bazie modelu przykładowej rakiety. Zaprojektowano i wytworzono elektronikę mogącą zrealizować sterowanie na rzeczywistym obiekcie. Powstała również konstrukcja mechaniczna pozwalająca na użycie systemu w rzeczywistej rakiecie. Zaproponowano szkielet programu, która umożliwiła zastosowanie takiego systemu w rzeczywistości.

Summary

The thesis presents the design of a rotation stabilization for a rocket in the designed direction of its movement. In the course of the thesis, such a system was simulated using the MuJoCo motor on the basis of a model of an exemplary rocket. Electronics capable of realizing the control were designed and manufactured. A mechanical design was also created that would allow the system to be used in an actual rocket. A sample program schema was also proposed that would allow the use of such a system in the real world.

Słowa kluczowe: rakieta, sterowanie, rakieta eksperymentalna, kontrola orientacji, stabilizacja

Keywords: rocket, control, experimental rocket, orientation control, stabilization

Spis treści

Rozdział 1

Wstęp

Za początek współczesnych rakiet uznaje się 16 marca 1926 roku, kiedy odbył się lot rakiety stworzonej przez Roberta H. Goddarda. Uznaje się ją za pierwszą w historii rakietę na paliwo ciekłe. Podczas lotu trwającego 2,5 sekundy, rakieta osiągnęła wysokość 13m i wylądowała w polu kapusty. Było to preludium do wydarzeń, które nastąpiły zaledwie 40 lat później, gdy człowiek znalazł się w kosmosie.

Po intensywnym wyścigu na Księżyc podczas Zimnej Wojny, rządy największych mocarstw na Świecie, zaczęły stopniowo ograniczać budżety agencji kosmicznych. Pierwotne założenia, aby postawić stopę na Księżycu, nieważne jakim kosztem i jakim nakładem pracy, przestały obowiązywać wraz z wycofaniem się z wyścigu Związku Radzieckiego.

W efekcie tych wydarzeń i decyzji nastąpiła stagnacja przemysłu kosmicznego. Loty kosmiczne zaczęły jawić się jako zdecydowanie zbyt drogie i skomplikowane. Koszty lotów na orbitę starano się ograniczyć poprzez odzyskiwanie rakiet nośnych. Tak zrodziła się idea statków kosmicznych wielokrotnego użytku, jednak elektronika oraz systemy sterowania w tamtym okresie nie były odpowiednio zaawansowane, aby pozwolić na wykonywanie takich operacji bezzałogowo.

Dopiero rozwój elektroniki oraz systemów sterowania w XXI wieku sprawił, że pierwotny cel stał się możliwy do osiągnięcia. W 2016 dokonała tego prywatna firma SpaceX, której rakieta Falcon 9 zdołała wysłać ładunek w kosmos, a następnie w pełni autonomicznie wylądować. Zadanie to było o tyle trudne, że rakieta ląduje pionowo, używając tych samych silników co do startu. Sterowanie takim obiektem można porównać do próby balansowania długim kijem na końcu palca.

Elementem kluczowym do udanego lądowania rakiety była miniaturyzacja elektroniki połączona z jednoczesnym wzrostem mocy obliczeniowej. Pierwsze komputery miały jej zbyt mało, aby móc doprowadzić do udanego autonomicznego lądowania.

Jednym z beneficjentów rozwoju technologii było również modelarstwo rakietowe, które prężnie się rozwija dzięki popularyzacji mikroprocesorów. Jest to hobby polegające na konstruowaniu rakiet eksperymentalnych. Systemy wbudowane dostarczają modelarzom narzędzi do zbierania danych, sterowania kolejnymi fazami lotu, zwiększenia bezpieczeństwa oraz do sterowania rakietą podczas lotu właściwego.

Ostatni element jest szczególnie ważny w optymalizacji osiągów rakiety. Bez żadnej kontroli nad kierunkiem lotu rakiety, modelarze są zdani w dużej mierze na warunki atmosferyczne, co jest problematyczne ze względu na ich zmienność na różnych wysokościach lotu. Modele rakiet są również niedoskonałe, nie są one wykonywane z taką dokładnością, co profesjonalne rakiety. Implementacja modułu odpowiedzialnego za kontrolę lotu rakiety pozwala na optymalizację trajektorii lotu rakiety.

Wyróżnia się dwa sposoby kontroli rakiety. Pierwszy z nich to wektorowanie ciągu, czyli regulowanie kierunku wektora siły napędu rakietowego[11]. Jest to metoda używana w większości profesjonalnych rakiet. Pozwala ona jednak na kontrolę, tylko kiedy napęd produkuje ciąg. Modele rakietowe mają zdecydowanie krótszy czas pracy na silniku, większość lotu odbywa się w warunkach lotu ślizgowego. Lepsze rezultaty daje sterowanie z użyciem powierzchni sterowanych [20].

Najważniejszym aspektem podczas sterowania orientacją rakiety jest zapewnienie stabilności rakiety przy obrocie wokół własnej osi. Brak takiej stabilności minimalizuje wydajność stabilizacji w dwóch pozostałych płaszczyznach. W skrajnych przypadkach sprawia nawet, że sterowanie takie staje się niemożliwe do wykonania.

Celem pracy było stworzenie i zasymulowanie takiego systemu kontroli orientacji rakiety, który pozwoli na stabilizację jej obrotu wokół jednej z osi. Jednym z ograniczeń narzuconych na projekt, było przystosowanie systemu do jego użycia podczas zawodów rakietowych, gdzie regulamin dopuszcza stosowanie takiego systemu jedynie podczas lotu właściwego, czyli od momentu końca pracy napędu, do osiągnięcia apogeum.

1.1 Teza

Możliwe jest wykorzystanie regulatora PID jako sterownika kontroli orientacji rakiety eksperymentalnej.

1.2 Podział pracy

W drugim rozdziale zostało opisane przeprowadzenie badań symulacyjnych. Omówiony został sposób na zamodelowanie przykładowej rakiety w której możliwe byłoby zaimplementowanie systemu stabilizacji. W dalszej kolejności w rozdziale znalazł się opis badań oraz ich wyniki.

Trzeci rozdział zawiera opis elektroniki przygotowanej z myślą o zastosowaniu takiego systemu w rzeczywistej rakiecie. Opisane zostały sekcje: zasilania, czujników oraz mikrokontrolera.

Następny rozdział opisuje teoretyczną implementację pętli sterującej w oparciu o wykorzystaną elektronikę. Natomiast rozdział piąty, opisuje model mechaniki pozwalającej na realizację systemu.

Rozdział końcowy stanowi krótkie podsumowanie pracy, w którym wyszczególniono główne wnioski z badań oraz przedstawiono sugestie dotyczące przyszłych działań. Wnioski obejmują skuteczność zastosowanego modelu stabilizacji rakiety, a dalszy rozwój może skoncentrować się na doskonaleniu istniejącego modelu i ewentualnym dodaniu zaawansowanych funkcji kontroli.

Rozdział 2

Środowisko symulacyjne

W tworzeniu rakiet eksperymentalnych oraz ich systemów bardzo ważnym aspektem jest przetestowanie ich działania w środowisku symulacyjnym. Ze względu na koszt i skomplikowanie takich rakiet, najlepszym rozwiązaniem jest wykonywanie startów tylko takich rakiet, które zostały wcześniej sprawdzone i występuje duże prawdopodobieństwo sukcesu wykonanego lotu. Niestety podczas realizacji pracy nie udało się znaleźć środowiska, które pozwoli na jednoczesne wyznaczenie współczynników aerodynamicznych oraz zasymulowanie systemu stabilizującego lot.

Dlatego zdecydowano się przeprowadzić pracę kaskadowo. Pierwszym elementem symulacji było zaprojektowanie samego modelu rakiety w środowisku symulacyjnym *OpenRocket*, gdzie wyznaczone zostały takie parametry jak wielkości lotek, kształt głowicy czy sama długość rakiety [15]. Znając model rakiety, możliwe było przeprowadzenie symulacji CFD (z ang. *Computational fluid dynamics*), które pozwoliły na wyznaczenie współczynników aerodynamicznych [17].

Na podstawie dostatecznej znajomości parametrów modelu rakiety, możliwe było wykonanie i przeprowadzenie symulacji systemu stabilizującego obrót rakiety. Zostało to wykonane z wykorzystaniem środowiska symulacyjnego *MuJoCo*, które pozwala w prosty sposób zasymulować dowolny model, od robota humanoidalnego, do rakiety [21]. Same symulacje są jednocześnie wizualizowane, dzięki czemu w łatwy i szybki sposób można ocenić działanie systemu.

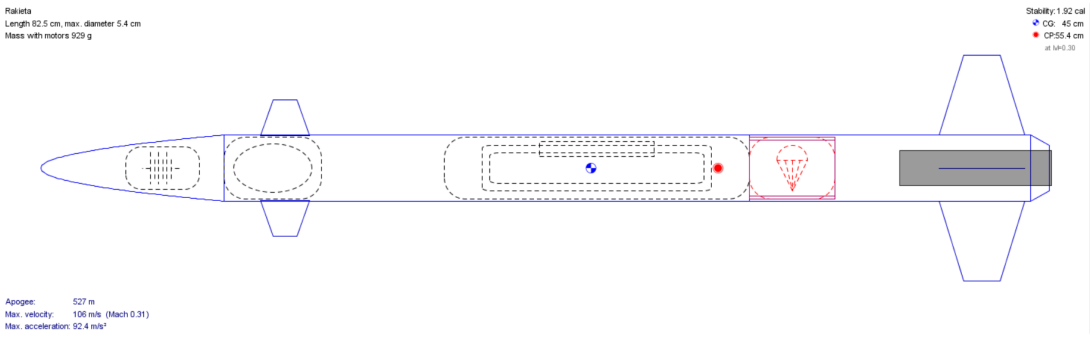
2.1 Opracowanie modelu rakiety

Najpopularniejszym środowiskiem używanym przez modelarzy rakiet jest *OpenRocket*, czyli program pozwalający na przeprowadzanie symulacji bardzo prostych rakiet. Rozwiązuje on problem doboru parametrów rakiety tak, aby rakieta miała odpowiednią prędkość, wysokość lotu oraz stabilność [15]. Praca w środowisku polega na opisaniu rakiety poprzez wybór odpowiednich komponentów takich jak lotki, mocowanie silnika, korpus, czy głowica z listy gotowych elementów. Następnie do każdej części ustawiane są jej wymiary oraz materiał z którego są wykonane, po czym program sam przelicza jej masę.

Celem doboru parametrów jest przede wszystkim osiągnięcie zadowalającej stabilności. Mierzona jest ona w kalibrach (z ang. *caliber*), które określają stosunek średnicy rakiety do odległości pomiędzy środkiem parcia a środkiem ciężkości. Przyjęte jest, że stabilność na poziomie $1cal$, jest minimalną wartością, którą należy osiągnąć w celu zapewnienia stabilnego lotu w trakcie pracy silnika.

OpenRocket oferuje również szeroką gamę dostępnych silników, dla których zdefinio-

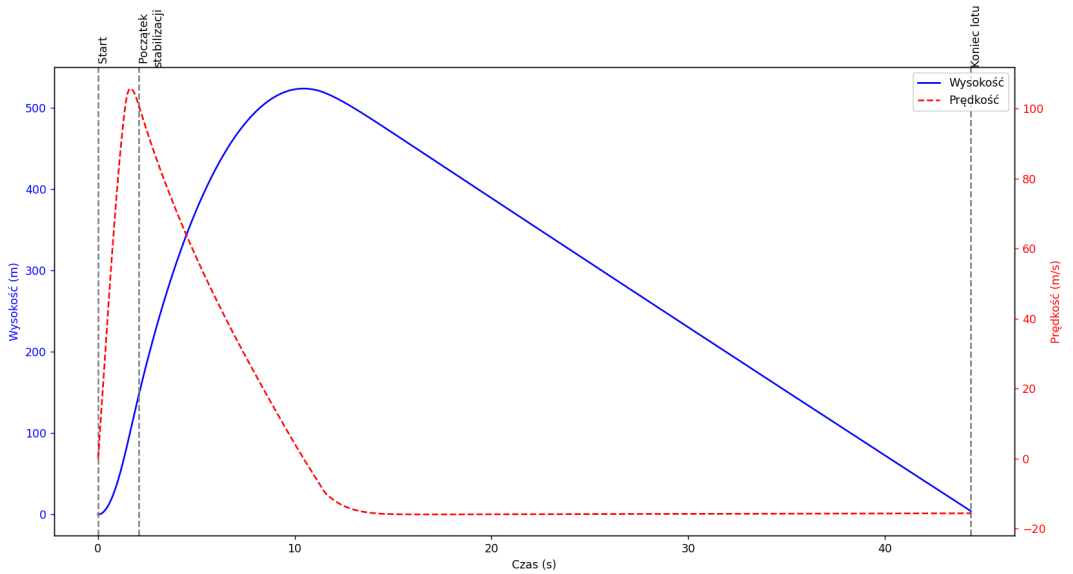
wane są masa, rozmiar oraz krzywa ciągu, na podstawie której dokonywane są obliczenia dotyczące prędkości oraz wysokości lotu.



Rysunek 2.1: Model rakiety zaprojektowany z wykorzystaniem środowiska *OpenRocket*

Na rysunku 2.1 przedstawiono przykładowy model rakiety stworzony na potrzeby pracy. Widać, że stabilność w przypadku tego modelu wyniosła $1,92cal$, natomiast sama rakieta powinna polecieć na wysokość około $530m$ i osiągnąć najwyższą prędkość około $100\frac{m}{s}$.

Środowisko oferuje również przeprowadzenie pełnych symulacji lotu, które pozwalają na stworzenie wykresów zależności wysokości i prędkości rakiety od czasu. Podczas przeprowadzania symulacji, można zdefiniować: kąt ataku, prędkość i kierunek wiatru oraz długość wyrzutni. Szczególnie ważny jest ten ostatni parametr, ponieważ przez pierwsze ułamki sekund od startu, rakieta ma stosunkowo niską prędkość, co wpływa negatywnie na jej stabilność. Dlatego stosuje się wyrzutnie, czyli pionowo zamontowane szyny, do których przyczepiana jest rakieta, o długości kilku do kilkunastu metrów. Stabilizują one rakietę w pierwszej fazie lotu tak, aby rakieta, opuszczając wyrzutnię, miała odpowiednią prędkość i orientację. Zapewnia to odpowiednią stabilność rakiety w momencie działania największych przyspieszeń, optymalizując jednocześnie jej lot.



Rysunek 2.2: Symulacja lotu rakiety przeprowadzona w środowisku *OpenRocket*

Celem modelowania rakiety, było uzyskanie odpowiedniego profilu lotu. Wzrost prędkości wpływa na wzrost wydajności działania systemu, stąd starano się uzyskać jak najwyższą prędkość podczas lotu właściwego. Ograniczeniem narzuconym na model było

wykorzystanie silnika klasy F, czyli najmocniejszego silnika dostępnego do zakupu bez posiadania licencji startowej rakiet dużej mocy [8]. Rysunek 2.2 przedstawia uzyskany profil lotu. Widzimy, że rakieta osiąga pułap $500m$, a w momencie aktywacji stabilizacji, prędkość wynosi około $100\frac{m}{s}$. Przy takiej konfiguracji startowej lot właściwy powinien trwać około 7s.

Gotowy model zapewniający odpowiednie warunki, pozwolił na odczytanie parametrów charakteryzujących aerodynamikę rakiety. Ze środowiska *OpenRocket*, wyznaczono:

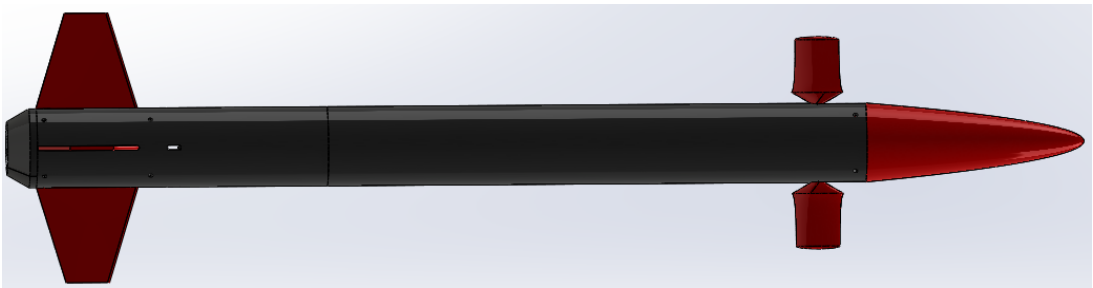
- współczynnik oporu $c_D = 0,59$,
- powierzchnię rzutu ciała na płaszczyznę $A = 0,002m^2$,
- prędkość w momencie aktywacji systemu $v_0 = 100\frac{m}{s}$.

2.1.1 Symulacje przepływowe

Aby przeprowadzić symulację działania systemu w środowisku *MuJoCo*, istotnym elementem jest uzyskanie informacji na temat momentów siły generowanych przez system w zależności od prędkości rakiety i wychylenia lotek. Aby dokładnie określić te zależności, zdecydowano się skorzystać z symulacji przepływowych CFD (z ang. *Computational Fluid Dynamics*) [2].

Symulacje przepływowe CFD opierają się na równaniach Naviera-Stokesa, które są fundamentalnymi równaniami opisującymi ruch płynu[4]. Równania te uwzględniają zasady fizyki, takie jak równowaga pędu i zachowanie masy, co pozwala na modelowanie złożonych procesów przepływu. W kontekście symulacji systemu rakietowego CFD umożliwiają analizę oddziaływania strumienia powietrza na konkretną geometrię rakiety, co pozwala na prognozowanie generowanych sił i momentów obrotowych.

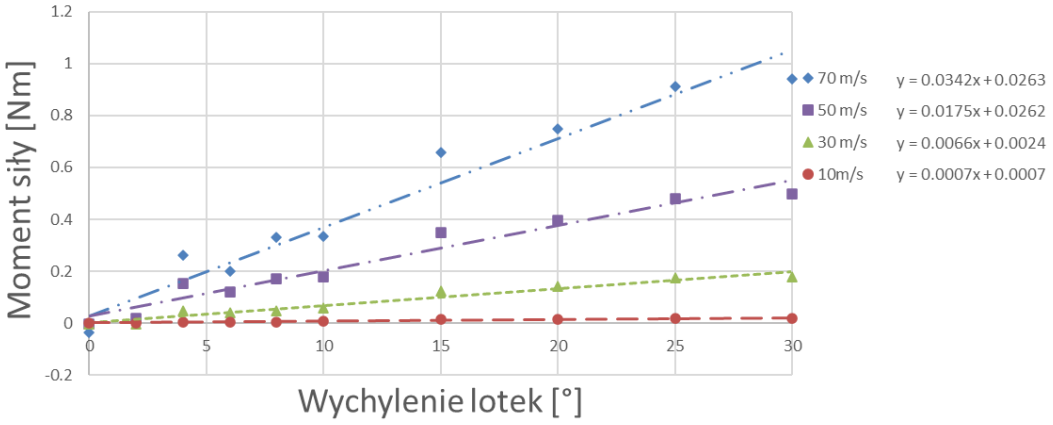
Proces ten rozpoczął się od stworzenia w środowisku *SolidWorks* projektu rakiety, który odzwierciedlał ten ze środowiska *OpenRocket* [12]. Projekt widoczny jest na rysunku 2.3. Widzimy, zestawiając z rysunkiem 2.1, że udało się uzyskać wierne odwzorowanie. Wymiary zostały przeniesione tak, aby zgadzały się 1:1. Wykorzystując wtyczkę symulacja przepływu (z ang. *Flow Simulation*), zdefiniowano symulację przepływu gazu, którego parametry zostały zdefiniowane tak, aby odzwierciedlały przepływ powietrza wzdłuż rakiety w trakcie lotu.



Rysunek 2.3: Model rakiety stworzony w programie *SolidWorks*

Następnie przeprowadzono szereg symulacji, dla zmiennych parametrów prędkości oraz kąta wychylenia lotek. Z każdego scenariusza symulacyjnego odczytano, jakie momenty siły działają na rakietę wzdłuż osi ruchu. Stworzono symulację dla kątów wychylenia: $0, 2, 4, 6, 8, 10, 15, 20, 25, 30°$ oraz dla prędkości $10, 30, 50, 70\frac{m}{s}$. Zestawienie momentów siły względem kąta wychylenia lotek, można zaobserwować na wykresie 2.4. Można zauważyć,

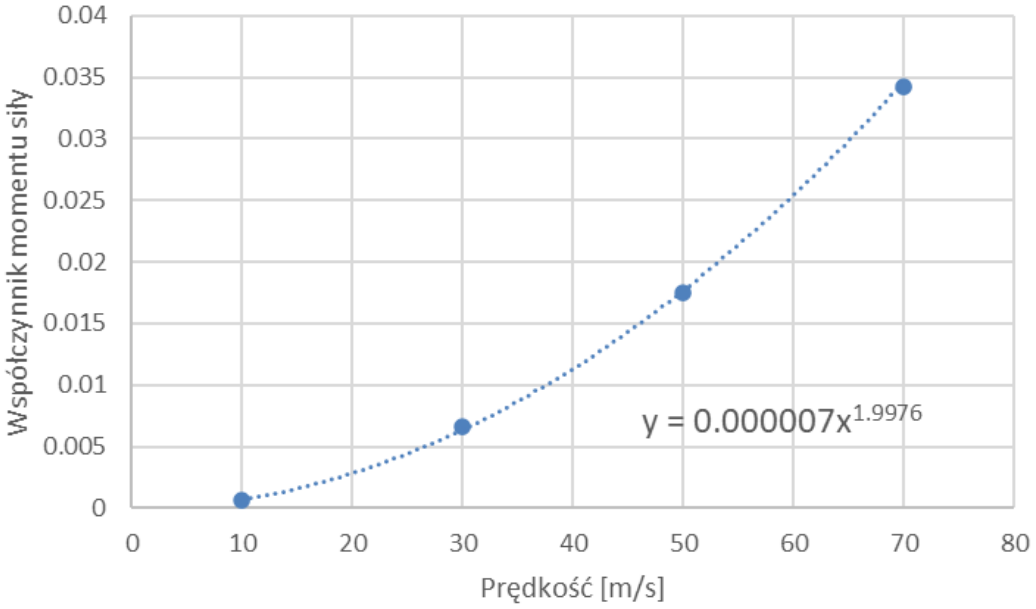
że moment rośnie wprost proporcjonalnie względem kąta wychylenia. Regresje liniowe zbiegają się w środku układu współrzędnych, co oznacza, że przy braku prędkości, lub przy braku wychylenia lotek, system nie generuje żadnych momentów siły.



Rysunek 2.4: Zależność momentu siły od kąta wychylenia lotek, dla różnych prędkości

Znając zależność pomiędzy momentem siły a kątem wychylenia lotek, sprawdzono, w jaki sposób prędkość rakiety wpływa na generowane momenty. Zostało to zrealizowane poprzez wyznaczenie zależności pomiędzy współczynnikami a odczytanymi z regresji liniowych z wykresu 2.4, a prędkością rakiety. Następnie ponownie została wykorzystana regresja liniowa, która pozwoliła ustalić, że generowane momenty rosną w kwadracie do aktualnej prędkości liniowej rakiety. Wyznaczony dzięki temu wzór (2.1) pozwala w przybliżeniu obliczyć moment siły działający w danym momencie na rakietę.

$$M_Z = c_C \alpha v^2 \tag{2.1}$$



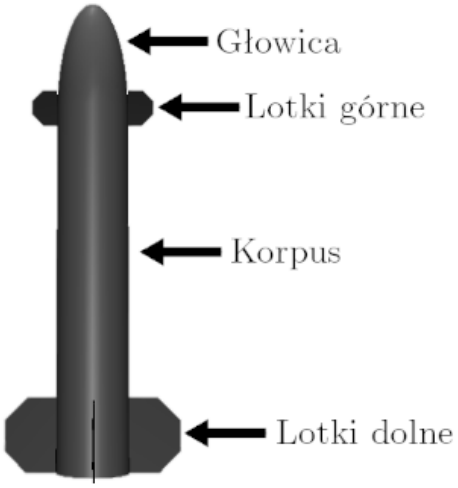
Rysunek 2.5: Zależność współczynnika momentu siły od prędkości rakiety

2.2 Symulacje działania systemu kontroli orientacji

Po opracowaniu modelu rakiety oraz wyznaczeniu jej współczynników aerodynamicznych, możliwe było przystąpienie do przeprowadzenia badań symulacyjnych systemu w środowisku *MuJoCo*. Celem tego etapu pracy było przeprowadzenie symulacji, które pozwolą na odpowiedni dobór nastaw regulatora PID. Została przeprowadzona również symulacyjna walidacja działania systemu dla różnych zadań, takich jak: odpowiedź skokowa systemu, śledzenie trajektorii oraz reakcja na warunki losowo generowane zakłócenia.

2.2.1 Model rakiety

MuJoCo pozwala na definiowanie obiektu i otaczającego go świata z wykorzystaniem plików *.xml*, na podstawie którego, przeprowadzana jest później symulacja. Modele składają się z podstawowych elementów takich jak: ciało (z ang. *body*), przegub (z ang. *joint*) i kontroler (z ang. *control*). Dla wskazanych elementów możliwe jest ustawienie odpowiednich parametrów jak: masa (z ang. *mass*), inercja (z ang. *inertial*), tarcie (z ang. *friction*).



Rysunek 2.6: Model stworzony w celu wizualizacji w środowisku *MuJoCo*

Model rakiety widoczny na rysunku 2.6 powstał w oparciu o kształt walca o odpowiednim rozmiarze, do którego zostały przymocowane kolejne elementy geometryczne reprezentujące elementy rakiety. Na szczycie zamontowano głowicę, która została zdefiniowana jako elipsoida, co stanowi najwierniejsze odwzorowanie rzeczywistego kształtu w środowisku symulacyjnym. Wizualna różnica kształtu nie ma jednak wpływu na właściwości aerodynamiczne obiektu, ponieważ te są definiowane osobno, niezależnie od kształtu modelu.

Lotki dolne oraz górne, podobnie jak w symulacjach pochodzących ze środowiska *OpenRocket*, mają kształt trapezoidalny i zostały zdefiniowane przy pomocy opcji siatka (z ang. *mesh*), gdzie podaje się kolejne punkty na płaszczyźnie współrzędnych. Wykorzystując parametry masa (z ang. *mass*) i inercja (z ang. *inertial*), do którego dodatkowo został dodany parametr pos (z ang.*pos*), została zdefiniowana masa oraz środek ciężkości modelu.

Oprócz samego modelu geometrycznego obiekt zawiera również dwa przeguby rotacyjne odpowiedzialne za obracanie lotek górnych. Narzucone zostało na nie ograniczenie działania w zakresie $[-30°; +30°]$, co jest odwzorowaniem działania lotek w rzeczywistym modelu. Wychylenie lotek o kąt większy niż dopuszczalny limit spowodowałoby działanie

bardziej zbliżone do działania hamulca aerodynamicznego niż sterowania, stąd obecność tego ograniczenia. Kolejnym zagrożeniem płynącym ze zbyt dużego wychylenia, byłoby zbytnie obciążenie serwomechanizmu, który nie byłby on w stanie pokonać siły działającej na lotki. Dodatkowo dodano wyznaczone metodą prób i błędów współczynniki tarcia oraz tłumienia, które sprawiają, że lotki nie przemieszczają się z jednej pozycji do drugiej natychmiastowo, tylko z pewnym opóźnieniem, co jest odzwierciedleniem działania rzeczywistego serwomechanizmu.

2.2.2 Badania symulacyjne

Środowisko *MuJoCo* oferuje szereg narzędzi pozwalających na kontrolowanie obiektu oraz odczytywanie jego stanów [5]. Do ustawiania warunków początkowych w przestrzeni trójwymiarowej takich jak prędkości i momenty obrotowe wykorzystano zmienną q_{vel}, która pozwala na zdefiniowanie lub odczytanie prędkości liniowych oraz kątowych ciała w jego układzie współrzędnych

$$q_{vel} = \begin{pmatrix} v_X \\ v_Y \\ v_Z \\ \omega_X \\ \omega_Y \\ \omega_Z \end{pmatrix}. \tag{2.2}$$

Pierwsze trzy współrzędne wektora (2.2) reprezentują prędkości liniowe, zaś trzy ostatnie odpowiadają za prędkości kątowe ciała.

Wektor q_{frc} został wykorzystany do zdefiniowania dodatkowych sił działających na obiekt, poza siłą grawitacji, która jest domyślnie uwzględniana w symulacji przez silnik *MuJoCo*. W przypadku symulowanej rakiety wektor ten został wykorzystany do zdefiniowania sił wynikających z oporu powietrza. Analogicznie jak w wektorze (2.2), pierwsze trzy współrzędne wektora (2.3), odpowiadają za siły działające na rakietę w układzie współrzędnych ciała. Kolejne trzy współrzędne reprezentują momenty siły działające na poszczególne osie układu.

$$q_{frc} = \begin{pmatrix} F_{DX} \\ F_{DY} \\ F_{DZ} \\ 0 \\ 0 \\ M_Z \end{pmatrix}. \tag{2.3}$$

Pierwsze trzy składowe wektora (2.3), można wyznaczyć na bazie ogólnego wzoru na opór aerodynamiczny oraz parametrów odczytanych ze środowiska *OpenRocket*

$$F_D = \begin{pmatrix} F_{DX} \\ F_{DY} \\ F_{DZ} \end{pmatrix} = \begin{pmatrix} -\frac{1}{2} c_D \rho v v_X A \\ -\frac{1}{2} c_D \rho v v_Y A \\ -\frac{1}{2} c_D \rho v v_Z A \end{pmatrix} = -\frac{1}{2} c_D \rho v A \begin{pmatrix} v_X \\ v_Y \\ v_Z \end{pmatrix}, \tag{2.4}$$

gdzie:

- c_D – współczynnik oporu powietrza, wyznaczony w środowisku *OpenRocket*,
- ρ – gęstość powietrza,
- v – prędkość bezwzględna obiektu,
- v_{XYZ} – prędkość wzdłuż konkretnej osi,
- A – powierzchnia rzutu ciała na płaszczyznę.

Moment siły powstający w wyniku pracy systemu stabilizującego został wyznaczony na podstawie analizy przeprowadzonych poprzednio symulacji CFD i przedstawia się on wzorem (2.1).

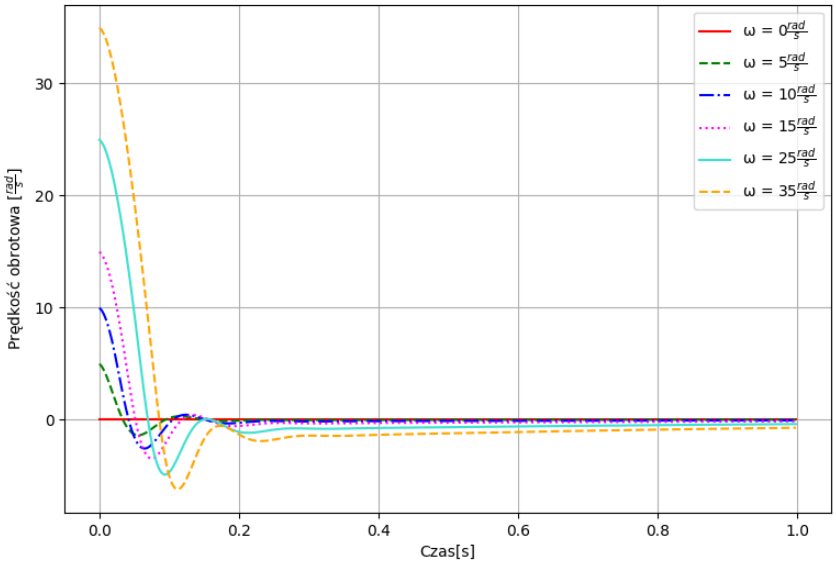
Kolejnym ważnym elementem symulacji była funkcja *mj_step* odpowiadająca za wykonywanie kolejnych kroków symulacji z krokiem czasowym $\Delta t = 0,002s$. W każdym przebiegu pętli program wyliczał na nowo siły działające na rakietę oraz dokonywał wizualizacji obliczeń. Dodatkowo co krok, było wyznaczane nowe sterowanie z użyciem algorytmu PID. Zmienna *qpose* była wykorzystywana do odczytywania aktualnej prędkości obrotowej wzdłuż osi Z, na jej podstawie był wyznaczany uchyb regulacji. Samo sterowanie zostało zdefiniowane w pliku *.xml* w taki sposób, że zmiana wartości zmiennej *ctrl* w zakresie $[-1; 1]$, powodowała zmianę pozycji lotek.

2.2.3 Wyniki symulacji

Aby sprawdzić działanie systemu, wykonano symulację, dla różnych zadań takich jak minimalizowanie prędkości obrotowe, śledzenie trajektorii, czy reakcja modelu na warunki losowe. W tym celu zostały zdefiniowane pewne warunki początkowe. Ze względu na potencjalne zastosowania takiego systemu na zawodach hobbystów modelarstwa rakietowego, narzucone są na niego pewne ograniczenia wynikające z regulaminu tychże zawodów, do których się zastosowano. Jednym z takich ograniczeń, jest ograniczenie pracy systemu tylko w wąskim oknie czasowym, od wypalenia się silnika głównego, do osiągnięcia przez rakietę apogeum. Symulacja zaczyna się w momencie wypalenia silnika, kiedy prędkość rakiety przestaje rosnąć. Jako wartość prędkości początkowej w symulacji, przyjęto $70\frac{m}{s}$.

Na podstawie modelu ze środowiska *OpenRocket*, jako współczynnik oporu powietrza c_D, przyjęto wartość $0,51$, pole powierzchni rzutu ciała na płaszczyznę A wyniosło $0,002m^2$. Natomiast za gęstość powietrza ρ, przyjęto zgodnie z definicją atmosfery wzorcowej, której używa się w lotnictwie, wartość $1,2255\frac{kg}{m^3}$, co stanowi gęstość powietrza na poziomie morza przy ciśnieniu $1013,25hPa$ i temperaturze $15°C$.

Pierwszą z przeprowadzonych symulacji było zestawienie wyników działania systemu w zależności od początkowej prędkości obrotowej rakiety. Na wykresie 2.7 przedstawiono zależność prędkości obrotowej rakiety od czasu. Można zauważyć, że nawet przy relatywnie dużej prędkości obrotowej wynoszącej $\omega = 35\frac{rad}{s}$, układ jest w stanie ją zniwelować do wartości bliskich zeru w ciągu sekundy.

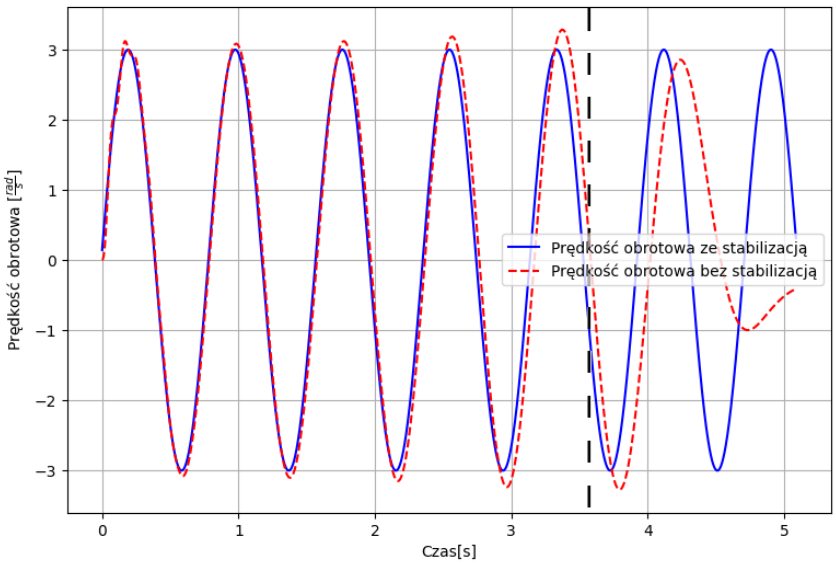


Rysunek 2.7: Porównanie działania systemu dla różnych warunków początkowych

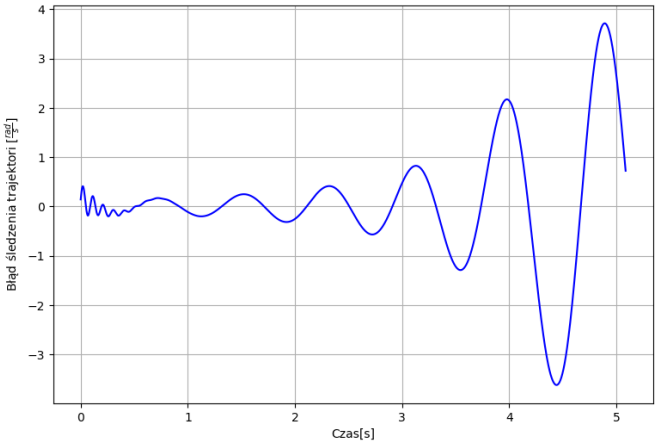
Kolejną z symulacji było sprawdzenie efektywności działania systemu przy śledzeniu trajektorii zadanej, za którą przyjęto funkcję

$$f(t) = 3sin(4t). \tag{2.5}$$

Porównanie sygnału docelowego oraz wynikowego zostało przedstawione na wykresie 2.8. Przez pierwsze $3,5$ sekundy lotu, rakieta jest w stanie dość dobrze śledzić trajektorię, jednak po tym momencie wydajność sterowania znacząco spada i narasta błąd śledzenia. Obrazuje to wykres 2.9 przedstawiający zmianę błędu śledzenia trajektorii w czasie. Powodem takiego stanu rzeczy, jest powiązanie pomiędzy momentem obrotowym generowanym przez lotki a prędkością liniową. Moment rośnie wprost proporcjonalnie do kwadratu prędkości, dlatego w momencie, kiedy rakieta zbliża się do apogeum, a tym samym prędkość spada do zera, moment siły również zaczyna zbiegać do zera.



Rysunek 2.8: Śledzenie trajektorii zadanej



Rysunek 2.9: Błąd śledzenia trajektorii zadanej

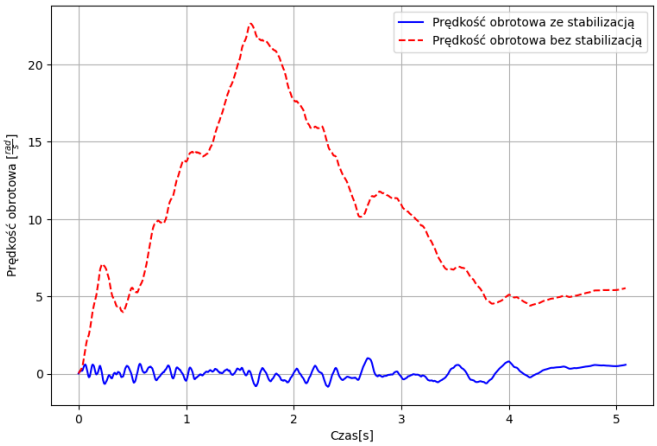
Jako ostatnią przeprowadzono symulację prawdziwego lotu, kiedy to rakieta napotyka turbulencje wynikające z niedoskonałości jej budowy oraz z nierównego rozkładu powietrza w atmosferze. Aby zasymulować ten efekt, skorzystano z funkcji *rand()* pochodzącej z biblioteki standardowej języka C [13]. Generuje ona pseudolosową liczbę z przedziału $[0; 32767]$. Przedział ten został przekształcony tak, aby otrzymywać dwie liczby

$$\phi \in [-1,5; 1,5], \ \theta \in [-4; 4]. \tag{2.6}$$

ϕ jest składową stałą turbulencji, generowaną 5 razy na sekundę, natomiast θ generowana jest w każdym korku symulacji. Turbulencje są wyliczane zgodnie ze wzorem (2.7). Gdzie $F()$ to funkcja odpowiedzialna za filtrację sygnału. Jest to filtr dolnoprzepustowy pierwszego rzędu. Waży on aktualny sygnał z wagą 0,1, natomiast poprzednia wartość sygnału jest ważona z wartością 0,9. Pozwala to na wygładzenie sygnału.

$$x = F(\phi + \theta) \tag{2.7}$$

Wykres 2.10 przedstawia prędkość obrotową rakiety dla takich samych turbulencji, w przypadku gdy system stabilizujący jest aktywny i nieaktywny. Przy nieaktywnych lotkach, rakieta osiąga prędkość obrotową około $24\frac{rad}{s}$, natomiast dla symulacji ze stabilizacją, wartość ta nie przekracza $1\frac{rad}{s}$.



Rysunek 2.10: Porównanie rakiety z i bez stabilizacji, dla losowo generowanych zakłóceń

Rozdział 3

Obwód elektroniczy

Implementacja systemu w rzeczywistej rakiecie eksperymentalnej wymagała zaprojektowania elektroniki pozwalającej na określenie położenia rakiety w ogólnym układzie współrzędnych oraz na sterowanie serwomechanizmami odpowiedzialnym za zmianę położenia lotek.

W celu zmniejszenia masy rakiety zdecydowano się na połączenie elektroniki sterującej lotkami oraz komputera pokładowego rakiety sterującego poszczególnymi fazami lotu, w jeden moduł. Taki zabieg pozwolił na stworzenie uniwersalnego i taniego komputera pokładowego, który mógłby znaleźć zastosowania w różnych rakietach, nawet gdy system kontroli orientacji byłby nieaktywny.

Poza mikrokontrolerem, złączami do serwomechanizmów i inercyjną jednostką pomiarową (z ang. *inertial measurement unit*, IMU), w architekturze systemu uwzględniono:

- kartę pamięci, odpowiedzialną za redundancję przy akwizycji danych pomiarowych,
- generator piezoelektryczny informujący użytkownika o stanie rakiety, bez konieczności zbliżania się do niej,
- konwerter USB-UART, odpowiadający za proste wgrywanie oprogramowania na mikrokontroler ESP,
- system odzysku, czyli tranzystory sterujące pirotechniką odpowiedzialną za np. odpalenie spadochronu.

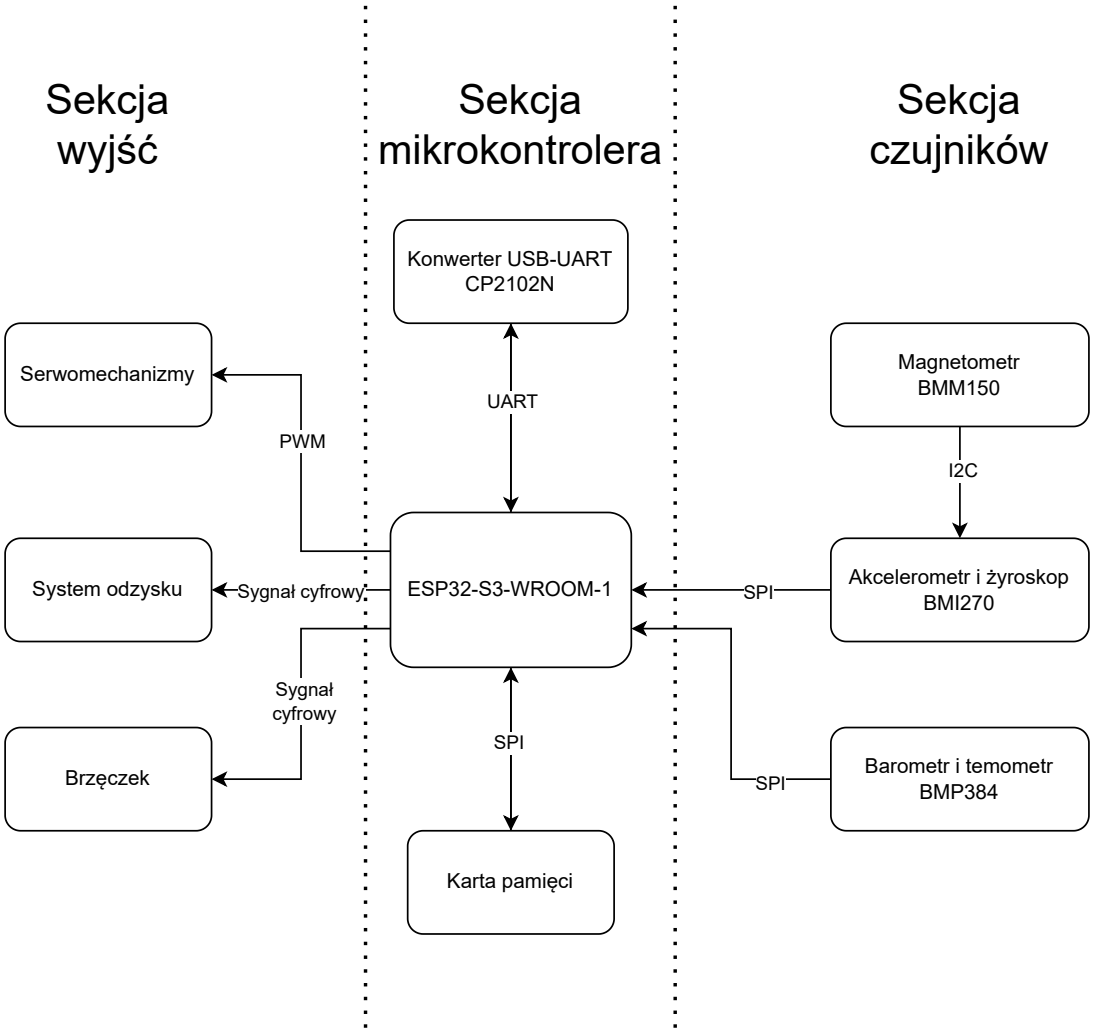
Cała architektura została przedstawiona na rysunku 3.1. Natomiast cały schemat połączeń elektronicznych widoczny jest w załączniku B na rysunku 1. Widać na nim, że trzonem całego systemu jest mikrokontroler *ESP ESP32-S3-WROOM-1* [7]. Jego zadanie polegało na komunikacji z czujnikami, a następnym badaniu zebranych danych o siłach, momentach i położeniu były przetwarzane i zapisywaniu w pamięci flash oraz na karcie SD. Równocześnie na podstawie informacji o prędkości obrotowej było obliczane sterowanie dla serwomechanizmów.

Dobór mikrokontrolera został oparty o łatwość i szeroki zakres możliwości konfiguracji, większość z jego wyjść można dowolnie skonfigurować, co pozwala na skonfigurowanie dowolnego interfejsu na dowolnym wyjściu mikrokontrolera, co upraszcza poprowadzenie ścieżek, a także skraca ich długość. Znaczącą rolę odegrała również pojemność dostępnej pamięci RAM i Flash, co stanowiło kluczowy parametr w zapewnieniu redundancji dla zbierania telemetrii rakiety.

Konstrukcja elektroniki została wykonana w technologii płytki drukowanej PCB (z ang. *Printed Circuit Board*), co pozwoliło na minimalizację jej rozmiaru, przy jednoczesnym

zapewnieniu stabilnych połączeń pomiędzy komponentami. Tego rodzaju płytka jest również bardzo łatwa w reprodukcji w przyszłości, znając wykaz elementów znajdujących się na niej i ich umiejscowienie, można szybko wytworzyć nowy egzemplarz elektroniki.

Gotowy moduł elektroniki zgodny z poniższym opisem, przedstawiono na rysunku 3.6.



Rysunek 3.1: Architektura systemu

3.1 Sekcja zasilania

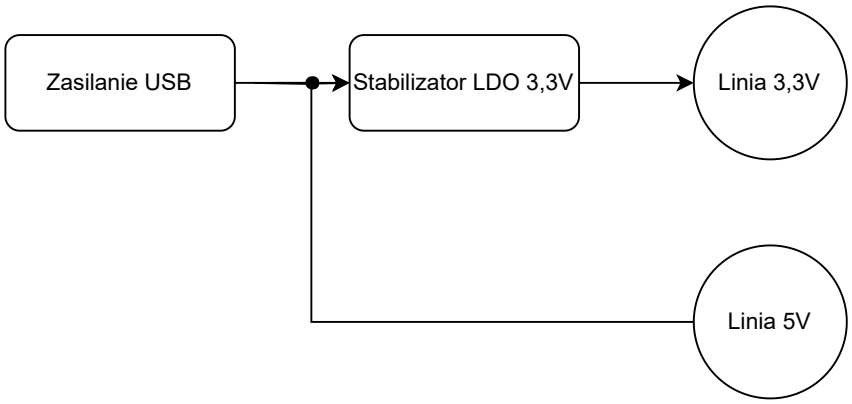
Projekt sekcji zasilania został wykonany z myślą o możliwości zasilania elektroniki z dwóch różnych źródeł oraz o zabezpieczeniu prądowym elektroniki. Zabieg ten był wymagany, aby moduł mógł być zasilony przy użyciu przewodu USB. Ułatwiło to rozwój oprogramowania, dzięki usprawnieniu procesu wgrywania programu do mikrokontrolera. Natomiast zasilanie zewnętrze z użyciem akumulatorów, było wymagane, aby możliwe było zamocowanie sekcji sterującej w rakiecie.

Dualne zasilanie było możliwe dzięki zastosowaniu diod prostowniczych Schottky'ego. Są to diody typu metal-półprzewodnik, co odróżnia je od standardowych diod typu p-n. Charakteryzują się one wyjątkowo małą bezwładnością, dzięki czemu doskonale nadają się do sygnałów szybkozmiennych. Innymi zaletami są minimalna strata mocy oraz duża odporność na wstrząsy, co sprawiło że doskonale nadawały się do zastosowania w rakiecie.

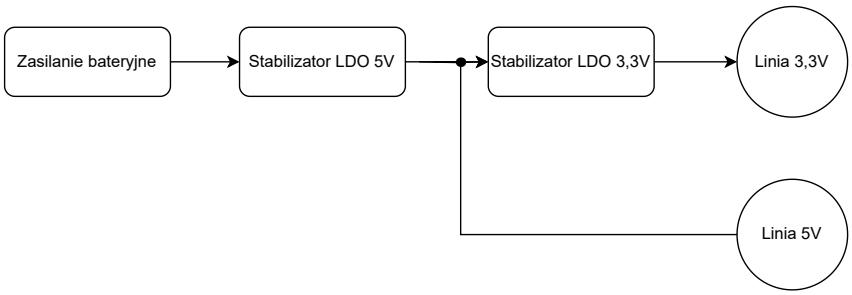
W tym przypadku wybrano model *1N5819HW-7-F*, który jest w stanie pracować przy napięciu do 40V w kierunku przewodzenia, przy poborze prądu do 1A [6].

Na liniach pochodzących ze złącza USB-C, dodatkowo zastosowano diody ochronne przed wyładowaniami elektrostatycznymi, które dość często pojawiają się przy czynnościach takich jak podpinanie lub odpinanie złącza. Brak zabezpieczenia mógłby spowodować uszkodzenie sprzętu. W projekcie zostały użyte diody *LESD5D5.0CT1G* [10].

Po przejściu przez sekcję zasilania, napięcie w zależności od źródła trafiają do jednego z dwóch stabilizatorów napięć LDO (z ang. *Low Dropout Regulator*). Napięcie pochodzące ze złącza USB, jest równe 5V, trafia bezpośrednio na stabilizator, który obniża je do 3,3V oraz na linię 5V. Natomiast napięcie bateryjne może być znacznie wyższe, zatem najpierw trafia ono na stabilizator obniżający je do wartości 5V, dopiero stamtąd, przechodzi ono przez stabilizator generujący napięcie 3,3V. Zdecydowano się na wybór siostrzanych stabilizatorów *AMS1117-3.3* oraz *AMS1117-5.0*, są to jedne z najszerzej dostępnych na rynku komponentów tego typu, co udowadnia ich niezawodność i szeroką gamę zastosowań. Podobnie jak diody Schottky'ego, są one w stanie pracować na natężeniu prądu do 1A.



Rysunek 3.2: Architektura sekcji zasilania dla zasilania poprzez USB



Rysunek 3.3: Architektura sekcji zasilania dla zasilania bateryjnego

Architektura sekcji zasilania została przedstawiona na rysunkach 3.2 i 3.3. Napięcie z lini 3,3V, trafia do układów odpowiadających za logikę systemu: mikrokontrolera, sekcji czujników, konwertera USB-UART. Natomiast napięcie z linii 5V potrzebne jest do zasilania serwomechanizmów, systemu odzysku oraz generatora piezoelektrycznego.

Pozostałymi elementami zaliczanymi do sekcji zasilania są dwie diody LED sygnalizujące o obecności zasilania na liniach 5V i 3,3V. Kondensatory i rezystory odpowiedzialne za filtrację, które również zostały uwzględnione przy projektowaniu układu, zostały dobrane zgodnie z zaleceniami producentów.

3.2 Mikrokontroler

Wybór mikronontrolera został oparty o wcześniejsze doświadczenie z jego wykorzystaniem. Posiada on szeroki zakres interfejsów komunikacyjnych, które zostały wykorzystane podczas realizacji pracy. Do komunikacji z czujnikami, użyty został interfejs SPI (z ang. *Serial Peripheral Interface*), w którym komunikacja z mikrokontrolerem odbywa się w pełnym trybie dwukierunkowym. Składa się on z 4 linii:

- CS (z ang. *Chip Select*), która służy do wyboru urządzenia peryferyjnego, z którym będziemy się aktualnie komunikować, ponieważ pozostałe linie są wspólne dla każdego z peryferjów,
- SCK (z ang. *Serial Clock*), czyli sygnał zegarowy synchronizujący komunikację obu urządzeń,
- MOSI (z ang. *Master Out Slave In*), która jest wykorzystywana do komunikacji jednokierunkowej z kontrolera do peryferium,
- MISO (z ang. *Master In Slave Out*), która jest wykorzystywana do komunikacji jednokierunkowej z peryferium do kontrolera.

Interfejs charakteryzuje się wysoką prędkością komunikacji, co jest kluczowe w osiągnięciu możliwie najmniejszego czasu odbierania danych z czujników. Interfejs ten został również wykorzystany do przesyłu danych na kartę SD, służącą do archiwizacji danych o trajektorii.

Kolejnym ważnym interfejsem wykorzystanym w projekcie, był interfejs UART (z ang. *Universal Asynchronous Receiver-Transmitter*), który został wykorzystany do komunikacji pomiędzy programatorem a mikrokontrolerem. Pozwala on na asynchroniczną komunikację dwukierunkową, przy założeniu odpowiedniego zsynchronizowania częstotliwości komunikacji w nadajniku i odbiorniku.

(a) ESP32-S3-WROOM-1 (b) ESP32-S3-WROOM-1U

Rysunek 3.4: Możliwe konfiguracje mikrokontrolera EPS32-S3-WROOM-1

Mikrokontroler był również połączony z systemem odzysku, który byłby odpowiedzialny za sterowanie odpalaniem ładunków pirotechnicznych. Zaimplementowane zostały trzy

tego typu wyjścia w celu zapewnienia redundancji, gdy jeden z ładunków nie wypali lub gdy rakieta będzie miała więcej niż jeden ładunek. W ramach działalności w jednym z kół naukowych planowane było późniejsze użycie tej płytki w rakiecie składającej się z dwóch stopni, zatem jeden ładunek byłby użyty do rozdzielenia stopni w trakcie lotu, drugi do odpalenia silnika w drugim stopniu, natomiast trzecie wyjście sterowałoby odpaleniem spadochronu dla drugiego stopnia. Sterowanie oparte jest o wyjście analogowe, przy pomocy którego przełączany jest tranzystor typu N, zwierając obwód odpowiedzialny za działanie materiałów pirotechnicznych.

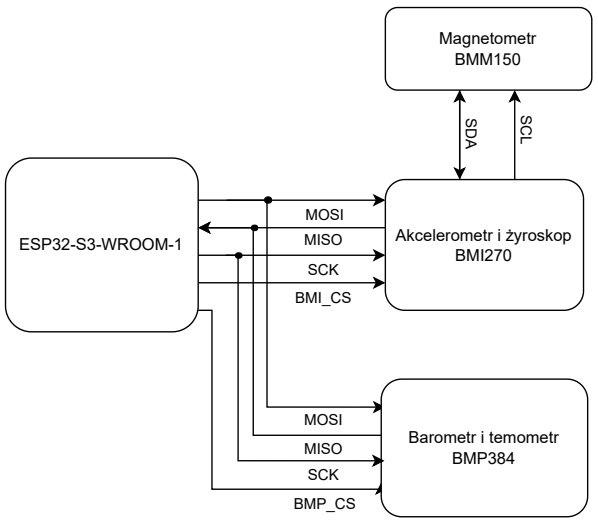
Do sterowania generatorem piezoelektrycznym oraz serwomechanizmami, wykorzystywany jest modulowany sygnał o zmiennej szerokości impulsów, PWM (z ang. *Pulse-Width Modulation*). Jest to prosty sposób na zasilanie układów działających na napięciu przemiennym, jakim jest np. generator piezoelektryczny, gdy źródło generuje napięcie stałe.

W zaprojektowanej elektronice możliwe jest użycie mikrokontrolera w konfiguracji 3.4a, jak i 3.4b. Różnią się one jedynie obecnością anteny. Konfiguracja (a) zawiera wbudowaną antenę, która może być wykorzystana do połączenia się z mikrokontrlerem poprzez *Wi-Fi* oraz *Bluetooth*. Natomiast druga konfiguracja nie posiada anteny, a w jej miejscu umiejscowione jest złącze, pozwalające jest na podłączenie anteny zewnętrznej.

3.3 Czujniki

Sekcja czujników składa się z trzech czujników, odpowiedzialnych za właściwe określanie położenia rakiety w zewnętrznym układzie współrzędnych. Akcelerometr, żyroskop i magnetometr, odpowiadają za określenie orientacji rakiety w przestrzeni. Natomiast na podstawie zmian ciśnienia określanych z wykorzystaniem barometru, możliwe jest wyznaczenie prędkości i wysokości, na jakiej znajduje się rakieta.

Całą topologię połączeń pomiędzy czujnikami, a mikrokontrolerem, można zaobserować na rysunku 3.5. Komunikacja z barometrem i akcelerometrem odbywa się z wykorzystaniem interfejsu SPI. Dane z magnetometru są odczytywane z użyciem interfejsu I^2C, po którym komunikuje się on z akcelerometrem, niezależnie od mikrokontrolera [9].



Rysunek 3.5: Schemat połaczeń czujników z mikrokontrolerem

3.3.1 Barometr

Użyty barometr to Bosch BMP384 [19]. Oferuje on dużą dokładność odczytów, do $\pm 50Pa$. Natomiast na zakresie ciśnienia od $900hPa$ do $1110hPa$ producent gwarantuje dokładność $\pm 9Pa$. Komunikacja z nim możliwa jest poprzez jeden z dwóch interfejsów SPI lub I^2C. W projekcie zdecydowano się na wykorzystanie pierwszego z interfejsów, ze względu na możliwość jednoczesnej komunikacji w dwie strony oraz samą prędkość tejże komunikacji. W przypadku SPI prędkość wynosi 10 Mbps, natomiast standardowe zastosowania interfejsu I^2C, oferują prędkość na poziomie 100 kbps. Jest to kluczowe, przy komunikacji z trzema czujnikami jednocześnie, aby zminimalizować czas wykonania pełnej pętli sterującej. Dodatkowym parametrem, który ten czujnik mierzy, oprócz ciśnienia, jest temperatura.

W celu odpowiedniego zarządzania poszczególnymi fazami lotu najwieksze znaczenie ma przeyspieszenie działające na rakietę oraz wysokość, na jakiej się ona znajduje. Sama znajomość wartości przyspieszenia jest niewystarczająca, ponieważ lekki wstrząs podczas przygotowywania rakiety do startu, mógłby doprowadzić do odpalenia ładunków pirotechnicznych. Dlatego przyspieszenie jest korelowane z wysokością względną. Znając aktualną wartość ciśnienia podczas lotu, jesteśmy w stanie określić, na jakiej wysokości znajduje się obecnie rakieta.

$$h = \frac{T_0}{L}\left(1 - (\frac{P}{P_0})^M\right) \tag{3.1}$$

Wyznaczanie aktualnej wysokości realizowane jest z pomocą (3.1) [3]. Wartość L oznacza gradient temperatury w atmosferze i jest to stała wynosząca $0,0065\frac{K}{m}$, M opisuje zmianę ciśnienia w funkcji wysokości i w przyjętym modelu wynosi $0,1903$. Wartości T_0 oraz P_0 opisują odpowiednio temperaturę i ciśnienie początkowe, czyli w momencie startu. Natomiast P to aktualne wskazanie barometru, które determinuje aktualną wysokość.

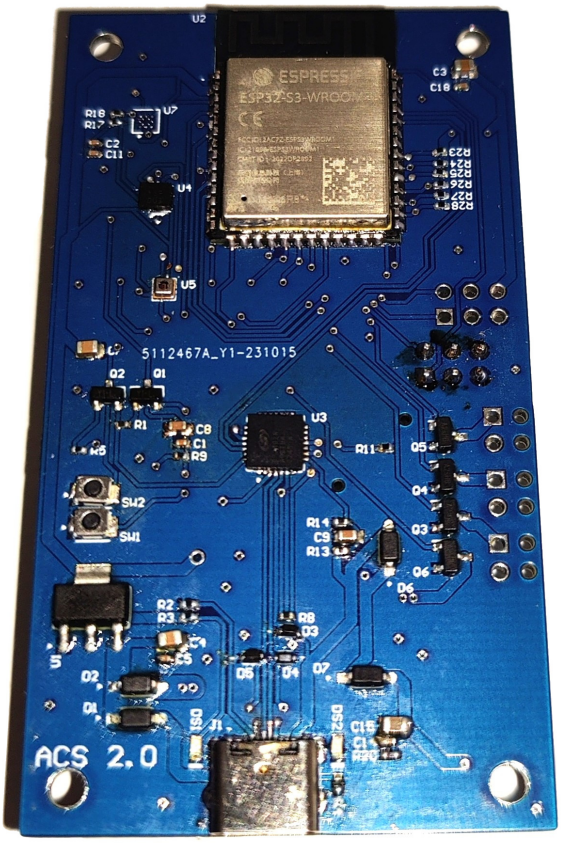
3.3.2 Inercyjna jednostka pomiarowa

Określanie orientacji rakiety, zostało oparte o dane pochodzące z trzech czujników: akcelerometru, żyroskopu i magnetometru. Pierwszy z nich mierzy przyspieszenie liniowe, co umożliwia określenie kierunku oraz siły działającej na rakietę. Żyroskop mierzy prędkość kątową, co pozwala na określenie orientacji układu współrzędnych rakiety w układzie globalnym. Natomiast magnetometr, mierzy natężenie pola magnetycznego, co pomaga w określeniu orientacji układu współrzędnych ciała w globalnym układzie współrzędnych.

Za akcelerometr i żyroskop odpowiada jeden układ scalony Bosch BMI270 [1]. Umożliwia on odczytywanie przyspieszeń w zakresie pomiarowym $\pm 16g$. Prędkość kątowa może być mierzona w zakresie $\pm 2000\frac{^\circ}{s}$. Dodatkowo układ oferuje możliwość podłączenia do niego zewnętrznych czujników na podstawie interfejsu I^2C. Pozwala to na jednoczesny odczyt danych z wbudowanych czujników oraz z zewnętrznych czujników, które są kompatybilne z interfejsem AUX. Producent zaimplementował kolejkę, pozwalającą przechowywać w pamięci układu, do 2kB danych. Możliwe jest również włączenie filtru dolnoprzepustowego, który działa na samym układzie. Taka opcja może przyspieszyć pętle sterowania, dzięki zmniejszeniu liczby operacji potrzebnych do wykonania na mikrokontrolerze.

Taki sposób podłączenia został wykorzystany przy połączeniu z czujnikiem Bosch BMM150, który odpowiada za odczyt pola magnetycznego[18]. Podobnie jak akcelerometr wskazuje kierunek pola grawitacyjnego Ziemi, magnetometr wskazuje na kierunek magnetyczny. Ponieważ w momencie działania na rakietę sił zewnętrznych takich, jak

opór powietrza, czy ciąg silnika, akcelerometr stwarza pewne problemy z prawidłowym wyznaczaniem orientacji. Dlatego przy filtrach takich, jak filtr Madgwicka, integruje się dane z trzech czujników [14]. Dopiero takie połączenie pozwala na uzyskanie kompleksowej informacji o ruchu i orientacji obiektu.

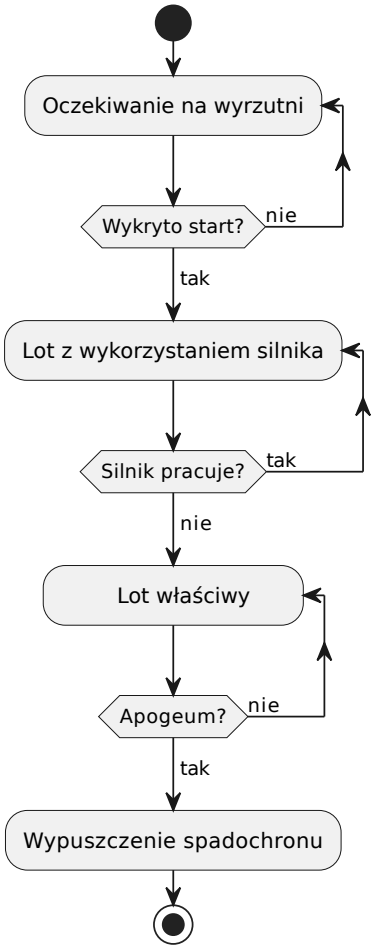


Rysunek 3.6: Gotowy moduł elektroniki

Rozdział 4

Oprogramowanie

Oprogramowanie, które umożliwiłoby zaimplementowanie systemu w rzeczywistym rozwiązaniu, powinno być zrealizowane z wykorzystaniem maszyny stanów. Jest to koncept, który znajduje doskonałe zastosowanie w tworzeniu kodu do rakiet, kiedy jest pożądane, aby pętla programu, wykonywała różne zadania w różnych fazach lotu. Przykład pętli działającej na podstawie maszyny stanów, można zaobserwować na rysunku 4.1.



Rysunek 4.1: Diagram aktywności dla maszyny stanów

Przejście pomiędzy poszczególnymi stanami następuje tylko w momencie spełnienia odpowiednich warunków. Przejścia są jednokierunkowe, czyli nie można przejść z fazy lotu

właściwego, do faz oczekiwania na wyrzutni. Właściwą implementacją jest wykorzystanie takiej maszyny stanów, która zapewni niezbędne stany, tranzycje oraz tranzycje.

Włączenie systemu powoduje oczekiwanie na wyrzutni, które w nieskończonej pętli, sprawdza, czy rakieta rozpoczęła lot. Następuje to po wykryciu przeciążenia wynikającego z uruchomienia silnika. Drugim warunkiem logicznym, który zabezpiecza program, przed niepożądanym przełączeniem pomiędzy stanami, jest zmiana wysokości, wykrywana poprzez barometr.

Lot na silniku trwa do momentu, gdy przestanie działać na rakietę przeciążenie. Następuje wtedy przejście do stanu lotu właściwego, gdzie odbywa się stabilizacja w jednej z osi. Pętla dokonuje równoczesnego odczytu z czujników, sterowania na podstawie odczytów z żyroskopu oraz zapisu telemetrii na kartę pamięci i do pamięci Flash.

Równocześnie sprawdzany jest warunek osiągnięcia apogeum. Mogłoby to zostać zrealizowane z wykorzystaniem akcelerometru, który w najwyższym punkcie lotu, powinien przez krótki moment wskazać wartość zbliżoną do stanu nieważkości. Redundancję takiego przejścia, można by dodatkowo zapewnić wykorzystując sprawdzanie aktualnej wysokości z użyciem barometru. Kiedy wysokość zacznie spadać, oznacza to, że rakieta przekroczyła apogeum.

Następuje wtedy włączenie ładunków pirotechnicznych, które odpowiadają za wypuszczenie spadochronu. Równocześnie dezaktywowany jest system sterujący. Tutaj może nastąpić wysyłanie okresowych sygnałów dźwiękowych z generatora piezoelektrycznego, które mogą ułatwić lokalizację rakiety po wylądowaniu.

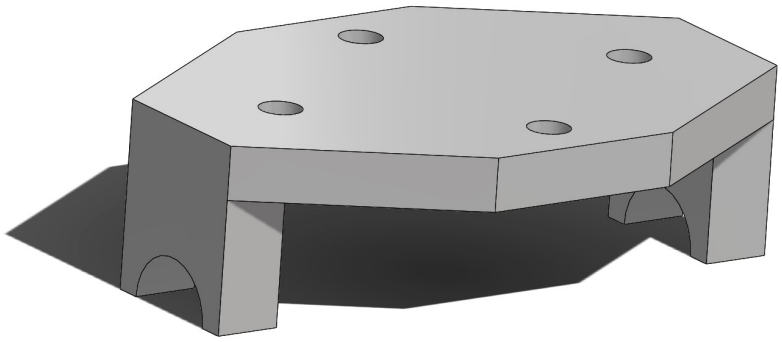
Rozdział 5

Konstrukcja mechaniczna

Konstrukcję rzeczywistego systemu rozpoczęto od zaprojektowania kosza, który byłby w stanie zostać z łatwością osadzony w rakiecie. Głównym ograniczeniem, które zostało nałożone na konstrukcję mechaniczną, było przyjęcie maksymalnej średnicy korpusu $\phi = 5mm$. Jest to standardowa średnica używana w rakietach eksperymentalnych, dla których nie stosuje się silników rakietowych wymagających licencji Polskiego Towarzystwa Rakietowego [16].

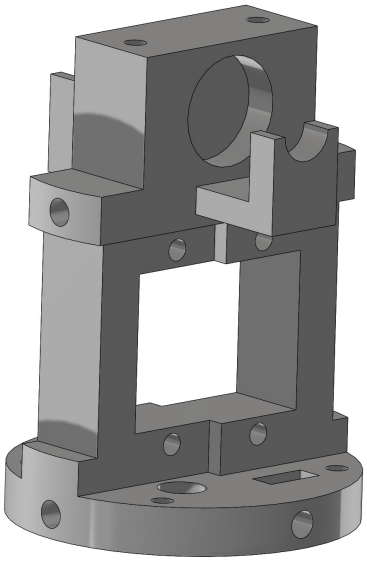
Do generowania momentów siły na lotkach, wykorzystano serwomechanizmy modelarskie. Zdecydowano się na model MG-90S, który jest w stanie wygenerować zadowalający moment $0,2Nm$, jednocześnie mając rozmiar, pozwalający na osadzenie dwóch takich serwomechanizmów w konstrukcji. Są one zasilane napięciem 5V, które pozwala na uzyskanie prędkości obrotowej na poziomie około $\frac{60^\circ}{0,1s}$. Założenie zakresu dopuszczalnego obrotu lotek od -30° do $+30^\circ$, oznacza że system jest w stanie obrócić lotki pomiędzy dwoma skrajnymi położeniami w $0,1s$.

Górę i dół mocowania, przedstawiono na rysunkach 5.1 oraz 5.2. Górna część mocowania posiada 4 otwory pozwalające na połączenie ze sobą obu elementów kosza oraz ramiona, które służą jako podpory i łożyska dla wału lotki.



Rysunek 5.1: Góra mocowania

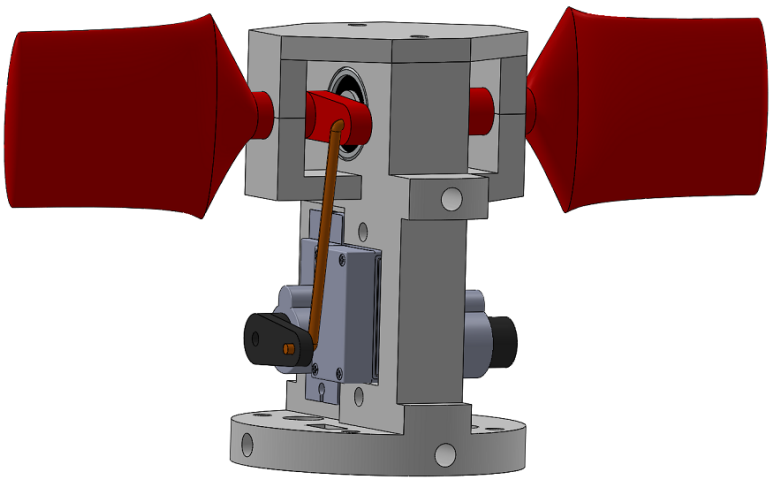
Dolna część kosza posiada 6 otworów, które pozwalają na przytwierdzenie systemu do korpusu rakiety, przy pomocy śrub M3. W górnej części projektu można zauważyć miejsce na przyklejenie łożysk kulkowych, które są stosowane w celu minimalizacji oporów ruchu i stabilizacji lotki. Widoczne jest również identyczne ramię jak w górnej części kosza, które stanowi dodatkowy punkt podparcia i stabilizacji dla wału lotki.



Rysunek 5.2: Dół mocowania

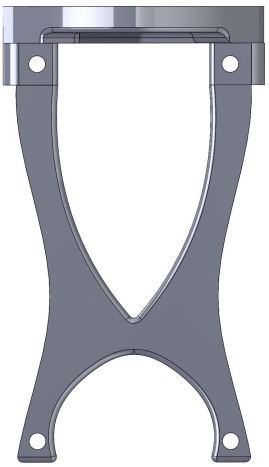
Poniżej mocowania lotek, znalazło się miejsce na zamocowanie dwóch serwomechanizmów MG-90S. Złożenie kosza wraz z zamocowanymi lotkami, orczykami i cięgnami widoczne jest na rysunku 5.3. Takie złożenie elementów mechanicznych powoduje bezpośrednie przeniesienie napędu z serwomechanizmu na wał lotki. Szeregowe ustawienie serwomechanizmów znacząco zmniejszyło rozmiar całego kosza. Na samym dole konstrukcji, umiejscowiono 4 otwory, które pozwalają na przymocowanie dowolnej wersji elektroniki tak, aby cały system był prosty w modyfikacji i przystosowany do rozwoju w przyszłości.

Obok otworów na śruby, znalazły się również 4 wypusty na przewody, które pozwalają na proste przeprowadzenie przewodów zasilających i sterujących do serwomechanizmów. Dzięki przepustom na przewody, możliwe jest również łatwe połączenie z ewentualną dodatkową elektroniką, znajdującą się w konstrukcji ponad koszem mocującym system. Przykładem takiej dodatkowej elektroniki, może być na przykład kamerka rejestrująca lot rakiety.



Rysunek 5.3: Złożenie kosza systemu stabilizującego

Rysunek 5.4 prezentuje dodatkowy element całego kosza, którego zadaniem jest połączenie elektroniki z koszem zawierającym serwomechanizmy i lotki. W obecnej implementacji lotki wychylają się zawsze w taki sposób. Obie lotki zawsze są wychylane w tym samym kierunku o ten sam kąt. Jednak możliwe jest wychylanie lotek o kąty przeciwne, czyli gdy jedna z lotek odchylana jest o kąt α, druga wychyla się o kąt $-\alpha$. Efektem takiego ustawienia, jest możliwość sterowania w dwóch osiach.



Rysunek 5.4: Mocowanie

Cała konstrukcja przedstawiona na rysunku 5.5, została wykonana w technologii druku 3D. Kosz oraz elementy mocujące, wykonane zostały z materiału PLA, który został wybrany ze względu na cenę. Natomiast lotki oraz orczyki wydrukowano z materiału ABS, które pomimo wyższej ceny i trudniejszego procesu druku gwarantują większą wytrzymałość, co jest kluczowe przy elementach, na które działają największe siły.

Rysunek 5.5: Rzeczywista konstrukcja

Rozdział 6

Podsumowanie

Celem niniejszej pracy inżynierskiej było stworzenie oraz symulacyjne zbadanie zachowania regulatora PID w sterowaniu orientacją rakiet eksperymentalnych. Przeprowadzenie symulacji, wymagało zamodelowania rakiety w środowiskach *SolidWorks* oraz *OpenRocket*. Pozwoliło to na zdefiniowanie współczynników aerodynamicznych przykładowej rakiety. Same badania były przeprowadzone z wykorzystaniem środowiska *MuJoCo*. Następnie przedstawiono projekt elektroniki i konstrukcji, które mogłyby zostać wykorzystane w celu umieszczenia takiego systemu w rzeczywistej rakiecie. Sama konstrukcja została z powodzeniem wykonana z wykorzystaniem technologii druku 3D. Elektronikę wykonano w technologii montażu powierzchniowego oraz przetestowano wszystkie niezbędne komponenty systemu.

Przeprowadzone badania wykazały, że regulator PID, jest w stanie dobrze ograniczać rotację rakiety, tym samym stabilizując jej lot. Badanie zachowania systemu, z wykorzystaniem śledzenia sygnału, wykazało, że wydajność systemu znacząco spada wraz ze spadkiem prędkości rakiety. Dodatkowe zastosowanie pewnego rodzaju kompensacji, która uwzględniałaby aktualną prędkość rakiety, mogłoby zniwelować takie zachowanie systemu. Efektem takiego rozwiązania byłaby dokładniejsza stabilizacja w przypadku końcowej fazy lotu właściwego, gdy prędkość rakiety znacząco maleje.

Przeprowadzone badania potwierdziły postawioną tezę pracy.

Zaprojektowana elektronika pozwala na określanie orientacji z wykorzystaniem np. filtru Madgwicka oraz na określenie aktualnej prędkości i wysokości lotu z wykorzystaniem danych pochodzących z czujnika ciśnienia. Dodatkowe umiejscowienie na płytce tranzystorów sterujących pirotechniką od systemu odzysku umożliwia implementację elektroniki jako głównego i jedynego komputera pokładowego w rakiecie. Taka konstrukcja pozwala ograniczyć koszty sprzętu, a także sprawia, że system jest kompaktowy i łatwo naprawialny.

Wykonane mocowanie lotek i serwomechanizmów pozwala na sterowanie lotkami w zakresie $[-30°; 30°]$, a dobrane serwa generują momenty, które są w stanie bez problemu obrócić lotki pomiędzy ich skrajnymi pozycjami w ciągu $0,1s$.

Osiągnięcie stabilizacji w osi równoległej do kierunku lotu rakiety stanowi podstawowy kamień milowy w rozwoju projektu w przyszłości. Następnym krokiem mogłoby być wprowadzenie systemu stabilizacji wieloosiowej. Umożliwiłoby to rakiecie kontrolowanie orientacji nie tylko w osi równoległej, ale również w osiach prostopadłych do kierunku ruchu. Jest to kluczowe, zwłaszcza gdy rakieta zmierza do bardziej złożonych misji, takich jak śledzenie założonej trajektorii czy sterowanie do punktu.

Rozwinięcie rakiety o stabilizację wieloosiową, po uprzednim zapewnieniu podstawowej stabilności przy użyciu przedstawionego modelu, pozwala na osiągnięcie zaawansowanych zdolności kontroli, co jednak wymagałoby bardziej złożonych algorytmów sterowania.

Literatura

[1] BMI270 6-axis, smart, low power Inertial Measurement Unit for high-performance applications, 2023.

[2] Z. Aytaç, F. Aktaş. Utilization of CFD for the aerodynamic analysis of a subsonic rocket. *Politeknik Dergisi*, 2020.

[3] W. G. Brombacher. Altitude by measurement of air pressure and temperature. *Journal of the Washington Academy of Sciences*, 34(9):277–299, 1944.

[4] P. Constantin, C. Foias. *Navier-stokes equations*. University of Chicago press, 1988.

[5] DeepMind Technologies Limited. API Reference - MuJoCo Documentation. `https://mujoco.readthedocs.io/en/stable/APIreference/index.html`, 2023.

[6] Diodes Incorporated. 1.0a surface mount schottky barrier rectifier, 2014.

[7] Espressif Systems. ESP32-S3-WROOM-1 Documentation, 2023.

[8] P. Kuentzmann. Introduction to solid rocket propulsion. *Office National d'Etudes et de Recherches Aérospatiales*, 2002.

[9] F. Leens. An introduction to I2C and SPI protocols. *IEEE Instrumentation & Measurement Magazine*, 12(1):8–13, 2009.

[10] Leshan Radio Company, LTD. Transient Voltage Suppressors for ESD Protection, 2016.

[11] R. Lloyd, G. Thorp. A review of thrust vector control systems for tactical missiles. *Joint Propulsion Conference*, Las Vegas, Stany Zjednoczone, 1978.

[12] M. Lombard. *SolidWorks 2013 bible*. John Wiley & Sons, 2013.

[13] J. Lv, X. Li, T. Yang, H. Yu, B. Liu. A general pseudo-random number generator based on chaos. *EAI International conference on robotic sensor networks*, strony 103–109, Online, 2022. Springer.

[14] S. O. Madgwick, A. J. Harrison, R. Vaidyanathan. Estimation of imu and marg orientation using a gradient descent algorithm. *2011 IEEE international conference on rehabilitation robotics*, strony 1–7. IEEE, 2011.

[15] S. Niskanen. Openrocket technical documentation. *Development of an Open Source model rocket simulation software*, 2013.

[16] Polskie Towarzystwo Rakietowe. Statut polskiego towarzystwa rakietowego, 2019.

[17] N. Sahbon, S. Murpani, M. Michałów, D. Miedziński, M. Sochacki. A CFD study of the aerodynamic characteristics of Twardowsky and FOK rockets. *Transactions on Aerospace Research*, 2022(1):35–58, 2022.

[18] B. Sensortec. BMM150 Geomagnetic Sensor, 2020.

[19] B. Sensortec. BMP384 Digital pressure sensor, 2020.

[20] S. Theodoulis, V. Gassmann, P. Wernert, L. Dritsas, I. Kitsios, A. Tzes. Guidance and control design for a class of spin-stabilized fin-controlled projectiles. *Journal of Guidance, Control, and Dynamics*, 36(2):517–531, 2013.

[21] E. Todorov, T. Erez, Y. Tassa. Mujoco: A physics engine for model-based control. *IEEE/RSJ international conference on intelligent robots and systems*, strony 5026–5033, Kioto, Japonia, 2012.

Załącznik A

Do pracy załączono płytę DVD zawierającą w poszczególnych katalogach:

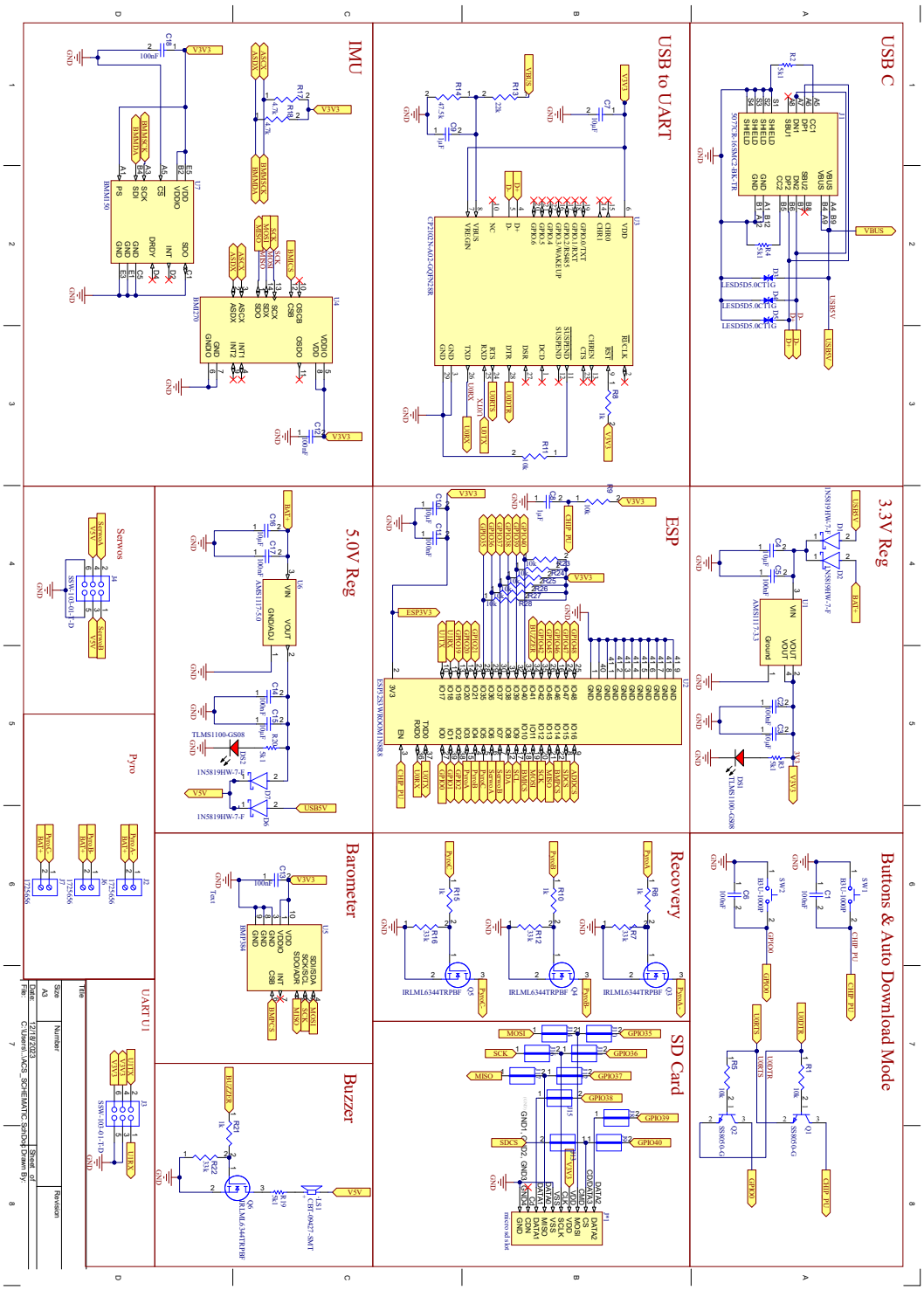
`/Praca_inzynierska.pdf` — wersja cyfrowa pracy,

`/Kod_zrodlowy` — kod źródłowy oprogramowania wykorzystanego do symulacji,

`/PCB.zip` – projekt płytki PCB,

`/Czesci_mechaniczne.zip` – archiwum z częściami mechanicznymi wykorzystanymi w projekcie.

Załącznik B



Rysunek 1: Schemat elektroniczny