POLITECHNIKA WROCŁAWSKA
WYDZIAŁ ELEKTRONIKI

KIERUNEK: Automatyka i Robotyka (AIR)
SPECJALNOŚĆ: Technologie Informacyjne w Systemach Automatyki (ART)

PRACA DYPLOMOWA INŻYNIERSKA

System lokalizacji samolot'ow z wykorzystaniem ADS-B

Airplane tracking system using ADS-B

AUTOR:
Karol Szpila

PROWADZĄCY PRACĘ:

dr inż. Krzysztof Halawa

Katedra Informatyki Technicznej

OCENA PRACY:

WROCŁAW 2017

Spis treści

Rozdział 1

Wstęp

ADS-B (ang. Automatic Dependent Surveillance–Broadcast) to system służący do śledzenia statków powietrznych wykorzystywany w kontroli ruchu powietrznego. Powstał, aby uzupełniać pracę PSR (ang. Primary Surveilance Radar) i w przyszłości całkowicie go zastąpić. PSR to radar aktywny bazujący na wysyłaniu fal elektromagnetycznych oraz pomiarze czasu powrotu fali odbitej od obiektu. Wadą takich systemów jest brak informacji o wykrytym obiekcie poza jego lokalizacją i rozmiarem, wrażliwość na ukształtowanie terenu oraz warunki pogodowe. ADS-B bazuje na lokalizacji przy pomocy satelit GPS. Statki powietrzne w sposób ciągły ustalają swoją pozycję oraz nadają drogą radiową swoje położenie, prędkość oraz identyfikator.

Mode-S to system SSR (ang. Secondary Surveillance Radar) wykorzystywania do odpytywania statku powietrznego o danym 24-bitowym adresie ICAO (ang International Civil Aviation Organization). SSR to wtórny radar dozorowania, uzupełniający prace PSR o dodatkowe informacje odebrane od statku powietrznego. Zapytania wysyłane są przez ATM (ang. Air Traffic Management) na częstotliwości 1030Mhz, natomiast odpowiedzi na 1090Mhz.ADS-B wykorzystuje Mode-S jako technologię do przesyłania informacji.

Tabela 1.1: *Rodzaje wiadomości ADS-B*

Rodzaj wiadomości	Downlink Format	Zawartość
ADS-B (56 bitów)	DF0	Odpowiedz Short Air to Air ACAS
	DF4	Poziom lotu
	DF5	Identyfikator (Roll-call)
	DF11	Mode-S odpowiedz All-call
ADS-B (112 bitów)	DF16	Odpowiedz Long Air to Air ACAS
	DF17	Pozycja powietrzna pozycja lądowa status ID i rodzaj samolotu prędkość powietrzna
Mode-S EHS (112 bitów)	DF20	Poziom lotu oraz (BDS 4.0/5.0/6.0)
	DF21	ID oraz (BDS 4.0/5.0/6.0)

Powyższa tabela prezentuje podział wiadomości ADS-B ze względu na rozmiar bloku danych oraz DF (ang. Downlink Format), czyli format odebranej ramki. Wiadomosci można podzielić według długości bloku danych na normalne o długości 56 bitów i rozszerzone 112 bitowe.DF0 i DF16 wykorzystywane są w ACAS (ang. Airborne Collision Avoidance System).Wyróżniamy odpowiedz Short air to air ACAS (DF0), odbierane przez stacje ATM, oraz Long air to air ACAS (DF16), stosowaną do informowania poszczególnych statków o możliwości kolizji. Statki wysyłające ramkę DF5 potwierdzają, że są wyposarzone w transponder ADS-B. Ponadto, można wyróżnić odpowiedzi Mode-S EHS (ang. Enhanced Surveillance) DF20 i DF21, uzyskiwanie na zapytanie kontroli naziemnej, które zawierają dodatkowe informacja niedostępne w Mode-S ADS-B, w zależności od wysłanego przez kontrolę naziemną BDS (ang. Comm-B Data Selector). BDS to numer określający żądaną w odpowiedzi informację. Tylko SSR który wysłał zapytanie Mode-S EHS jest wstanie zdekodować otrzymaną odpowiedź DF20 i DF21, ponieważ nie zawiera on wysłanego BDS. Poniżej przedstawiono tabelę z wymienionymi wcześniej BDS.

Tabela 1.2: *BDS dla wiadomości DF20 i DF21*

BDS Register	**Basic DAP Set** (if Track Angle Rate is available)	**Alternative DAP Set** (if Track Angle Rate is **not** available)
BDS 4,0	Selected Altitude	Selected Altitude
BDS 5,0	Roll Angle	Roll Angle
	Track Angle Rate	
	True Track Angle	True Track Angle
	Ground Speed	Ground Speed
BDS 6,0	Magnetic Heading	Magnetic Heading
	Indicated Airspeed (IAS) / Mach no. (Note: IAS and Mach no. are considered as 1 DAP (even if technically they are 2 separate ARINC labels). If the aircraft can provide both, it must do so).	Indicated Airspeed (IAS) / Mach no. (Note: IAS and Mach no. are considered as 1 DAP (even if technically they are 2 separate ARINC labels). If the aircraft can provide both, it must do so).
	Vertical Rate (Barometric rate of climb/descend or baro-inertial)	Vertical Rate (Barometric rate of climb/descend or baro-inertial)
		True Airspeed (provided if Track Angle Rate is not available)

Wiadomość ADS-B i Mode-S EHS można odbierać przy pomocy dowolnego odbiornika dostrojonego do częstotliwości 1090MHz. Tunery z układem RTL2832U można łatwo przekształcić w SDR (ang. Software Defined Radio) SDR to system komunikacji radiowej w którym strojenie odbywa się poprzez pogram bez ingerencji w sprzęt. Wspomniany odbiornik jest znacznie tańszy od profesjonalnych rozwiązań, co spopularyzowało ADS-B w zastosowaniach amatorskich.

Rozdział 2

Cele i założenia projektowe

Celem niniejszej pracy jest stworzenie prototypu systemu pozwalającego na lokalizację oraz zbieranie informacji o statkach powietrznych wyposażonych w transpondery ADS-B. W systemie można wyróżnić dwie zasadnicze części: Komunikacji radiowej, odpowiedzialna za wykrywanie, zbieranie i dekodowanie wiadomości oraz prezentacji danych, której celem jest przedstawienie informacji w przystępny sposób poprzez interfejs graficzny. Urządzenie można wykorzystać jako alternatywa dla drogich i profesjonalny urządzeń w przypadku zastosowań amatorskich.

2.1 Środowisko sprzętowe i wykorzystane narzędzia

W tym podrozdziale zostaną przedstawiono wykorzystanie w systemie urządzenia oraz środowisko programistyczne.
Jako odbiornik radiowy zostanie wykorzystany tuner DVB-T z układem RTL2832U wyposażony w interfejs USB. Urządzenie można łatwo zamienić w SDR. Zdecydowano się na wykorzystanie mikrokontrolera firmy ST z serii STM32F767, ze względu na posiadanie peryferiów niezbędnych do obłożenia urządzeń w systemie, oraz dostępność narzędzi i oprogramowania. W projekcie wykorzystano wyświetlacz LCD-TFT o przekątnej 10.1 cala. Poniżej wymieniono użyte biblioteki.

- librtl-sdr - obsługa Tunera DVB-T jako SDR.
- STM32 USB host library - komunikacja mikrokontolera z radiem porzez interfejs USB.
- STemWin - obsługa interfejsu graficznego.
- dump1090 - dekodowanie wiadomości ADS-B

Zdecydowano się na wykorzystanie systemu operacyjnego FreeRTOS, w celu zapewnienia wielowątkowości oraz mechanizmów do zarządzania zasobami sprzętowymi. Do programowania urządzenia użyto programatora ST-Link z zestawu uruchomieniowego STM32F429I-DISC1. Poniżej przedstawiono programy wykorzystanie przy rewitalizacji projektu.

- Atollic True Studio - środowisko do programowania mikrokontrolerów.
- STM32CubeMx - generowania kodu konfiguracyjnego dla układów STM32.
- Circuit Maker - wykonanie projektu PCB.

Rozdział 3

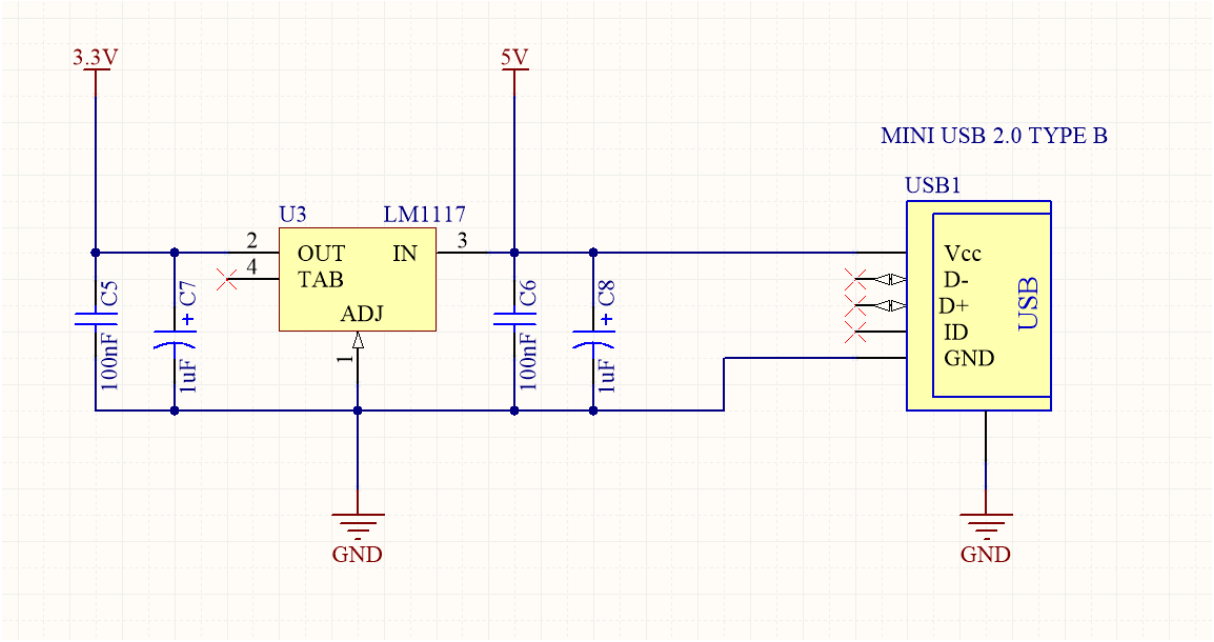
Architektura sprzętowa

3.1 Schemat Urządzenia

W tym rozdziale zostanie opisana część sprzętowa projektu, czyli schemat urządzenia oraz projekt PCB, wykorzystane elementy oraz zewnętrzne urządzenia.

3.1.1 Zasilanie

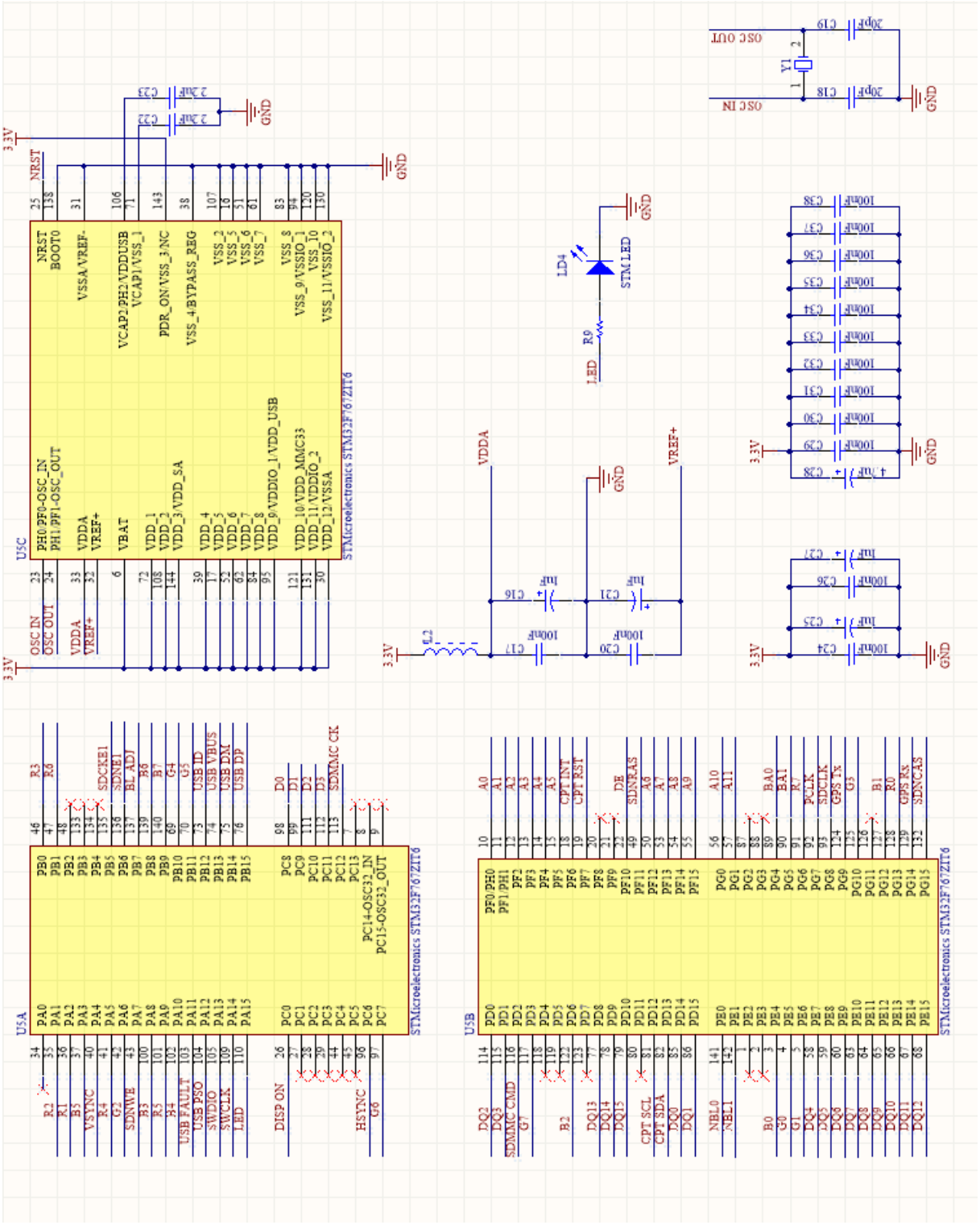
Urządzenie jest zasilanie z zewnętrznego źródła 5V poprzez port USB typu mini A. Pozwala to na podłącznie PCB zarówno do sieci przy pomocy ładowarki do telefonu jak i z komputera czy z przenośnego power banku. Ponieważ układy takie jak mikrokontroler czy pamieć SDRAM potrzebują napięcia 3.3V zastosowano stabilizator LM1117 Poniżej przedstawiono schemat podłączania.



Rysunek 3.1: *Schemat zasilania płZytki PCB*

3.1.2 Mikrokontroler

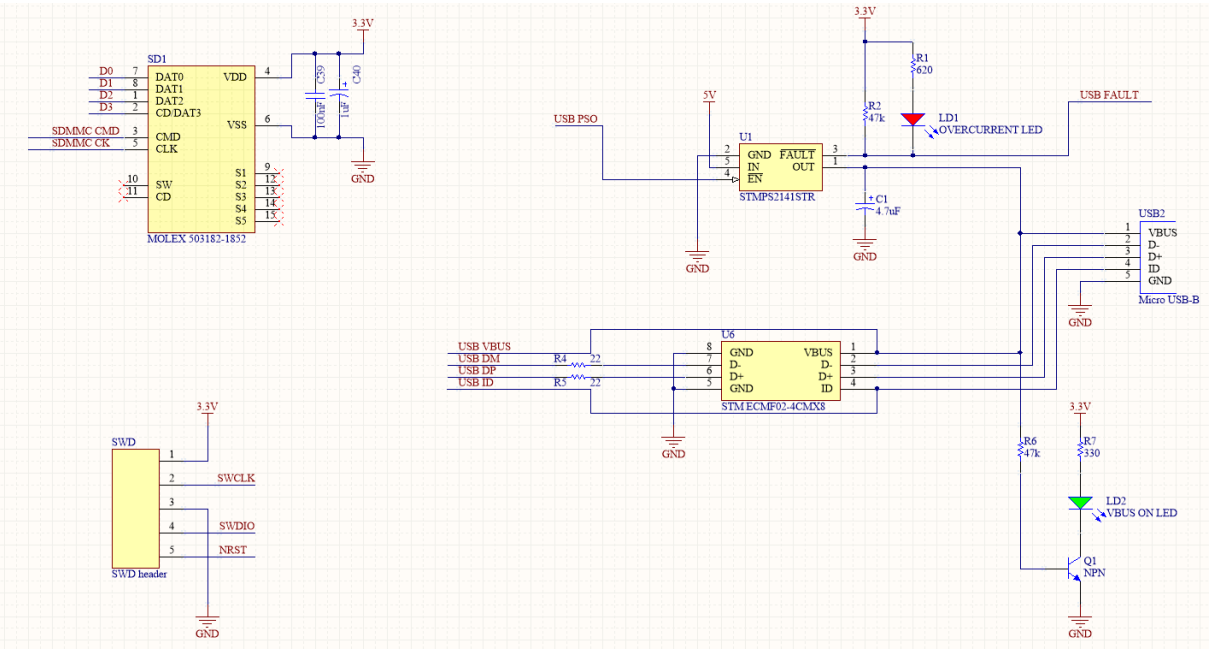
Wykorzystany mikrokontroler to STM32F767ZIT6. Układ został wybrany ze względu na zmieszczenie w najmniejszej obudowie wszystkich wymaganych układów peryferiów takich jak: kontroler LDTC do sprzętowej obsługi wyświetlacza LCD, kontroler pozwalający obsłużyć zewnętrzną pamięcią SDRAM i interfejsy komunikacyjne USB, I2C i UART. Poniżej przedstawiono schemat podłączenia.



Rysunek 3.2: *Schemat podłączenia mikrokontrolera STM32f767ZIT6*

3.1.3 Interfejsy komunikacyjne

System posiada gniazdo na kartę Micro SD, która może zostać wykorzysta do przechowywania skompresowanych obrazów do tła GUI lub co planowane jest w przyszłości map pozwalających lepiej orientować się w terenie na podstawie obrazu z radaru. Do programowania mikrokontrolera wykorzystany został dedykowany dwuprzeowodowy interfejs SWD zgodny z ST-Link. Ostatni zostanie omówiony interfejs USB do komunikacji z SDR (eng Software Defined Radio). Interfejs został wyprowadzony poprzez gniazdo Micro USB-B pozwalając w ten sposób zaoszczędzić miejsca na PCB. System jako Host USB będzie zasilać podłączony układy których pobór nie powinien przekroczyć, zgodnie ze stangardem USB2.0 500mA ,dlatego zastosowano power switch STMPS2141STR. W przypadku podania stanu wysokiego na pin USB PSO switch zostanie właczony. Jeżeli nie występują żadne sytuacje nieporządne takie jak przetężenie prądowe czy zwarcie, zapali się zielona dioda sygnalizująca poprawnie działanie zasilania. W przypadku jakichkolwiek problemów prąd zostanie natychmiast odcięty a układ wystawi wysoki stan na pin FAULT informując mikrokontroler i zapalając czerwona diodę. W celu zabezpieczania interfejsu przed nieporządanymi wyładowaniami elektrostatycznymi związanymi z dotykaniem urządzenia czy wkładaniem urządzenia do gniazda zastosowano układ ochrony ESD (ang . Electro Static Discharge) STMECMF02-4CMX8 dedykowany dla USB2.0. Poniżej przedstawiono schemat połacznia na PCB.

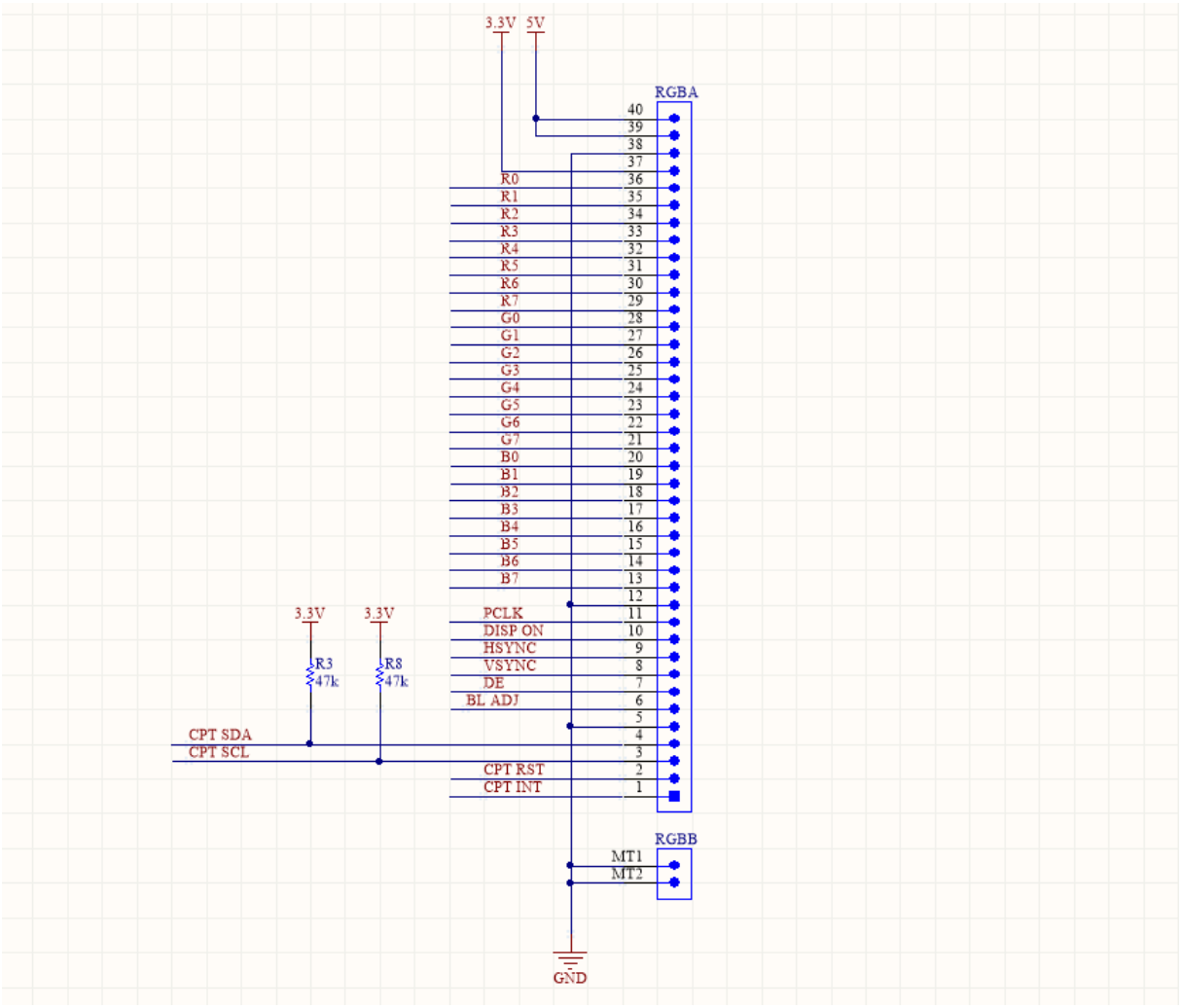


Rysunek 3.3: *Schemat podłączenia zewnętrych interfejsów*

3.1.4 Wyświetlacz

Zgodnie z założeniem, system ma posiadać interfejs graficzny dla użytkownika. Do tego celu wybrano wyświetlacz HY101CTP o przekątnej 10,1" i rozdzielczości 1024 na 600 pikseli. Matryca jest sterowana poprzez MIPI-DPI (ang. Mobile Industry Processor Interface - Display Parell Interface) 24bity na pikesl przy użyciu wbudowanego w układ STM32F7 kontrolara LTDC. Panel dotykowy komunikuje się z mikrokontrolerem poprzez interfejs I2C. Rezystory R3 i R8 mają za zadanie wymusić stan wysoki na liniach, ponieważ

są sterownie w trybie otwartego drenu. W tej konfiguracji można tylko zewrzeć linie do masy zmieniając stan wysoki na niski. Pozwoliło to wyeliminować sytuacje w której linia na jednym końcu jest zwarta do masy a na drugim do zasilania co doprowadziło by do zwarcia i gwałtownego wzrostu mocy mogącego uszkodzić układ.

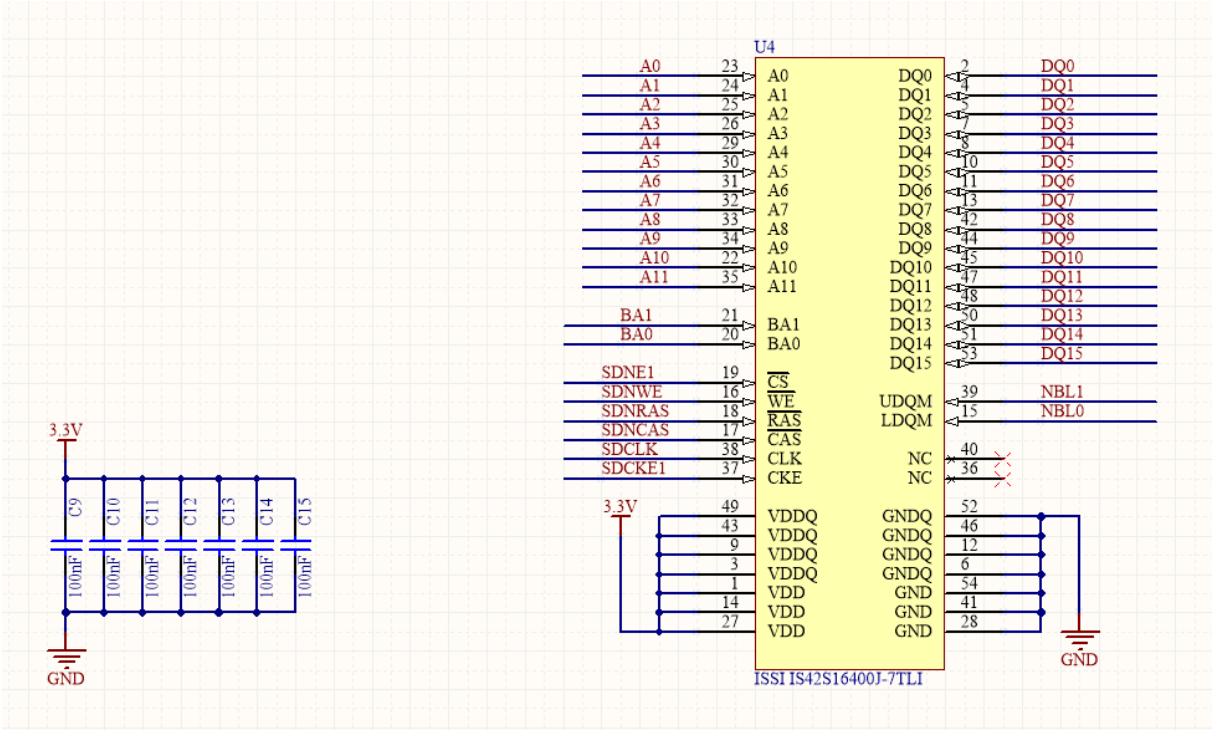


Rysunek 3.4: *Schemat podłączenia wyswietlacza*

3.1.5 SDRAM

Do działania interfejsu graficznego potrzebna jest pamięć do przechowywania ramek wysyłanych do wyświetlacza poprzez interfejs MIPI-DPI. Założono iż warstwy będą przechowywane w pamięci w formacie ARGB8888 (po bajcie na każdy kolor i kanał alfa). Kontroler LTDC jest w stanie sprzętowo mieszać dwie warstwy w wynikową która jest wysyłana do wyświetlacza. Zdecydowano, że potrzeba pamięci wystarczająco dużej, by zmieścić trzy bufory. Pierwsza warstwa byłaby przeznaczony na tło, które jest niezmienne podczas działania urządzenia. Dwie pozostałe warstwy służyłby do naprzemiennej prezentacji zmiennych danych takich jak położenie statku powietrznego. Będzie to implementacja mechanizmu podwójnego buforowania pozwalająca wyeliminować migotanie matrycy podczas modyfikacji bufora aktualnie wyświetlanego. Trzy warstwa o rozmiarze 1024 na 600 piskeli w formacie ARGB8888 zajmą 57600 Kb. W pracy zdecydowano się wykorzystać układ IS42S16400J-7TLI posiadający 65536 Kb pamięci. Poniżej przedstawiono

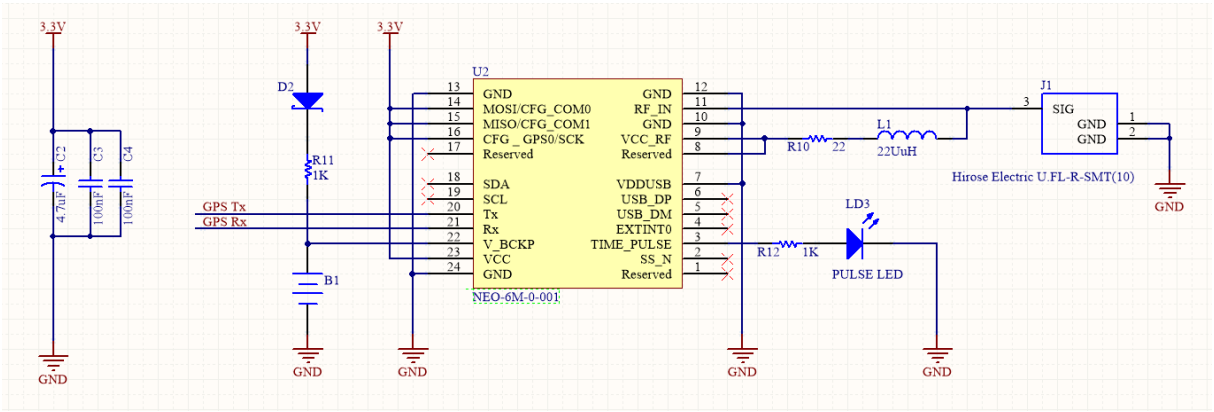
schemat połączenia z mikrokontrolerem.



Rysunek 3.5: *Schemat podłączenia zewnętrzej pamięci SDRAM*

3.1.6 Moduł GPS

Aby system był w stanie popranie obliczyć odległość od namierzonego statu powietrznego i poprawnie zaznaczyć jego pozycję na radarze, potrzebna znać pozycje urządzenia. Do tego zadania wybrano układ NEO-6M-0-001 z zewnętrzną aktywną anteną. Do układu została podłączono zewnętrza bateria podtrzymująca zasilanie. Dzieki temu urządzenie może uruchomić poprzez ciepły start, co pozwala zaoszczędzić czas potrzebny na znalezienie odpowiedniej liczby satelit Do komunikacji z mikrokontrolerem wybrano interfejs UART.



Rysunek 3.6: *Schemat podłączenia modułu GPS*

3.2 Wykonanie PCB

W tym rozdziale szczegółowo opisano projekt oraz parametry wykonanego PCB. Pokazano obliczenia uzasadniające decyzje projektowe oraz ograniczenia wynikające z technologi produkcji i zastosowanych układów.

3.2.1 Parametry PCB

W projekcie PCB zdecydowano się na wykonanie technologia czterowarstwową. Pozwoliło to zmniejszyć rozmiary urządzenia oraz zapewnić dobre ekranowanie dla pomiędzy warstwami sygnałowymi. Ograniczyło to przesłuchy pomiędzy warstwami. Ponadto ciągłość warstw referencyjnych (masy i zasilania) pozwala by prądy powrotne przepływało możliwe najkrótszą drogą o najmniejszej impedancji zmniejszając emisje EMC. Poniżej przedstawiono tabele z układem warstw PCB.

Tabela 3.1: *Kolejność i grubość warst PCB*

Warstwa	grubość [mm]	grubość [mil]
sygnały	0.03556	1,4
dielektryk	0,17018	6,7
masa	0,01778	0,7
rdzeń	1,1938	47
zasilanie	0,01778	0,7
dielektryk	0,17018	6,7
sygnały	0.03556	1,4

Warstwy przewodzące wykonane są z miedzi natomiast jako dielektryk wykorzystano materiał FR-408 o stałej dielektrycznej $\varepsilon_r = 3.66$

3.2.2 GPS

Zgodnie z notą katalogową układu NEO-6M-0-001, ścieka antenowa powinna mieć impedancję dopasowaną do $Z_0 = 50\Omega$, którą obliczono z następującego wzoru :

$$Z_0 = \frac{87}{\sqrt{\varepsilon_r + 1.41}} \ln\left(\frac{5.98H}{0.8W + T}\right) \tag{3.1}$$

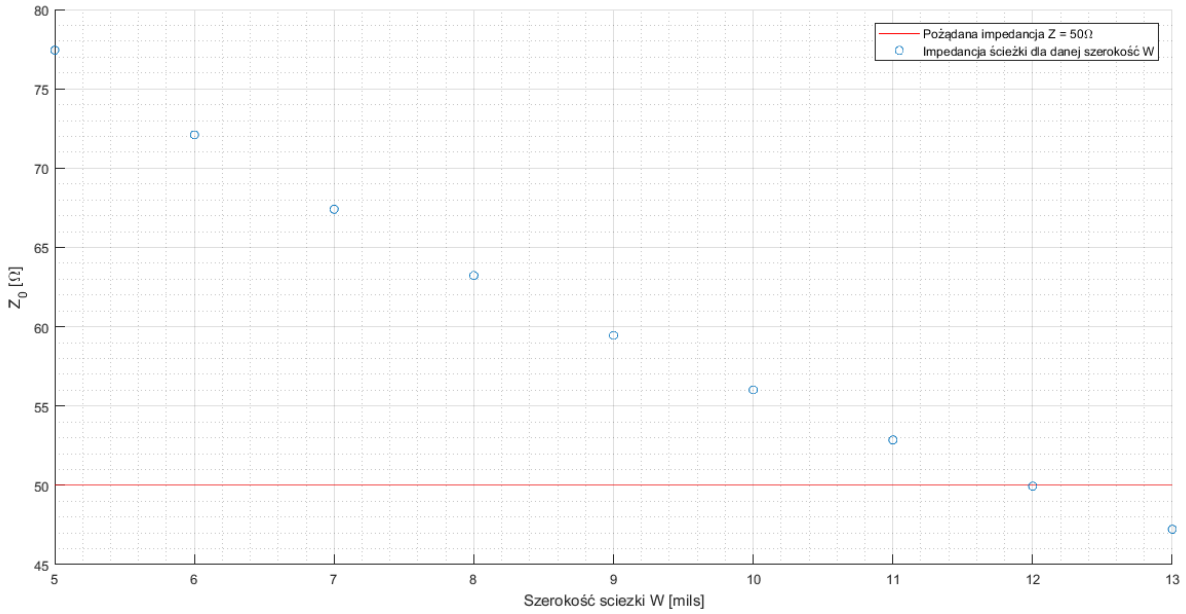
dla

$$0,1 < W/H < 2,0$$

$$1 < \varepsilon_r < 15$$

gdzie

- ε_r - stała dielektryczna
- H - grubość dielektryka
- W - szerokość ścieżki
- T - grubość ścieżki



Rysunek 3.7: *Impedancja ścieżki w zależności od szerokości ścieżki*

Powyższy wykres przedstawia impedancję ścieżki w zależności od szerokości dla anteny modułu GPS obliczoną ze wzoru 3.1 z parametrami $T = 1,4mils$ i $H = 6,7mils$ wziętymi z tabeli 3.1. Czerwoną linią zaznaczono pożądane dopasowanie $Z_0 = 50\Omega$. Założono że W zostanie obliczone z dokładnością do 1 mils. Obliczenia wykonano zaczynając od najmniejszej szerokości gwarantowanej przez producenta $W = 5mils$, aż do punktu pod prostą z optymalnym dopasowaniem. Najlepsze dopasanie otrzymano dla $W = 12mils$, gdzie $Z_0 \approx 49.95$ Dla takiego Z_0 obliczono współczynnik odbicia, który reprezentuje jaka część sygnału została stracona w wyniku niedopasowania impedancji przy przejściu pomiędzy liniami transmisyjnymi.

$$\Gamma = \frac{Z_l - Z_s}{Z_l + Z_s} \tag{3.2}$$

gdzie

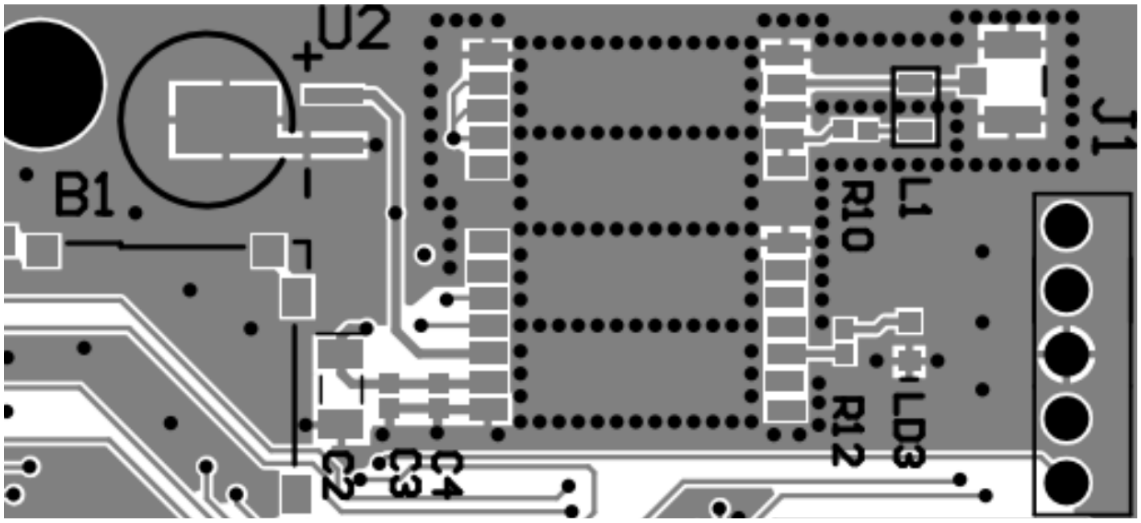
- Z_l - impedancja obciążenia, w tym przypadku impedancja lini transmisyjnej $Z_0 = 49,95\Omega$
- Z_s - impedancja źródła, czyli wejścia antenowego modułu GPS, zgodnie z notą katalogową $Z_s = 50\Omega$

Korzystając ze wzoru 3.2 obliczono:

$$\Gamma = \frac{Z_0 - Z_s}{Z_0 + Z_s} = \frac{49,59\Omega - 50\Omega}{49,59\Omega + 50\Omega} \approx 0.0005 \tag{3.3}$$

W przeliczeniu na procenty daje to $\Gamma \cdot 100\% = 0.0005 \cdot 100\% = 0.05\%$ sygnału odbitego przez linie transmisyjną. Jest to wartość akceptowalna zatem pozostano prze szerokości $W = 12mils$.

Zgodnie z zaleceniami noty katalogowej wykonano serię przelotek pod i do okola układu GPS (U2). Ten sam zabieg zastosowano również dla ścieżki antenowej (pomiędzy J1 i U2). Zapewniło to lepsze odprowadzanie ciepła przez moduł, a zatem niższą temperaturę pracy i mniejsza emisję EMC. Poniżej zaprezentowano zdjęcie z projektem.



Rysunek 3.8: *Projekt PCB dla GPS*

3.2.3 USB

W projekcie wykorzystano wbudowany w mikrokontroler kontroler interfejsu USB2.0 High-Speed o maksymalnej przepustowości 12Mb/s. Zgodnie ze standardem, należy dopasować impedancję różnicową pomiędzy liniami transmisyjnymi sygnału nieodwróconego i odwróconego do 90 Ω. Poniżej przedstawiono zastosowany wzór oraz obliczenia.

$$Z_0 = \frac{174}{\sqrt{\varepsilon_r + 1.41}} \ln\left(\frac{5.98H}{0.8W + T}\right)\left(1 - 0.48e^{(-0.96\frac{D}{H})}\right) \tag{3.4}$$

dla

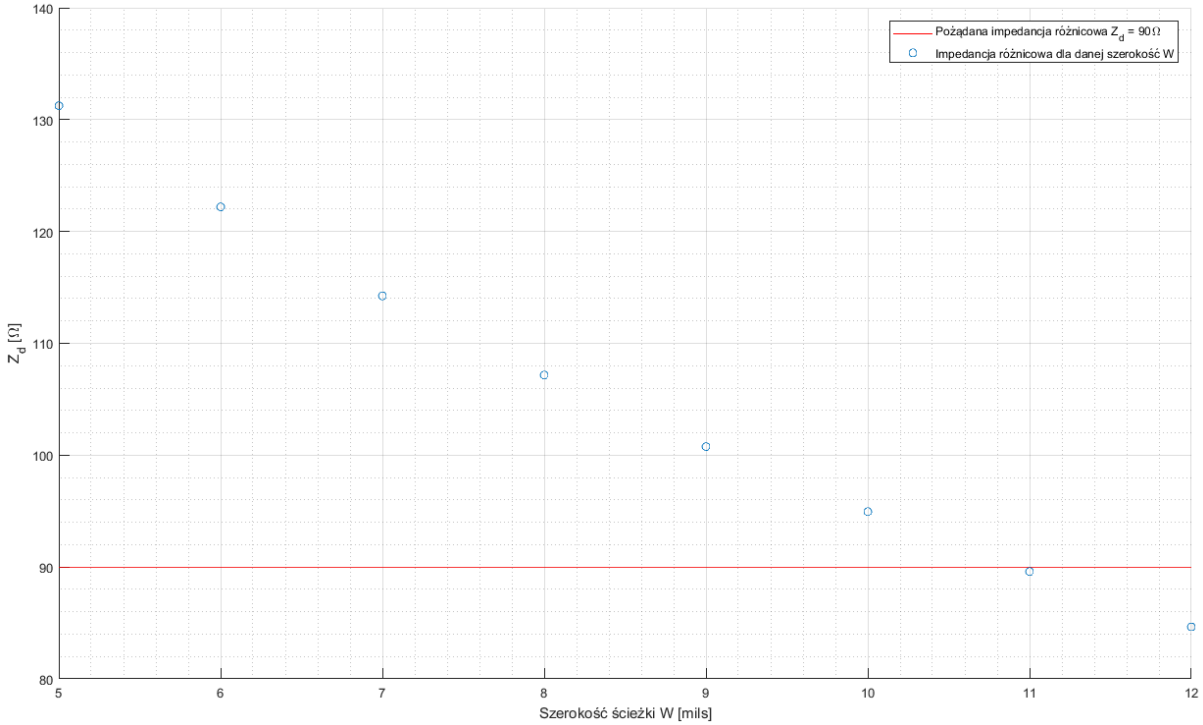
$$0,1 < W/H < 2,0$$

$$1 < \varepsilon_r < 15$$

gdzie

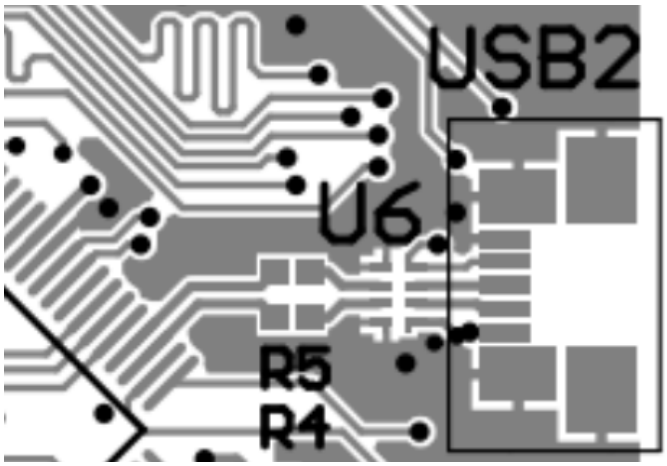
- ε_r - stała dielektryczna
- H - grubość dielektryka
- W - szerokość ścieżki
- T - grubość ścieżki
- D - odległość pomiędzy ścieżkami

Poniższy wykres przedstawia impedancję różnicową lini transmisyjnych interfejsu USB w zależności od szerokości obliczoną ze wzoru 3.4 z parametrami $T = 1,4mils$ i $H = 6,7mils$ wziętymi z tabeli 3.1. Założono odległość pomiędzy ścieżkami $D = 8mils$ równą rozstawowi padów w mikrokontrolerze. Czerwoną linią zaznaczono pożądane dopasowanie $Z_d = 90\Omega$. Założono że W zostanie obliczone z dokładnością do 1 mils. Obliczenia wykonano zaczynając od najmniejszej szerokości gwarantowanej przez producenta $W = 5mils$, aż do punktu pod prostą z optymalnym dopasowaniem. Najlepsze dopasanie otrzymano dla $W = 11mils$, gdzie $Z_d \approx 89.56$. Zgodnie ze specyfikacją interfejsu USB2.0 Full-speed (12 Mb/s) impedancja różnicowa musi wynosić $Z_d = 90 \pm 15\%$. Zatem $76,5\Omega <= Z_d <= 103,5\Omega$. Z_d obliczone dla $W = 11mils$ spełnia to wymaganie.



Rysunek 3.9: *Impedancja różnicowa w zależności od szerokości ścieżki*

Wszystkie elementy zostały umieszczone na górnej warstwie sygnałowej, aby nie stosować przelotek które wprowadzając niepożądane pojemności i indukcyjności linii transmisyjnym. Ścieżki są zaginane po kątem nie większym niż 45°.Warstwą referencyjną dla sygnałów jest warstwa masy co zapewnia lepsze ekranowanie dla sygnałów szybkich. Linie różnicowe są oddzielone polem masy od innych sygnałów o co najmniej 50mils (1.27mm). Zastosowanie powyższych reguł pozwoliło ograniczyć zjawiska utrudniające dopasowania impedancji. Wspominanie w rozdziale schematu rezystory R5 i R4 służą jako szeregowa terminacja sygnału. Dzięki temu zabiegowi czasy narastania zboczy się większe co zmniejsza generowanie zakłócenia EMC. Układ U6 działa jako zabezpieczenie przeciwko wyładowaniom elektrostatycznym mogącym uszkodzić urządzenie oraz jako dodatkowy filtr EMC.Zgodnie z notą katalogową katalogową układu STM32F767xx ich rezystancja wynosi 22 Ω. Poniżej przedstawiono część projektu PCB z interfejsem USB.



Rysunek 3.10: *Projekt PCB dla interfejsu USB*

3.2.4 Interfejsy szybkie

Częstotliwość sygnału zegarowego FMC wynosi $108Mhz$. W przypadku komunikacji z wyświetlaczem szybkość interfejsu zależy od oczekiwanej liczby klatek na sekundę. Częstotliwość sygnału zegarowego dla interfejsu MIPI-DPI wyraża się wzorem:

$$CLK = W \cdot H \cdot fps \tag{3.5}$$

gdzie:

- W - szerokość matrycy w pikselach
- H - wysokość matrycy w pikselach
- fps - częstotliwość odświeżania ekranu

Dla użytego w projekcie wyświetlacza przy założeniu odświeżania matrycy 60Hz korzystając z równania 3.5 otrzymujemy:

$$CLK = 1024 \cdot 600 \cdot 60Hz = 614400 \cdot 60Hz = 36864000Hz \approx 37MHz$$

Interfejsy o takich częstotliwościach zostały uznane za szybkie, co za tym idzie podjęto dodatkowe działania podczas projektowania ich linii transmisyjnych. Zadbano by warstwy referencyjne pod ścieżkami sygnałowymi były ciągłe, aby zapewnić ekranowanie i ograniczyć przesłuchy od linii po przeciwnej stronie płytki. Ponadto dopasowano długości wszystkich ścieżek w interfejsie, by sygnały przychodziły w tym możliwie podobnym czasie. Wzorując się na projekcie referencyjnym płytki ewaluacyjnej STM32F746G-DISCO zadbano, by różnica długości pomiędzy ścieżkami dla obu interfejsów nie była większa niż $100mils$.Opóźnienie linii transmisyjnej, czyli czas jaki sygnał potrzebuje na pokonanie określonej drogi wyraża się wzorem:

$$t = \frac{l\sqrt{\varepsilon_r}}{c} \tag{3.6}$$

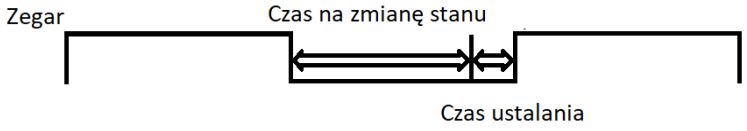
gdzie

- l - długość linii transmisyjnej
- ε_r - stała dielektryczna
- c - szybkość rozchodzenia się fali w próżni

Korzystając ze wzoru 3.6 dla parametrów $l = 0.00254m$ $(100mils)$ i $c = 299\ 792\ 458\frac{m}{s}$ oraz $\varepsilon_r = 3,66$ obliczono opóźnienie sygnału czyli maksymalny odstęp czasu w jakim dzieli sygnały w liniach o długości różniącej się o $100mils$.

$$t = \frac{0.00254m\sqrt{3,66}}{299792458\frac{m}{s}} = \frac{0.00485931m}{299792458\frac{m}{s}} = 1.621 \cdot 10^{-}11 = 16,21ps$$

Wyświetlacz jest sterowny bezpośrednio poprzez interfejs LVDS (ang. Low Voltage Differential Signaling), jednak posiada również wbudowany konwerter THC63LVDM83D pozwalający na komunikację ze pomocą MIPI-DPI. Oznacza to, że wszelkie ograniczenia czasowe powinny być rozpatrywane względem wspomnianego wcześniej układu. Wyjścia mikrokontrolera zmieniają się przy zboczu opadającym sygnału zegarowego, natomiast



Rysunek 3.11: *Diagram ograniczeń czasowych*

są zatrzaskiwane przez konwerter przy zboczu narastającym. Wszystkie sygnały powinny dojść do wyświetlacza od momentu wystąpieniem zbocza opadającego do narastającego z uwzględnieniem czasu ustalenia się sygnału (ang. Setup Time). Jest to najpóźniejszy moment w którym muszą ustalić się stany na wszystkich liniach przed przyjściem sygnału taktującego. Poniżej przedstawiono diagram ilustrujący opisane ograniczenia czasowe.

Zatem maksymalna rozbieżność czasowa pomiędzy sygnałami wyraża się wzorem.

$$t_{max} = t_{low} - t_{setup} \tag{3.7}$$

gdzie

- t_{low} - czas trwania stanu niskiego dla sygnału zegarowego
- t_{setup} - czas ustalania dla sygnału

Z noty katalogowej mikrokontrolera odczytano że stan niski dla sygnał zegarowego LTDC wynosi minimalnie 45% okresu sygnału zegarowego obliczonego z równania 3.5zatem:

$$t_{low} = \frac{1}{36864000Mhz} \cdot 45\% = 27127ps \cdot 45\% = 12207ps$$

W nocie THC63LVDM83D znaleziono $t_{setup} = 2000ps$. Korzystając z 3.7 obliczono

$$t_{max} = 4167ps - 2000ps = 2167ps$$

Ograniczenia czasowe nie zostały przekroczone, ponieważ $t_{max} = 12207ps >= t = 16,21$.

W przypadku FMC wykorzystywanego do obsługi SDRAM, komunikacja odbywa się w obie strony. Ponownie czas trwania stanu niskiego sygnału zegarowego wynosi 45% okresu. Zatem:

$$t_{low} = \frac{1}{108Mhz} \cdot 45\% = 9259ps \cdot 45\% = 4167ps$$

Poniżej przedstawiono tabelę z czasami ustalania dla wszystkich sygnałów interfejsu odczytanych z noty układu IS42S16400J-7TLI.

Tabela 3.2: *Czasy ustalania dla wszystkich sygnałów układu SDRAM*

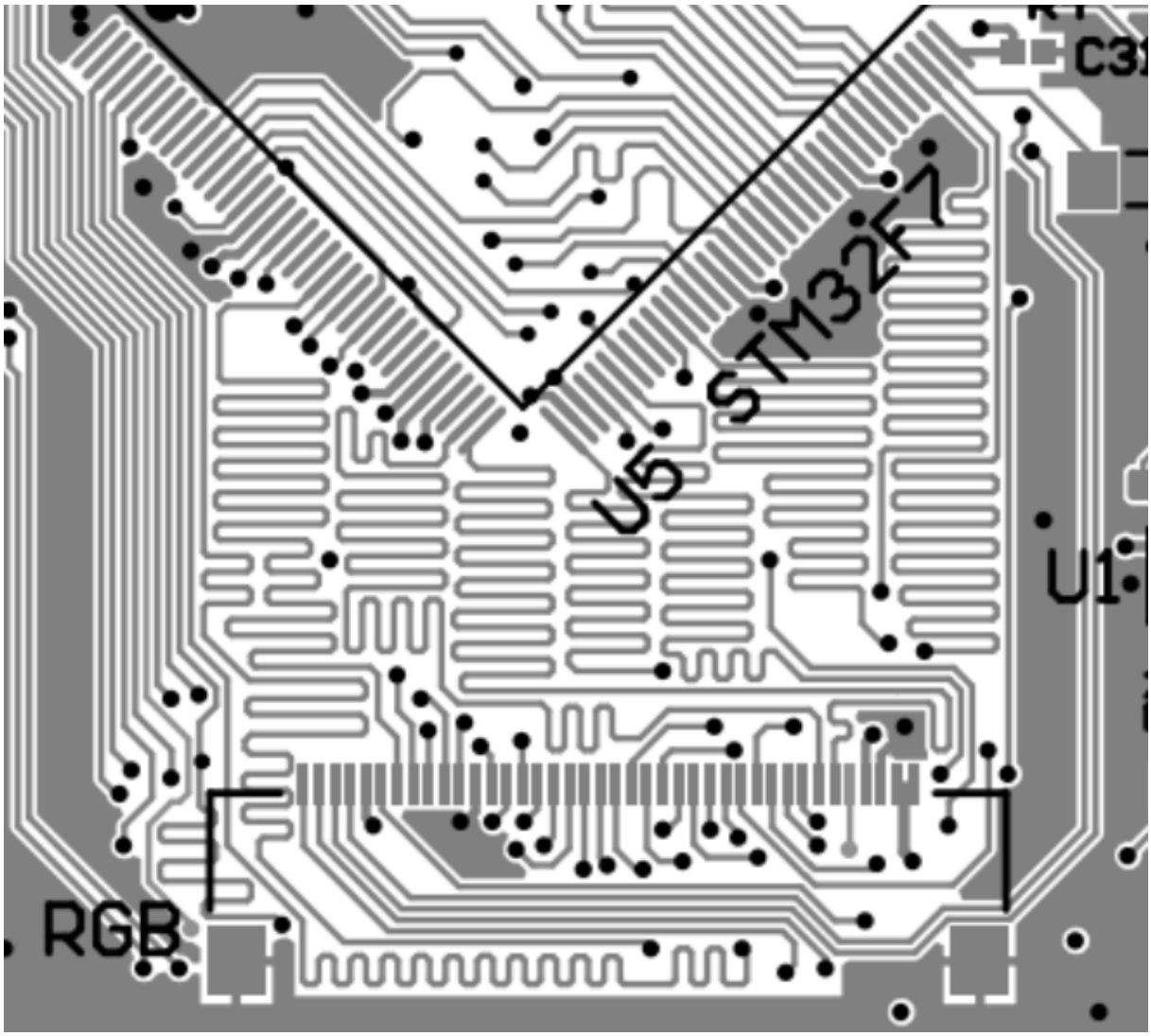
Parametr	czas [ps]
Input Data Setup Time	1500
Address Setup Time	1500
CKE Setup Time	1500
Command Setup Time	1500

Oznacza to, że można przyjąć $t_{setup} = 1500ps$. Korzystając z równania 3.7 obliczono:

$$t_{max} = 4167ps - 1500ps = 2667ps$$

Ponownie $t_{max} = 2667ps >= t = 16,21$, zatem ograniczenia czasowe nie zostały przekroczone.

Poniżej przedstawiono zbliżenie na projekt PCB, gdzie można zauważyć meandry na ścieżkach,których celem jest wyrównanie różnicy długości pomiędzy ścieżkami sygnałowymi.



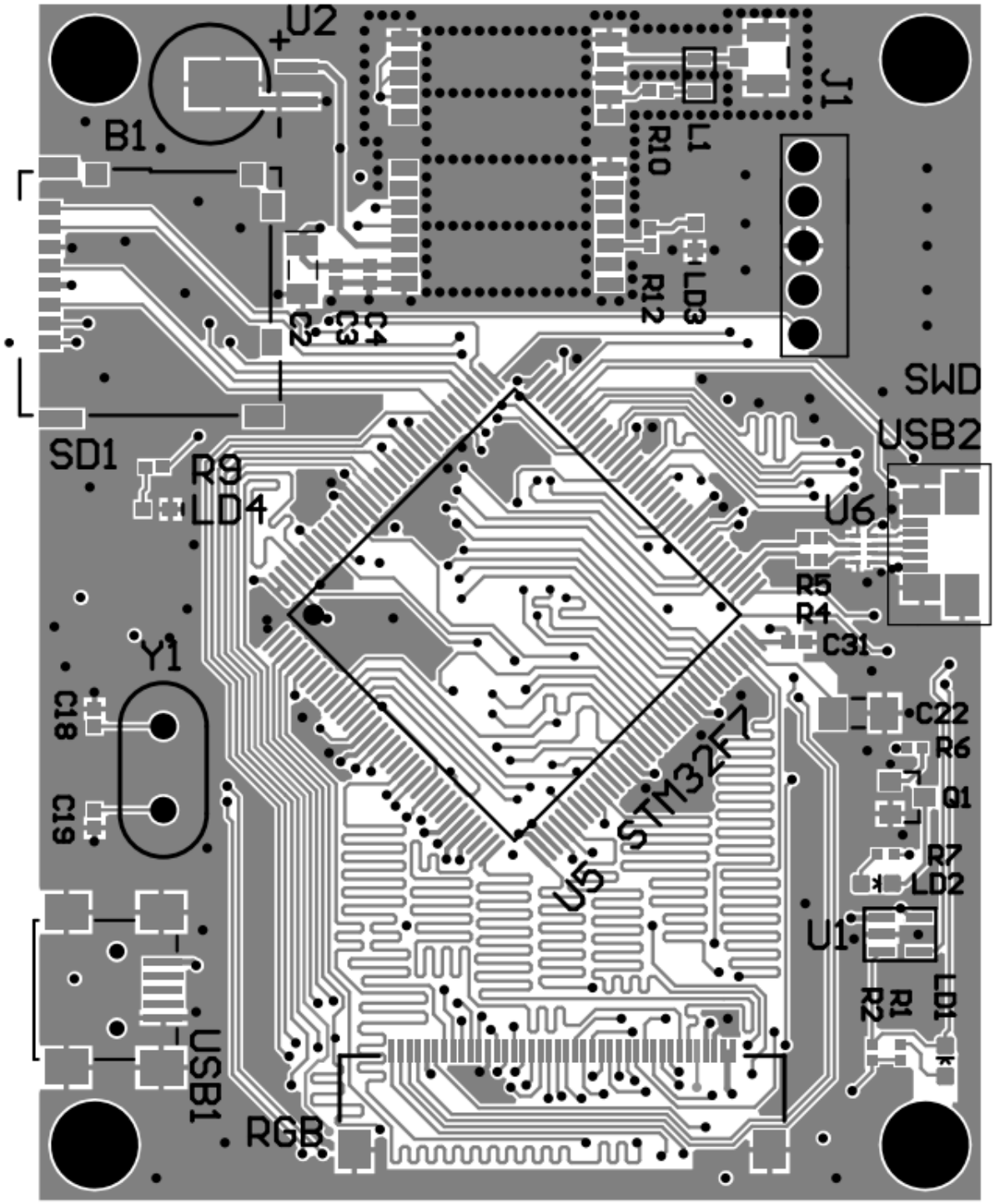
Rysunek 3.12: *Przykład dopasowania długości ścieżek na PCB*

3.2.5 Warstwa zasilania

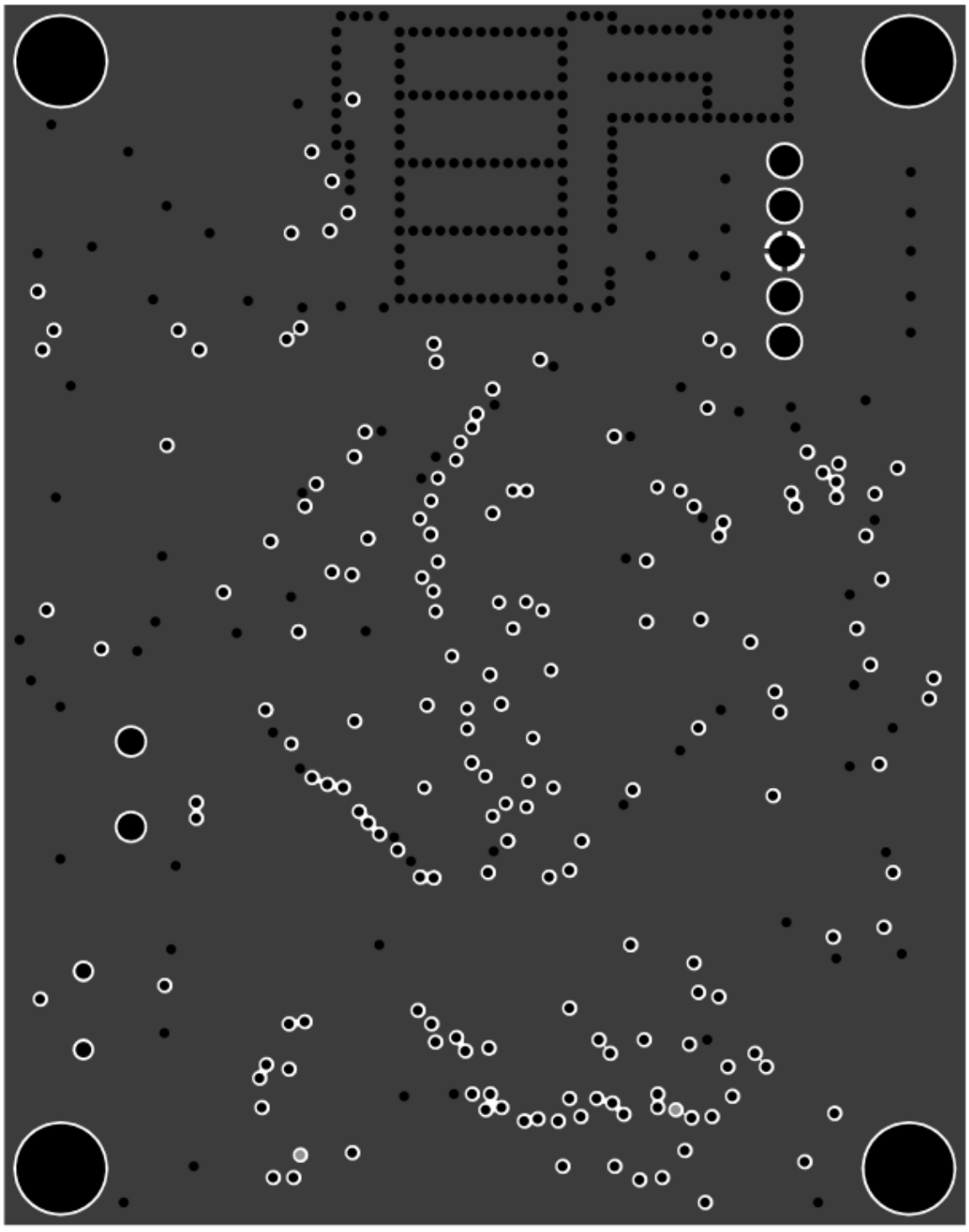
Większość układów na PCB jest zasilana napięciem 3,3V. Jednak interfejs USB i podświetlenie wyświetlacza wymaga napięciem 5V. Wymagało to odpowiedniego podzielania warstwy trzeciej. Zadbano o to żeby żadne ścieżki sygnałów wrażliwych na zakłócenia nie przechodziły nad przerwą pomiędzy polem zasilanie 3,3V i 5V. Nieciągłość warstwy referencyjne powoduje powstawanie pętli prądowych, ponieważ prąd powrotny szuka innej dostępnej ścieżki powrotu. Zjawisko to może powodować wzrost zakłóceń elektromagnetycznych emitowanych przez urządzenie. Warstwę przedstawia obraz **??** w podrozdziale 3.2.6.

3.2.6 Prezentacja PCB

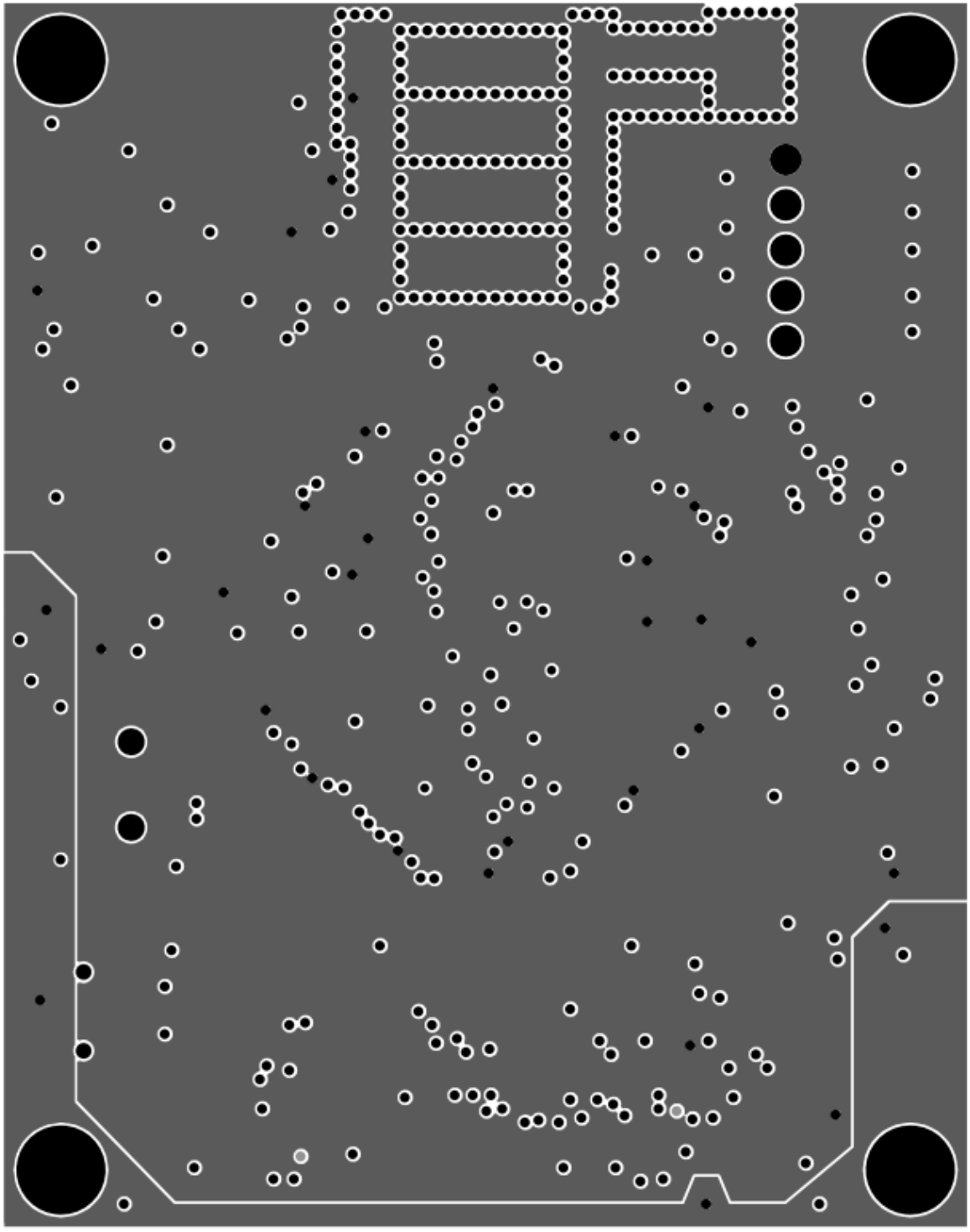
Poniżej przedstawiano obrazy wszystkich warstw z programu wykorzystanego do projektu oraz zdjęcie gotowego PCB.



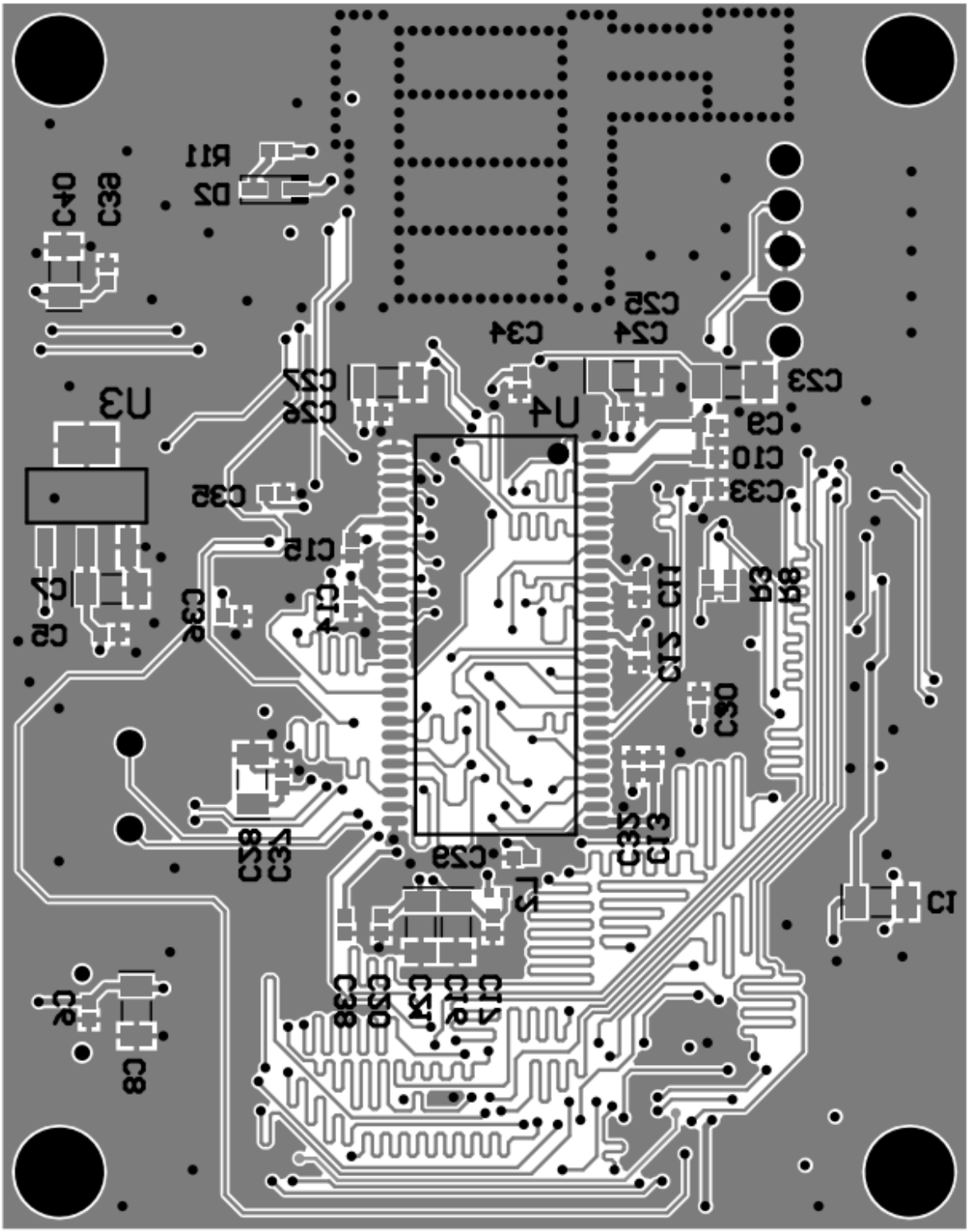
Rysunek 3.13: *Układ warstwy 1.*



Rysunek 3.14: *Układ warstwy 2.*



Rysunek 3.15: *Układ warstwy 3.*



Rysunek 3.16: *Układ warstwy 4.*

zdjecia płytki

Rozdział 4

Architektura oprogramowania

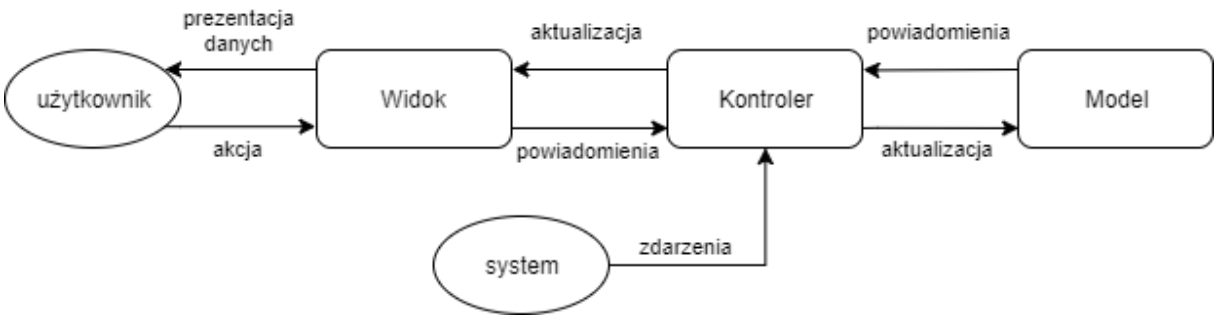
W tym rozdziale opisano strukturę programu wraz z modelem UML oraz opisem funkcjonalności poszczególnych klas. Przedstawiono wykorzystane wzorce projektowe, algorytmy i biblioteki. Opisano poszczególne elementy interfejsu graficznego oraz integrację oprogramowania z warstwą sprzętową.

4.1 Wzorzec projektowy MVC

Model Widok Kontroler (ang. Model View Controler) to wzorzec projektowy wykorzystywany przy tworzeniu aplikacji z interfejsem użytkownika. W MVC można wyróżnić trzy zasadnicze części:

- Model - reprezentacja logiki biznesowej (struktura danych oraz wykonywane na nich działania).
- Widok - warstwa prezentacji danych zawartych w modelu.
- Kontroler - wykonuje operacje na modelu.zapewnia komunikację pomiędzy widokiem i modelem. Obsługuję zadania użytkownika i systemy dotyczące modelu.

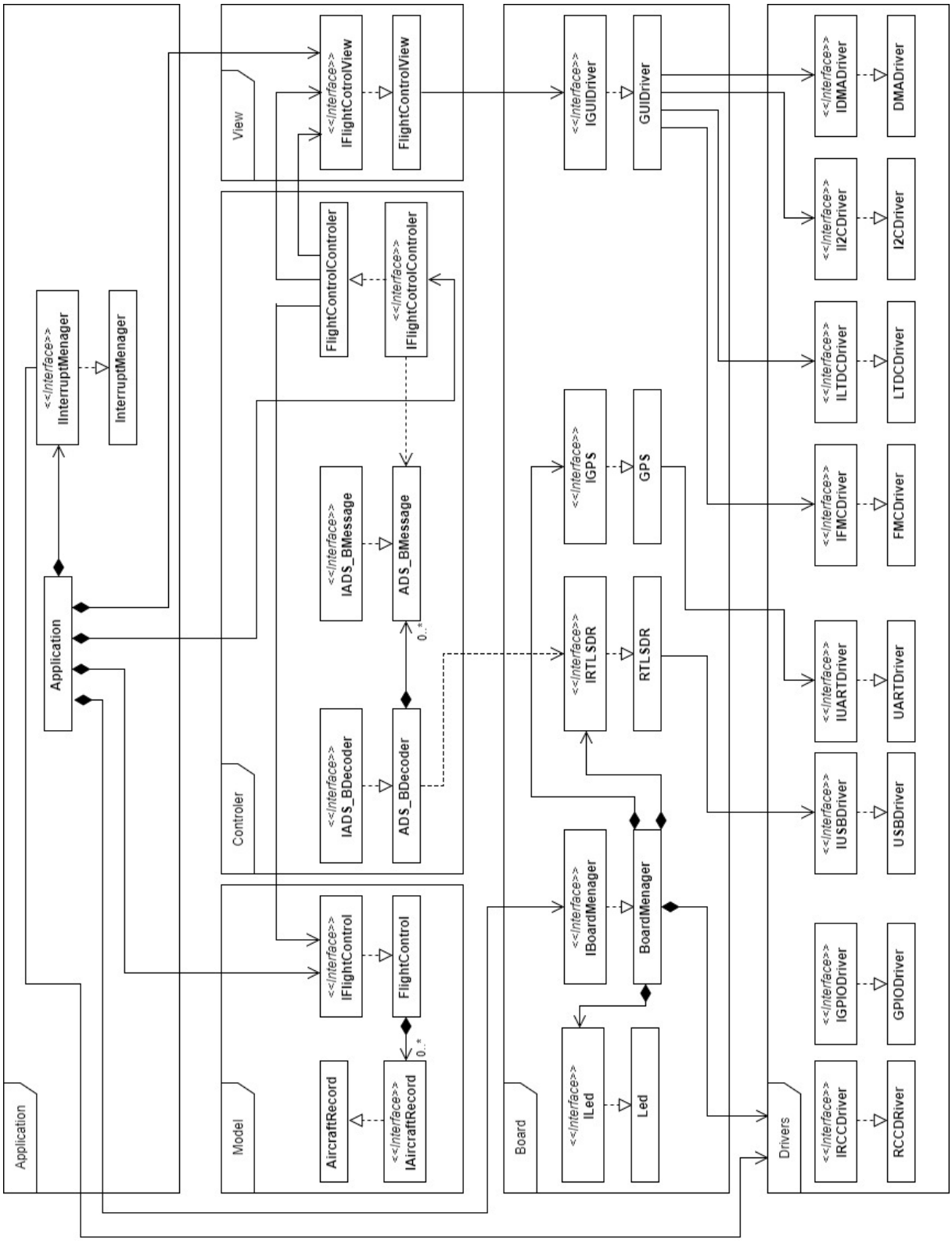
Wzorzec MVC zapewnia całkowitą niezależność modelu od widokiem, co ułatwia implementację obu części. Ponadto zapewnia przejrzystość projektu grupując klasy we wcześniej wymienione warstwy abstrakcji. Pozwala to ograniczyć liczbę powiązań pomiędzy komponentami. Poniżej przedstawiono diagram relacji dla wzorca MVC.



Rysunek 4.1: *Relacje pomiędzy komponentami MVC.*

4.2 Model UML

Poniżej przedstawiono diagram UML reprezentując powiązania i interakcje pomiędzy poszczególnymi klasami.



Rysunek 4.2: *Relacje pomiędzy komponentami MVC.*

Poniżej opisano funkcjonalności klas we wszystkich przestrzeniach nazw.

Application - przestrzeń zawierająca klasy najwyższego poziomu odpowiedzialne na działanie aplikacji.

> ***FlightRadarApp*** - główna klasa odpowiadająca za działanie i sterowanie aplikacją.
>
> ***InterruptMenager*** - klasa implementująca wywołania zwrotne dla obsługiwanych przerwań. Zapewnia synchronizacje pomiędzy systemem operacyjnym i procesami a odpowiednimi zdarzeniami sygnalizowanymi przez wywołania zwrotne.

Model - przestrzeń która, zawiera klasy związane z realizacją modelu z MVC

> ***AircraftRecord*** - klasa do przechowywania informacji o samolotach. Wpisy są zgrupowane w listę. Wezły są wyszukiwane według adresu ICAO. Odpowiada modelowi danyhc z MVC.
>
> ***FlightControl*** - implementuję logikę modelu, wraz z metodami dostępu i modyfikacji poszczególnych wpisów. Kontroluje czas życia poszczególnych węzłów oraz usuwa przeterminowane. Informuje kontroler o zmianach.

Controler - przestrzeń nazw zawierająca klasy związane z realizacją kontrolera z MVC.

> ***ADS_BDecoder*** - klasa odbierająca od obiektu RTLSDR ciągi próbek sygnału radiowego. Odpowiedzialna za wykrywanie ramek od dekodowanie ich do typu ADS_BMessage.
>
> ***ADS_BMessage*** - unia reprezentują odebraną wiadomość. W zależności od parametru downlink format jest interpretowana jako 56 lub 112 bitowa ramka.
>
> ***FlightControlControler*** - reprezentacja kontrolera w MVC. Klasa jest odpowiedzialna na dodawanie i modyfikowanie wpisów w zależności od odebranych wiadomości. Zajmuje się również przetwarzaniem akcji systemu (przerwania od zegary odliczającego czas życia wpisu) i użytkownika (interakcja z panelem dotykowym). Przekazuje dane z modelu do widoku i informuje o potrzebie odświeżenia interfejsu.

View - klasy realizujące funkcjonalność widoku z MVC

> ***FlightControlView*** - realizacja widoku w MVC. Klasa jest odpowiedzialna za prezentację danych z modelu oraz przesyłania akcji użytkownika do kontrolera.

Board - zawiera klasy będące wysokopoziomowymi interfejsami do obsługi urządzeń wchodzących w skład PCB. JEst odpowiedzialna za inicializację wszystkich klas z przestrzeni BoardSupport i Drivers.

> ***BoardMenager*** - klasa zapewniająca komunikację pomiędzy aplikacją a sprzętem.
>
> ***Led*** - interfejs do obsługi LED znajdującej się na płytce.
>
> ***RTLSDR*** - klasa

GPS - zajmuję się przetwarzaniem wiadomości przychodzących od modułu GPS poprzez UART. Zapewnia dostęp do zdekodowanych współrzędnych geograficznych.

Display - kontroler wyświetlacza realizujący wysokopoziomowe funkcjonalności takie jak menedżer okien i obsługa widżetów.

Drivers - przestrzeń nazw zawierająca wszystkie sterowniki. Zaimplementowane zapewniają interfejs do biblioteki HAL odpowiedzialnej za konfigurację i obsługę sprzętu.

RCCDriver - konfiguracją linii zegarowych oraz pętli PLL (ang. Phase Locked Loop) dla peryferiów oraz rdzenia w mikrokontrolerze.

GPIODriver - konfiguracja pinów dla wszystkich interfejsów wraz z obsługą wyjść i zewnętrznych przerwań.

USBDriver - sterownik do obsługi USB2.0 Full-Speed. Odpowiada za wykrywanie podłaczonych urządzeń oraz nawiązywnaia z nimi połączenia.

UARTDriver - zapewnia obsługę interfejsu UART (ang.Universal Asynchronous Receiver-Transmitter).

FMCDriver - konfiguracja kontrolera zewnętrznej pamięci mikrokontrolera. Wykorzystywany do komunikacji z układem SDRAM.

LTDCDriver - obsługa kontrolera wyświetlacza LCD-TFT wraz z konfiguracją wyświetlanych warstw.

I2CDriver - klasa do obsługi interfejsu I2C.

DMADriver - sterownik kontrolera DMA (ang Direct Memory Access). Wykorzystywany do przyspieszenia operacji renderowania kolejnych klatek dla wyświetlacza.

Komunikacja pomiędzy komponentami odbywa się przy pomocy interfejsów. Każda klasa dziedziczy po swoim interfejsie w którym zadeklarowane są wszystkie metody niezbędne do komunikacji z obiektem. Pozwala to na odseparowanie rzeczywistej implementacji klasy. Dzięki temu dany komponent można łatwo zmieniać bez wpływania na te które z nim oddziałują. Podejście to ułatwia testowanie oprogramowania, ponieważ wewnętrzną logikę można zastąpić spreparowanymi danymi. Za przykład może posłużyć klasa obsługująca sensor, która zawsze zwraca przygotowany wcześniej odczyt.

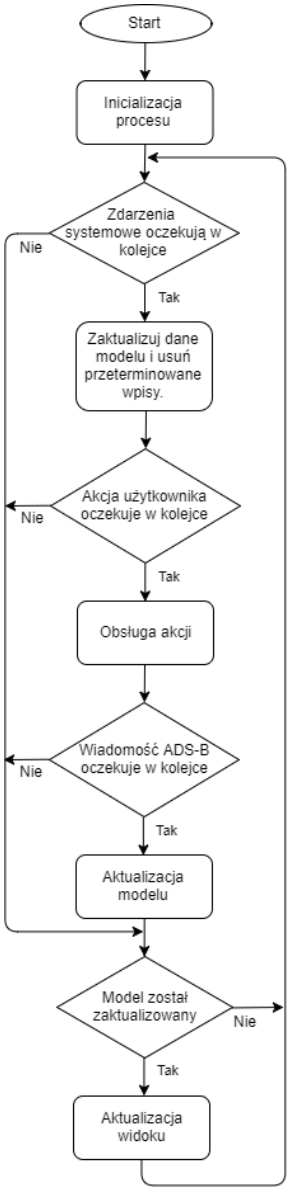
4.3 Procesy w systemie operacyjnym

W projekcie wykorzystano system operacyjny FreeRTOS, co pozwoliło podzielić poszczególne części aplikacji na procesy. Wielozadaniowość jest potrzebna by jednocześnie obsługiwać interfejs użytkownika, zarządzać akwizycją danych oraz wykonywać obliczenia związane z logiką programu, tak by użytkownik miał wrażenie płynnego działania aplikacji. Mechanizm semaforów wykorzystano to synchronizacji zadań z przerwaniami sprzętowym, natomiast kolejki wykorzystano do przesyłania pomiędzy nimi. danych i zdarzeń. Ponadto użyto liczników programowych do odliczania czasu życia poszczególnych wpisów w modelu. Ponadto użyto liczników programowych do odliczania czasu życia poszczególnych wpisów w modelu.

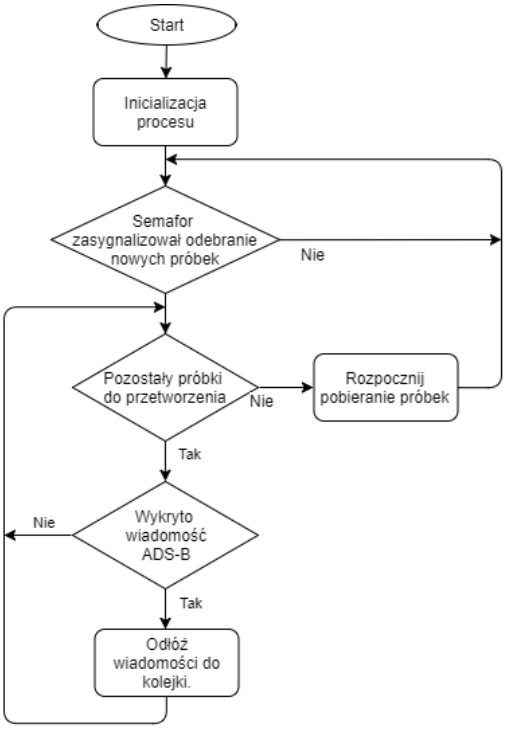
W systemie wydzielono następujące procesy:

- USBMonitorTask - odpowiedzialny za obsługę maszyny stanów USB oraz wykrywania i konfiguracje Tunera DVB-T jako SDR.
- RTLSDRDataAquisitionTask - proces zajmujący się obsługą komunikacji z SDR oraz wykrywaniem wiadomości ADS-B i przesyłaniem ich do FlightControlerTask poprzez kolejkę.
- FlightControlerTask - zadanie zajmuje się dekodowaniem wiadomości ADS-B. Przekazuje zdarzenia systemowe, takie jak przepełnienia liczników, oraz zdekodowanie ramki do kontrolera MVC FlightControlControler.
- GUITask - proces obsługujący interfejs graficzy poprzez klasę FlightControlView.

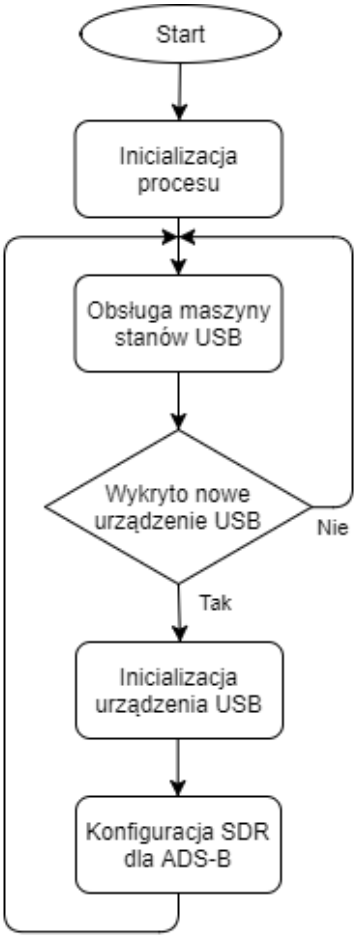
Poniżej przedstawiono schematy blokowe poszczególnych zadań.



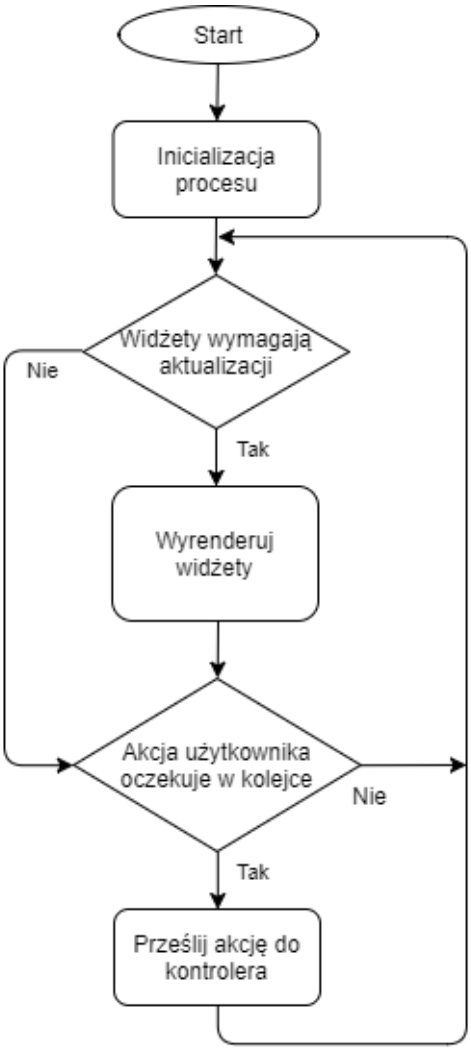
Rysunek 4.3: Schemat blokowy zadania FlightControlerTask



Rysunek 4.4: Schemat blokowy zadania RTLSDRDataAquisitionTask



Rysunek 4.5: Schemat blokowy zadania USBMonitorTask



Rysunek 4.6: Schemat blokowy zadania GUITask

4.4 Wykrywanie i dekodowanie wiadomości ADS-B

Wiadomości ADS-B składają się z preambuły czyli ciągu impulsów sygnalizujących wystąpienie ramki oraz bloku danych o długości odpowiednio 56 lub 112 bitów.

Rozdział 5

Interfejs użytkownika

W tym rozdziale opisano poszczególne elementy interfejsu użytkownika wraz z opisem użytych widżetów i formatem prezentowanych danych.

Rozdział 6

Podsumowanie