

POLITECHNIKA WROCŁAWSKA
WYDZIAŁ ELEKTRONIKI

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

PRACA DYPLOMOWA
INŻYNIERSKA

System lokalizacji samolotów z wykorzystaniem
ADS-B

Airplane tracking system using ADS-B

AUTOR:
Karol Szpila

PROWADZĄCY PRACĘ:

dr inż. Krzysztof Halawa
Katedra Informatyki Technicznej

OCENA PRACY:

Spis treści

1	Wstęp	2
2	Cele i założenia projektowe	4
2.1	Środowisko sprzętowe i wykorzystane narzędzia	4
3	Architektura sprzętowa	5
3.1	Schemat Urządzenia	5
3.2	Wykonanie PCB	10
4	Architektura oprogramowania	23
4.1	Wzorzec projektowy MVC	23
4.2	Model UML	24
4.3	Procesy w systemie operacyjnym	26
4.4	Wykrywanie i dekodowanie wiadomości ADS-B	28
5	Interfejs użytkownika	35
6	Podsumowanie	37
	Bibliografia	39

Rozdział 1

Wstęp

ADS-B (ang. Automatic Dependent Surveillance–Broadcast) to system służący do śledzenia statków powietrznych wykorzystywany w kontroli ruchu powietrznego. Powstał, aby uzupełnić pracę PSR (ang. Primary Surveillance Radar) i w przyszłości całkowicie go zastąpić. PSR jest radarem aktywnym bazującym na wysyłaniu fal elektromagnetycznych oraz pomiarze czasu ich powrotu po odbiciu od przeszkody. Jako wady takich systemów zaliczamy: brak informacji o wykrytym obiekcie poza jego położeniem i rozmiarem oraz wrażliwość na ukształtowanie terenu i warunki pogodowe. ADS-B bazuje na lokalizacji przy pomocy satelitów GPS. Statki powietrzne w sposób ciągły ustalają własną pozycję oraz nadają drogą radiową swoje położenie, prędkość oraz identyfikator.

SSR (ang. Secondary Surveillance Radar) został stworzony, aby uzupełnić pracę PSR o dodatkowe informacje odebrane od samolotu. Mode-S należy do takich systemów. Zapytania są wysyłane przez ATM (ang. Air Traffic Management) na częstotliwości 1030MHz, natomiast odpowiedzi zajmują pasmo 1090MHz. Nadawcę można zidentyfikować na podstawie 24-bitowego adresu ICAO (ang. International Civil Aviation Organization). ADS-B wykorzystuje Mode-S jako technologię transmisji danych.

Tabela 1.1: *Rodzaje wiadomości Mode-S*

Rodzaj wiadomości	Downlink Format	Zawartość
Mode-S (56 bitów)	DF0	Odpowiedz Short Air to Air ACAS
	DF4	Poziom lotu
	DF5	Identyfikator (Roll-call)
	DF11	Odpowiedz (All-call)
Mode-S (112 bitów)	DF16	Odpowiedz Long Air to Air ACAS
	DF17 (ADS-B)	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)

Tabela 4.5 prezentuje podział wiadomości Mode-S ze względu na rozmiar bloku danych oraz DF (ang. Downlink Format), czyli formatu odebranej ramki. Rozróżniamy pakiety krótkie o długości 56 bitów i rozszerzone 112 bitowe. DF0 i DF16 stosuje się w ACAS (ang. Airborne Collision Avoidance System), który służy zapobieganiu kolizji w kontroli ruchu powietrznego. Odpowiedz Short Air to Air (DF0), odbierają stacje ATM, natomiast Long Air to Air (DF16), informują poszczególne statki o możliwości zderzenia. Uczestnicy ruchu wysyłający ramkę DF5 potwierdzają, wyposażenie w transponder Mode-S. Ponadto, rozróżniamy odpowiedzi Mode-S EHS (ang. Enhanced Surveillance) DF20 i DF21 zawierające dodatkowe informacje niedostępne w ADS-B. Format pakietu zależy od wysłanego przez kontrolę naziemną BDS (ang. Comm-B Data Selector). Tylko SSR, który wysłał zapytanie Mode-S EHS jest w stanie zdekodować otrzymaną wiadomość, ponieważ nie zawiera ona wysłanego BDS. Poniżej przedstawiono parametry odpowiadające poszczególnym BDS.

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)

Rysunek 1.1: *BDS dla wiadomości DF20 i DF21*
<http://www.eurocontrol.int/articles/mode-s-operational-overview>

Wiadomość Mode-S można odbierać przy pomocy dowolnego odbiornika dostrojonego na częstotliwość 1090MHz. Tunery z układem RTL2832U da się łatwo przekształcić w SDR (ang. Software Defined Radio). SDR to system komunikacji radiowej w którym parametry radia są konfigurowane poprzez program bez ingerencji w sprzęt. Wspomniane urządzenie jest znacznie tańsze od profesjonalnych rozwiązań, co spopularyzowało ADS-B w zastosowaniach amatorskich.

Rozdział 2

Cele i założenia projektowe

Celem niniejszej pracy było stworzenie prototypu systemu pozwalającego na lokalizację oraz zbieranie danych o statkach powietrznych wyposażonych w transpondery ADS-B. W systemie można wyróżnić dwie zasadnicze części. Wewnętrzną odpowiedzialną za wykrywanie, zbieranie i dekodowanie wiadomości oraz wizualną, zajmującą się prezentacją danych oraz interakcją z użytkownikiem. Urządzenie jest alternatywą dla drogich i profesjonalnych systemów w przypadku zastosowań amatorskich.

2.1 Środowisko sprzętowe i wykorzystane narzędzia

W tym podrozdziale przedstawiono wykorzystanie w systemie urządzenia oraz środowiska programistyczne.

Jako odbiornik radiowy zostanie wykorzystany tuner DVB-T z układem RTL2832U wyposażony w interfejs USB. Urządzenie da się łatwo zamienić w SDR. Zdecydowano się na wykorzystanie mikrokontrolera firmy ST z serii STM32F767, ze względu na posiadanie peryferiów niezbędnych do obsługi urządzeń w systemie oraz dostępność narzędzi i oprogramowania. W projekcie wykorzystano wyświetlacz LCD-TFT o przekątnej 10.1 cala. Wybrano języki programowania C99 oraz C++11. Poniżej wymieniono użyte biblioteki:

- librtlsdr - obsługa Tunera DVB-T jako SDR,
- STM32 USB Host Library - komunikacja mikrokontrolera z radiem poprzez interfejs USB,
- STemWin - sterowanie wyświetlaczem z panelem dotykowym,
- STM32F7 HAL - konfiguracja mikrokontrolera i obsługa peryferiów.

System operacyjny FreeRTOS zastosowano w celu zapewnienia wielozadaniowości oraz mechanizmów zarządzania zasobami sprzętowymi. Do programowania urządzenia skorzystano z programatora ST-Link z zestawu uruchomieniowego STM32F429I-DISC1. Poniżej zaprezentowano wykorzystane programy:

- Atollic True Studio - środowisko do programowania mikrokontrolerów,
- STM32CubeMX - generowania kodu konfiguracyjnego dla układów STM32,
- CircuitMaker - wykonanie projektu PCB.

Rozdział 3

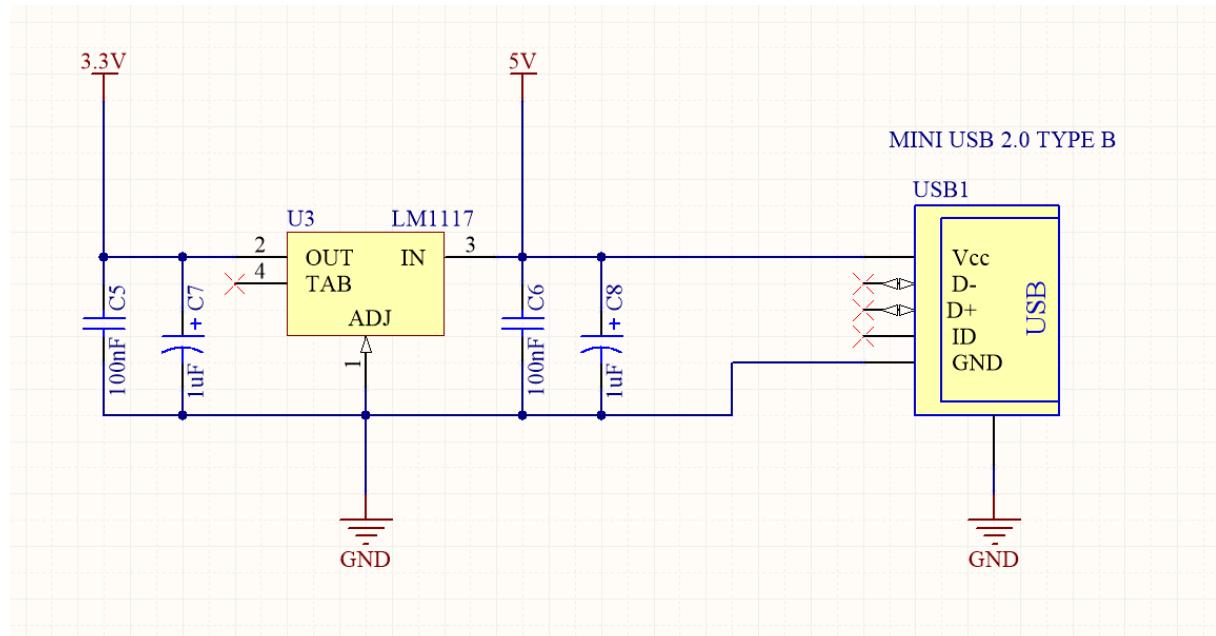
Architektura sprzętowa

W tym rozdziale zostanie opisana część sprzętowa projektu. Przedstawiono schemat oraz wykonanie PCB wraz z wykorzystanymi elementami oraz zewnętrznymi urządzeniami.

3.1 Schemat Urządzenia

Zasilanie

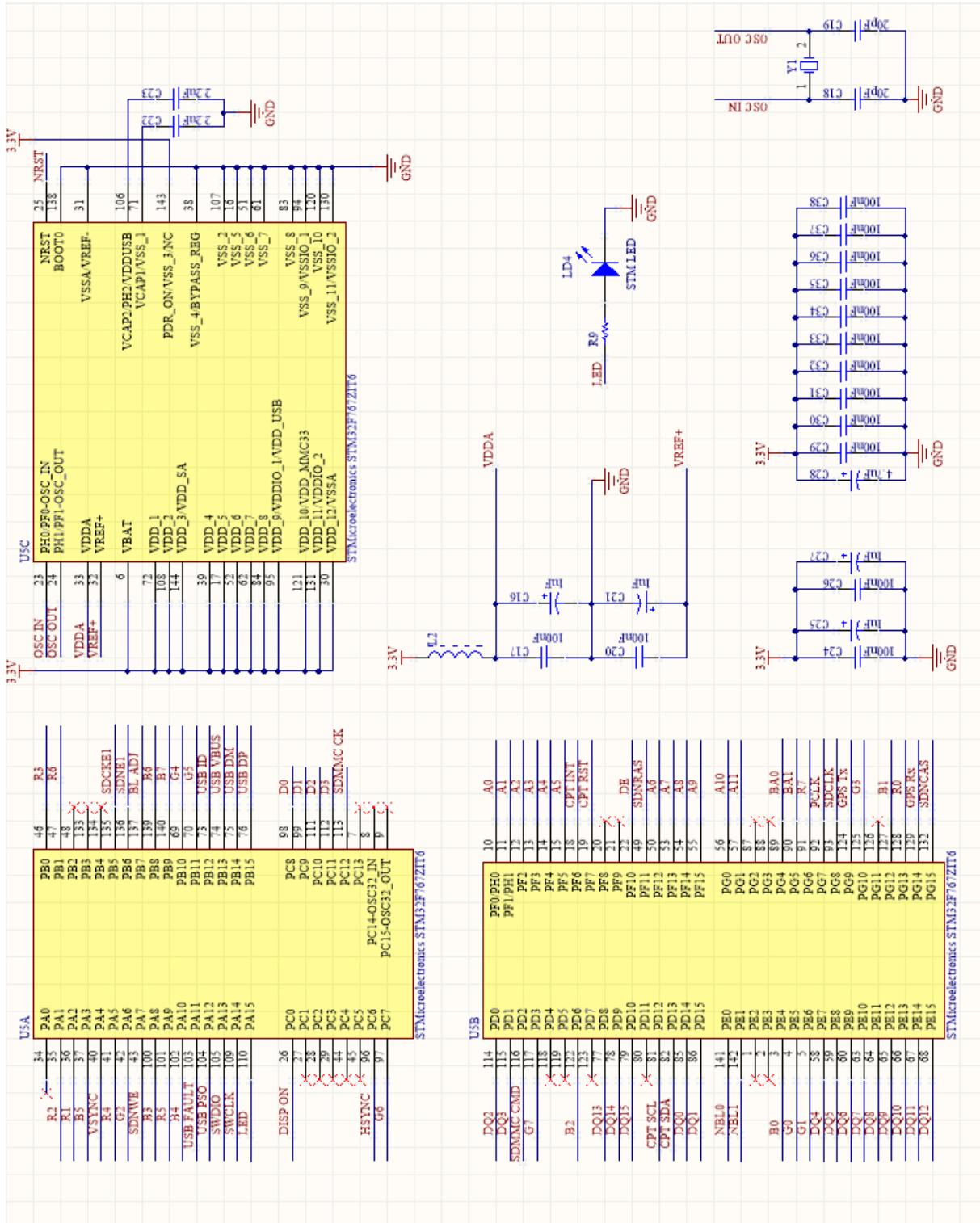
System jest zasilany z zewnętrznego źródła 5V poprzez port USB typu mini A. Pozwala to na podłączenie PCB do sieci poprzez przetwornicę, akumulatora lub innego urządzenia USB typu Host. Stabilizator LM1117 został zastosowany, aby obniżyć napięcie do wartości 3,3V przy jakiej pracują układy scalone znajdujące się na PCB. Kondensatory C5, C6, C7 i C8 zajmują się filtracją zasilania 3,3V i 5V [15]. Rysunek (3.1) przedstawia podłączenie poszczególnych elementów.



Rysunek 3.1: Schemat części zasilania PCB

Mikrokontroler

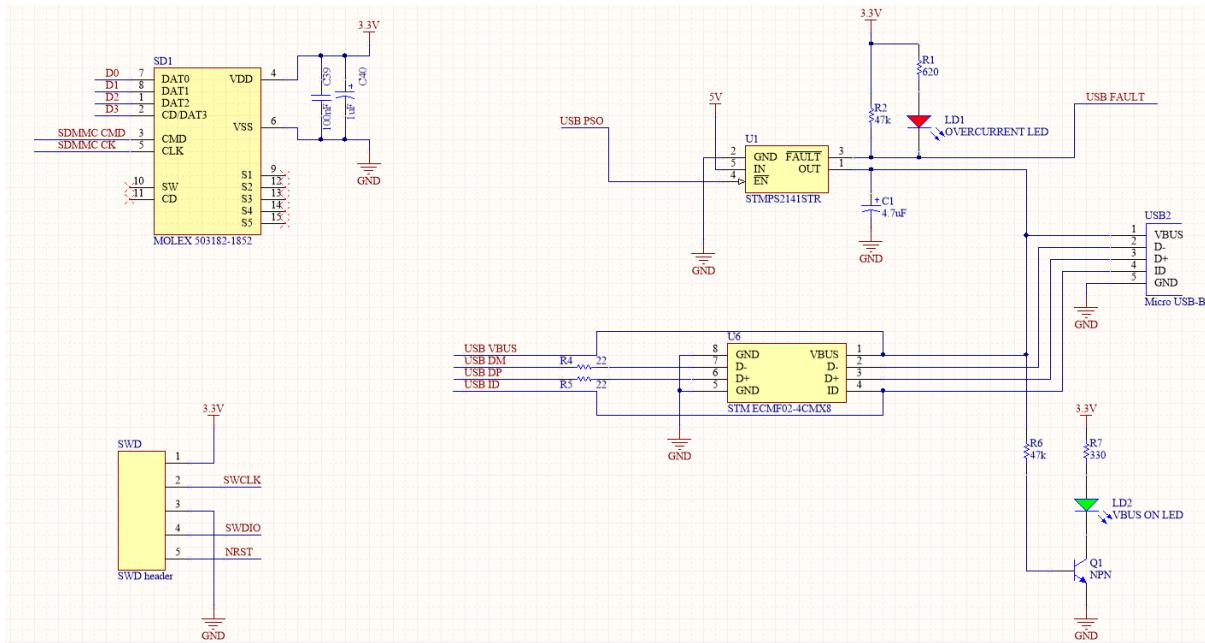
Wykorzystany mikrokontroler to STM32F767ZIT6. Układ został wybrany, ponieważ posiada najmniejszą obudowę (LQFP144) zawierającą wszystkie wymagane peryferia takie jak: LTDC do sterowania wyświetlaczem LCD, kontroler zewnętrznej pamięci pozwalający obsługiwać SDRAM i interfejsy komunikacyjne USB, I2C oraz UART. Poniżej przedstawiono schemat podłączenia [10].



Rysunek 3.2: Schemat podłączenia mikrokontrolera STM32F767ZIT6

Interfejsy komunikacyjne

System posiada gniazdo na kartę Micro SD, która może zostać wykorzystana, aby przechowywać skompresowane obrazy dla tła wyświetlacza lub mapy. Do programowania mikrokontrolera wykorzystano interfejs SWD ST-Link. Komunikacja z SDR odbywa się poprzez interfejs USB2.0 Full-Speed wyprowadzony poprzez gniazdo Micro USB-B, pozwalając w ten sposób zaoszczędzić na rozmiarze PCB. System jako Host będzie zasilać podłączony tuner. Zgodnie ze standardem maksymalny prąd wyjściowy wynosi 500mA [5]. Z tego powodu zastosowano przełącznik STMP2141STR [16]. W przypadku podania stanu wysokiego na pin USB PSO układ zostanie włączony. Jeżeli nie występują żadne sytuacje nieporządne takie jak przeciążenie prądowe czy zwarcie, zapali się zielona dioda sygnalizująca poprawne działanie. W przypadku jakichkolwiek problemów prąd zostanie natychmiast odcięty, a układ wystawi stan wysoki na pin FAULT informując o tym mikrokontroler i zapalając czerwoną diodę. W celu zabezpieczenia interfejsu przed niepożądanymi wyładowaniami elektrostatycznymi zastosowano układ STMECMF02-4CMX8 dedykowany dla USB2.0, służący do ochrony ESD (ang. Electrostatic Discharge), który pełni również funkcję filtra EMI, (ang. Electromagnetic Interference) [17]. Dalej pokazano schemat połączenia elementów.

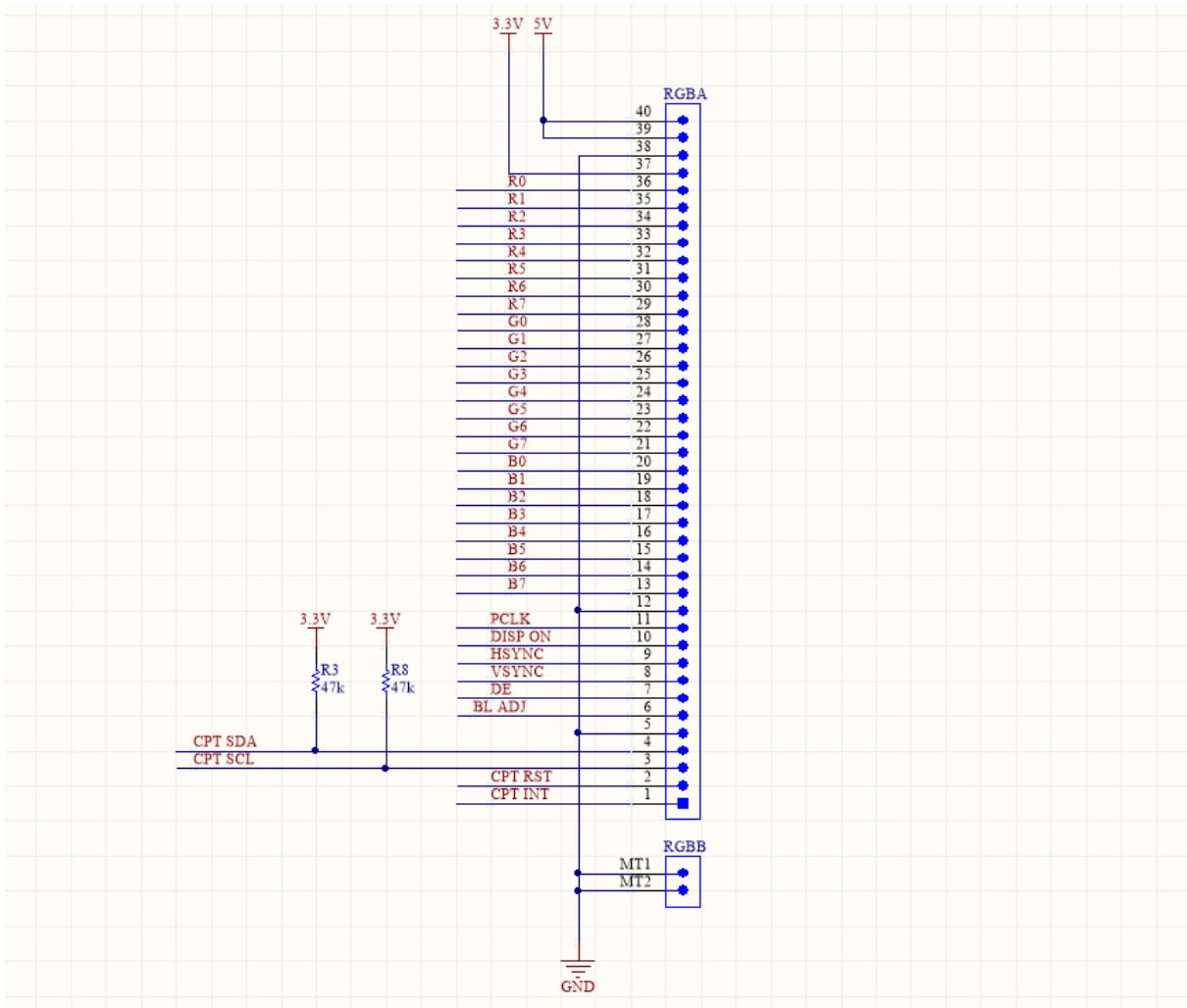


Rysunek 3.3: Schemat połączenia zewnętrznych interfejsów

Wyświetlacz

Zgodnie z założeniem, system ma posiadać interfejs użytkownika. Do tego celu wybrano wyświetlacz HY101CTP o przekątnej 10,1" i rozdzielcości 1024 na 600 pikseli. Matryca jest sterowana poprzez MIPI-DPI (ang. Mobile Industry Processor Interface - Display Parallel Interface). Pojemnościowy panel dotykowy komunikuje się z mikrokontrolerem za pośrednictwem interfejsu I2C. Stan wysokim na pinie DISP ON włącza podświetlenie, a jasność zależy od wypełniania sygnału PWM o częstotliwości 10 KHz na liniach BL ADJ. Rezystory R3 i R8 mają za zadanie wymusić logiczne 1 na liniach I2C, ponieważ są sterowane w trybie otwartego drenu. Taka konfiguracja pozwala tylko zewrzeć linie do masy

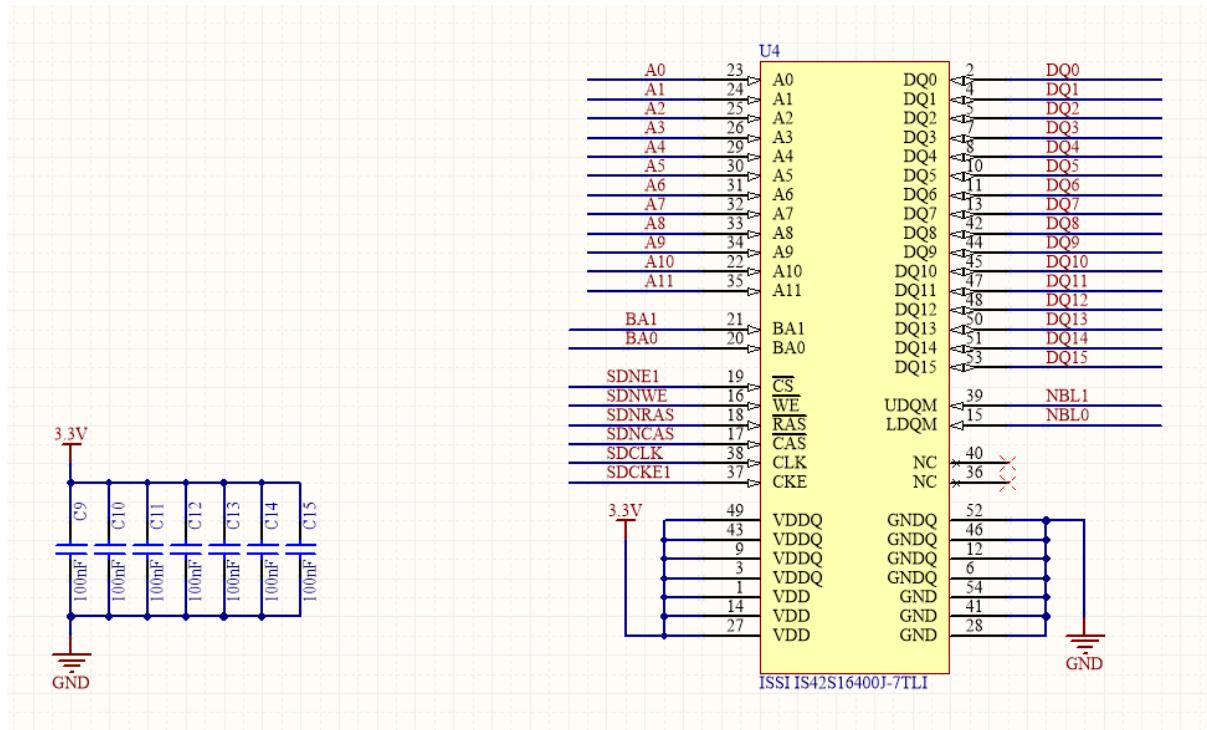
zmieniając stan na niski. Dzięki temu wyeliminowano możliwość podłączania zasilania i masy po obu stronach ścieżki, mogącego prowadzić do gwałtownego wzrostu mocy i uszkodzenia układu. Poniżej przedstawiono schemat podłączenia [18].



Rysunek 3.4: *Schemat podłączenia wyświetlacza*

SDRAM

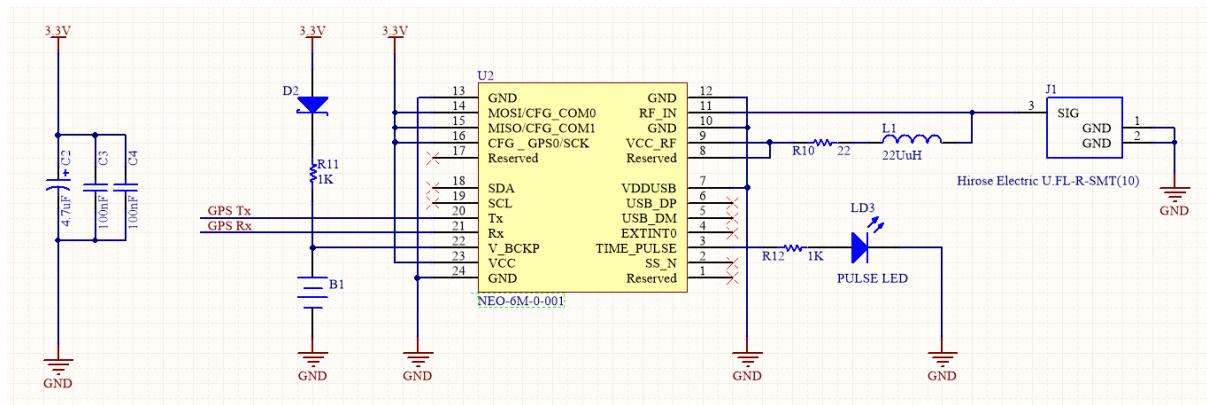
Interfejs graficzny potrzebuje pamięci, aby przechowywać wyświetlane ramki. Założono, że warstwy będą przechowywane w formacie ARGB8888 (po bajcie na każdy kolor i kanał alfa). Układ LTDC sprzętowo łączy dwie warstwy w wynikową, którą wysyła do wyświetlacza. Zdecydowano, że potrzeba pamięci wystarczająco dużej, by zmieścić trzy bufore. Pierwszy, przeznaczony na tło, będące niezmienne podczas działania aplikacji. Dwa pozostałe służą naprzemiennej prezentacji danych i położenia statków powietrznych. Jest to implementacja mechanizmu podwójnego buforowania, pozwalająca wyeliminować migotanie matrycy. Trzy warstwy o rozmiarze 1024 na 600 pikseli w formacie ARGB8888 zajmą 57600 Kb. W projekcie zdecydowano się wykorzystać układ IS42S16400J-7TLI posiadający 65536 Kb pamięci [13]. Na rysunku (3.5) pokazano schemat podłączenia układu.



Rysunek 3.5: Schemat podłączenia układu SDRAM

Moduł GPS

Aby system był w stanie obliczyć odległość od namierzonego statku powietrznego i poprawnie zaznaczyć jego pozycję na radarze, potrzeba lokalizacji odbiornika. Do tego zadania wybrano układ NEO-6M-0-001 z zewnętrzną anteną. Bateria służy potrymaniu zasilania, by urządzenie można było uruchomić poprzez ciepły start, co pozwala zaoszczędzić czas potrzebny na konfigurację modułu. Komunikacji z mikrokontrolerem odbywa się interfejs UART (ang. Universal Asynchronous Receiver Transmitter). Poniżej przedstawiono schemat połączenia



Rysunek 3.6: Schemat podłączenia modułu GPS

3.2 Wykonanie PCB

Poniżej szczegółowo opisano projekt oraz wykonanie PCB wraz z obliczeniami. Oписанo decyzje projektowe wynikające z ograniczeń technologicznych oraz występowania niepożądanych zjawisk fizycznych.

Parametry PCB

Zdecydowano się na wykonanie PCB technologią czterowarstwową. Dzięki temu udało się zmniejszyć rozmiary urządzenia oraz zapewnić dobre ekranowanie pomiędzy warstwami sygnałowymi. Ponadto ciągłość warstw referencyjnych (masy i zasilania) pozwala, by prądy powrotne przepływały możliwe najkrótszą drogą zmniejszając w ten sposób emisję EMI. Poniżej przedstawiono tabelę z układem warstw PCB [20].

Tabela 3.1: *Układ 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 sygnałowe są z miedzi natomiast jako dielektryk wykorzystano materiał FR-408 o stałej elektrycznej $\varepsilon_r = 3.66$ [7].

GPS

Zgodnie z instrukcją integracji sprzętowej układu NEO-6M-0-001, ścieżka antenowa powinna mieć impedancję dopasowaną do $Z_0 = 50\Omega$ [8], którą obliczono z następującego wzoru [1]:

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

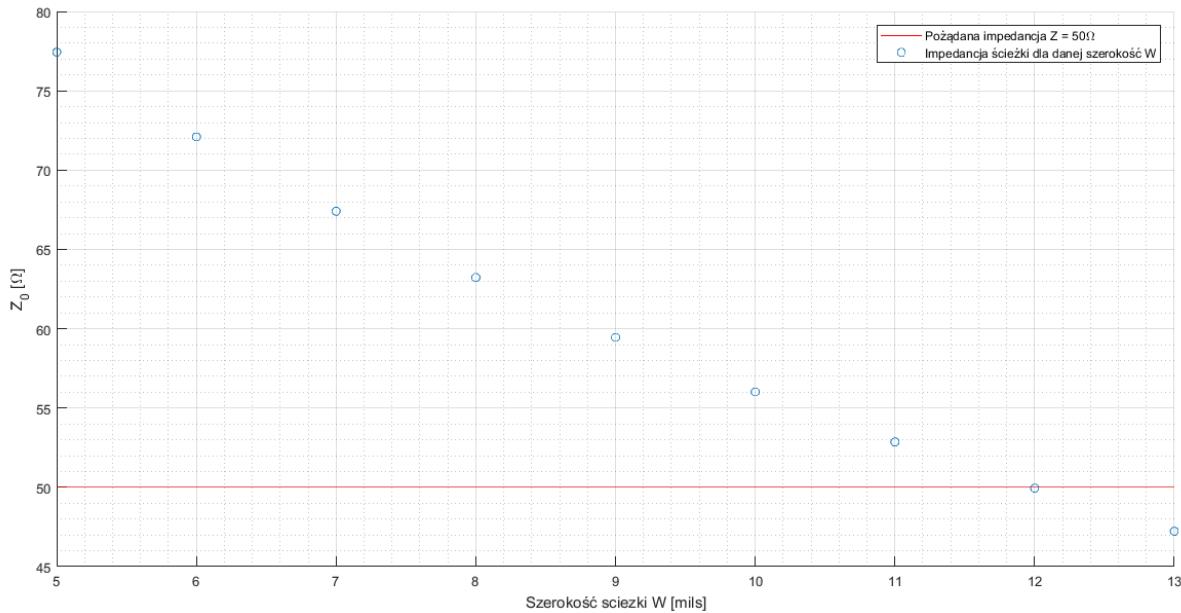
dla

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

$$1 < \varepsilon_r < 15 ,$$

gdzie:

- ε_r - stała elektryczna,
- H - grubość dielektryka,
- W - szerokość ścieżki,
- T - grubość ścieżki.



Rysunek 3.7: Wykres impedancji ścieżki w zależności od szerokości

Rysunek (3.7) przedstawia impedancję ścieżki antenowej modułu GPS w zależności od szerokości, obliczoną ze wzoru (3.1). Z tabeli 3.1 odczytano T = 1,4 mil, H = 6,7 mil oraz ε_r = 3,66. Czerwoną linią zaznaczono pożądane dopasowanie Z₀ = 50 Ω. Założono, że W zostanie wyznaczone z dokładnością do 1 mil. Obliczenia wykonano zaczynając od najmniejszej szerokości gwarantowanej przez producenta W = 5 mil, aż do pierwszego punktu pod prostą z optymalnym dopasowaniem. Najlepszy wynik uzyskano w przypadku W = 12 mil, gdzie Z₀ ≈ 49,95 Ω. Dla Z₀ wyznaczono współczynnik odbicia [1], który reprezentuje jaka część sygnału została stracona w wyniku niedopasowania impedancji przy przejściu pomiędzy liniami transmisyjnymi. Parametr ten wyraża się wzorem:

$$\Gamma = \frac{Z_l - Z_s}{Z_l + Z_s}, \quad (3.2)$$

gdzie:

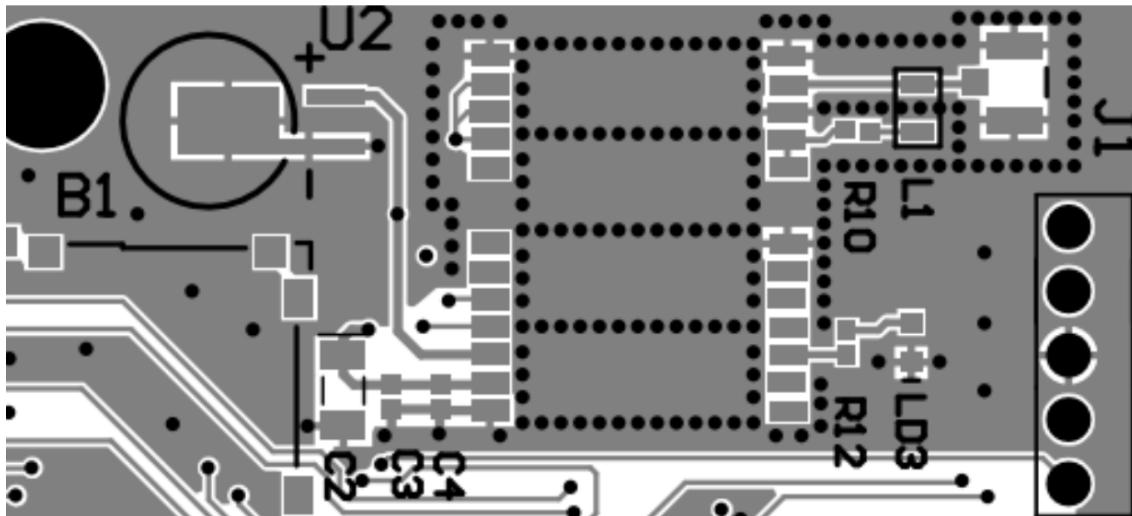
- Z_l - impedancja obciążenia, w tym przypadku linii transmisyjnej Z₀ = 49,95 Ω,
- Z_s - impedancja źródła, czyli wejścia antenowego modułu GPS Z_s = 50 Ω.

Korzystając ze wzoru (3.2) obliczono:

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

Po przeliczeniu na wartość procentową otrzymano Γ · 100% = 0.0005 · 100% = 0.05% sygnału odbitego przez linie transmisyjną. Strata jest pomijalnie mała, zatem pozostało przy szerokości W = 12 mil.

Zgodnie z zaleceniami instrukcji wykonano serię przelotek pod i dookoła układu GPS (U2) [8]. 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 mniejszą emisję EMI. Rysunek (3.8) prezentuje projekt PCB dla układu GPS.



Rysunek 3.8: Projekt PCB dla modułu GPS

USB

W projekcie wykorzystano wbudowany w mikrokontroler interfejs USB2.0 Full-speed o maksymalnej przepustowości 12Mb/s. Zgodnie ze standardem, należy dopasować impedancję różnicową pomiędzy liniami transmisyjnymi sygnału nieodwróconego i odwróconego do $Z_d = 90 \pm 15\% \Omega$ [5]. Parametr ten wyraża się wzorem [1]:

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

dla

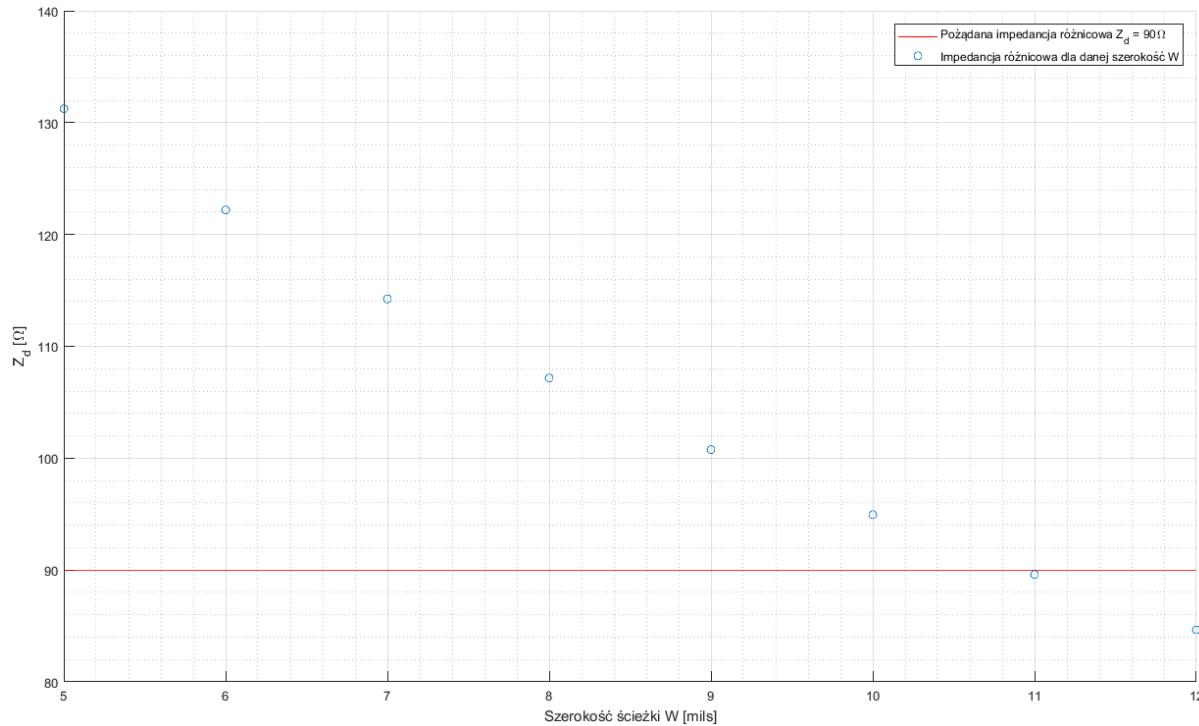
$$0,1 < W/H < 2,0 ,$$

$$1 < \varepsilon_r < 15 ,$$

gdzie:

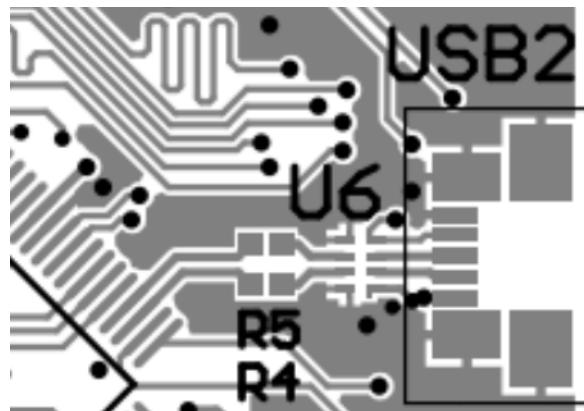
- ε_r - stała elektryczna,
- H - grubość dielektryka,
- W - szerokość ścieżki,
- T - grubość ścieżki,
- D - odległość pomiędzy ścieżkami.

Rysunek (3.9) przedstawia impedancję różnicową linii transmisyjnych w zależności od szerokości, obliczoną ze wzoru (3.4). Parametry $T = 1,4$ mil, $H = 6,7$ mil i $\varepsilon_r = 3,66$ zaczerpnięto z tabeli 3.1. Założono odległość pomiędzy ścieżkami $D = 8$ mil równą rozstawiowi pinów w mikrokontrolerze. Czerwoną linią zaznaczono pożądane dopasowanie wynoszące $Z_d = 90 \Omega$. Założono że W zostanie wyznaczone z dokładnością do 1 mil. Obliczenia wykonano zaczynając od najmniejszej szerokości gwarantowanej przez producenta $W = 5$ mil, aż do punktu pod prostą z optymalnym dopasowaniem. Najlepszy wynik otrzymano dla $W = 11$ mil, gdzie $Z_d \approx 89.56 \Omega$. $76,5\Omega \leq Z_d \leq 103,5\Omega$. Z_d dla $W = 11$ mil spełnia wymagania standardu.



Rysunek 3.9: Wykres impedancji różnicowej w zależności od szerokości ścieżki

W projekcie zastosowano następujące reguły, aby zapewnić integralność sygnałów [6]. Wszystkie elementy zostały umieszczone na górnej warstwie sygnałowej, aby nie stosować przelotek, które wprowadzają niepożądane pojemności i indukcyjności linii transmisyjnych. Ścieżki są zginane pod kątem nie większym niż 45° . Warstwą referencyjną dla sygnałów jest masa co zapewnia lepsze ekranowanie. Linie różnicowe oddzielono polem masy od innych sygnałów o co najmniej 50 mil. Powyższe zabiegi pozwoliły ograniczyć zjawiska utrudniające dopasowania impedancji. Układ U6 działa jako zabezpieczenie przeciwko wyładowaniom elektrostatycznym mogącym uszkodzić urządzenie oraz pełni funkcję filtra EMI. Rezystory R5 i R4 służą szeregowej terminacji. Zgodnie z notą katalogową mikrokontrolera ich rezystancja powinna wynosić 22Ω [10]. Dzięki temu zabiegowi czasy narastania zboczy sygnału są większe co zmniejsza emisję EMI. Poniżej przedstawiono część projektu PCB z interfejsem USB.



Rysunek 3.10: Projekt PCB dla interfejsu USB

Interfejsy szybkie

Częstotliwość sygnału zegarowego dla SDRAM 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 [11]:

$$CLK = W \cdot H \cdot fps , \quad (3.5)$$

gdzie:

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

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 [6]. Zadbano by warstwy referencyjne pod ścieżkami sygnałowymi były ciągłe, aby zapewnić ekranowanie i ograniczyć przesłuchy od linii po przeciwniej stronie płytki. Ponadto dopasowano długości wszystkich ścieżek w interfejsie, by sygnały przychodziły w możliwie podobnym czasie. Wzorując się na projekcie referencyjnym płytka ewaluacyjnej STM32F746G-DISCO [14] zadbano, by różnica długości pomiędzy liniami dla obu interfejsów nie była większa niż 100 mil. Opóźnienie linii transmisyjnej, czyli czas jaki sygnał potrzebuje na pokonanie określonej drogi wyraża się wzorem [1]:

$$t = \frac{l\sqrt{\varepsilon_r}}{c} , \quad (3.6)$$

gdzie:

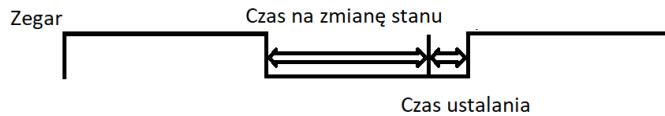
- l - długość linii transmisyjnej,
- ε_r - stała elektryczna,
- c - szybkość rozchodzenia się fali w próżni.

Korzystając ze wzoru (3.6) dla parametrów $l = 0.00254$ m (100 mil), $c = 299\ 792\ 458\frac{m}{s}$ oraz $\varepsilon_r = 3,66$ wyznaczono opóźnienie, czyli czas jaki dzieli sygnały w liniach o długości różniącej się o 100 mil. Poniżej przedstawiono wynik obliczeń.

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

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 są zatrzaskiwane przez konwerter przy zboczu narastającym. Wszystkie sygnały powinny dojść do wyświetlacza od momentu wystąpienia zboczka opadającego do narastającego z uwzględnieniem czasu na ustalenie 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.



Rysunek 3.11: Diagram ograniczeń czasowych dla interfejsów szybkich

Zatem maksymalna rozbieżność czasowa pomiędzy sygnałami wyraża się wzorem.

$$t_{max} = t_{low} - t_{setup} , \quad (3.7)$$

gdzie:

- t_{low} - czas trwania stanu niskiego dla sygnału zegarowego,
- t_{setup} - czas na ustalanie się sygnału.

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

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

W instrukcji THC63LVDM83D znaleziono $t_{setup} = 2000ps$ [12]. Korzystając ze wzoru (3.7) wyznaczono:

$$t_{max} = 12207ps - 2000ps = 10207ps .$$

Ograniczenia nie zostały przekroczone, ponieważ $t_{max} = 10207ps \geq t = 16,21ps$.

Poniżej przedstawiono tabelę z czasami ustalania dla wszystkich sygnałów interfejsu odczytanych z noty układu IS42S16400J-7TLI [13].

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 $t_{setup} = 1500ps$.

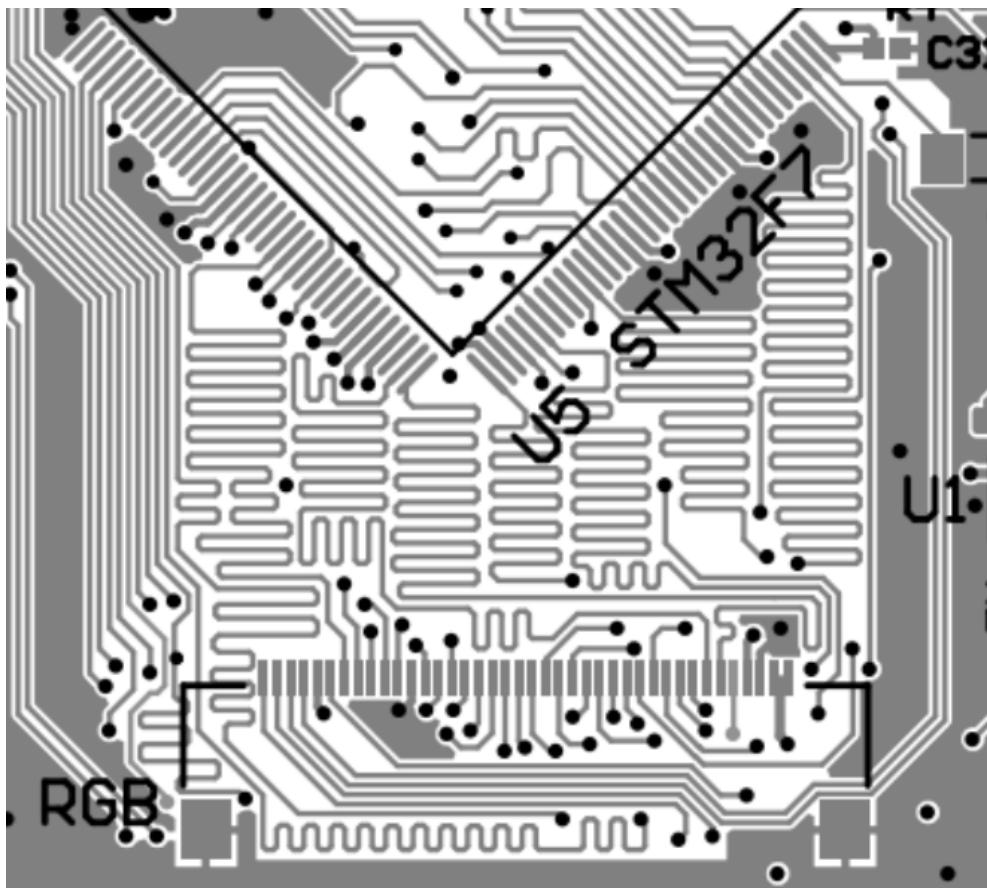
W przypadku układu FMC wykorzystywanego do obsługi SDRAM, komunikacja odbywa się w obie strony. Ponownie czas trwania stanu niskiego sygnału zegarowego wynosi minimalnie 45% okresu. Zatem:

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

Korzystając z równania (3.7) obliczono:

$$t_{max} = 4167\text{ps} - 1500\text{ps} = 2667\text{ps} .$$

Ponownie $t_{max} = 2667\text{ps} \geq t = 16,21\text{ps}$, zatem ograniczenia czasowe nie zostały przekroczone. Poniżej przedstawiono część projektu PCB, gdzie można zauważać meandry, których celem jest wyrównanie różnicy długości pomiędzy ścieżkami sygnałowymi.



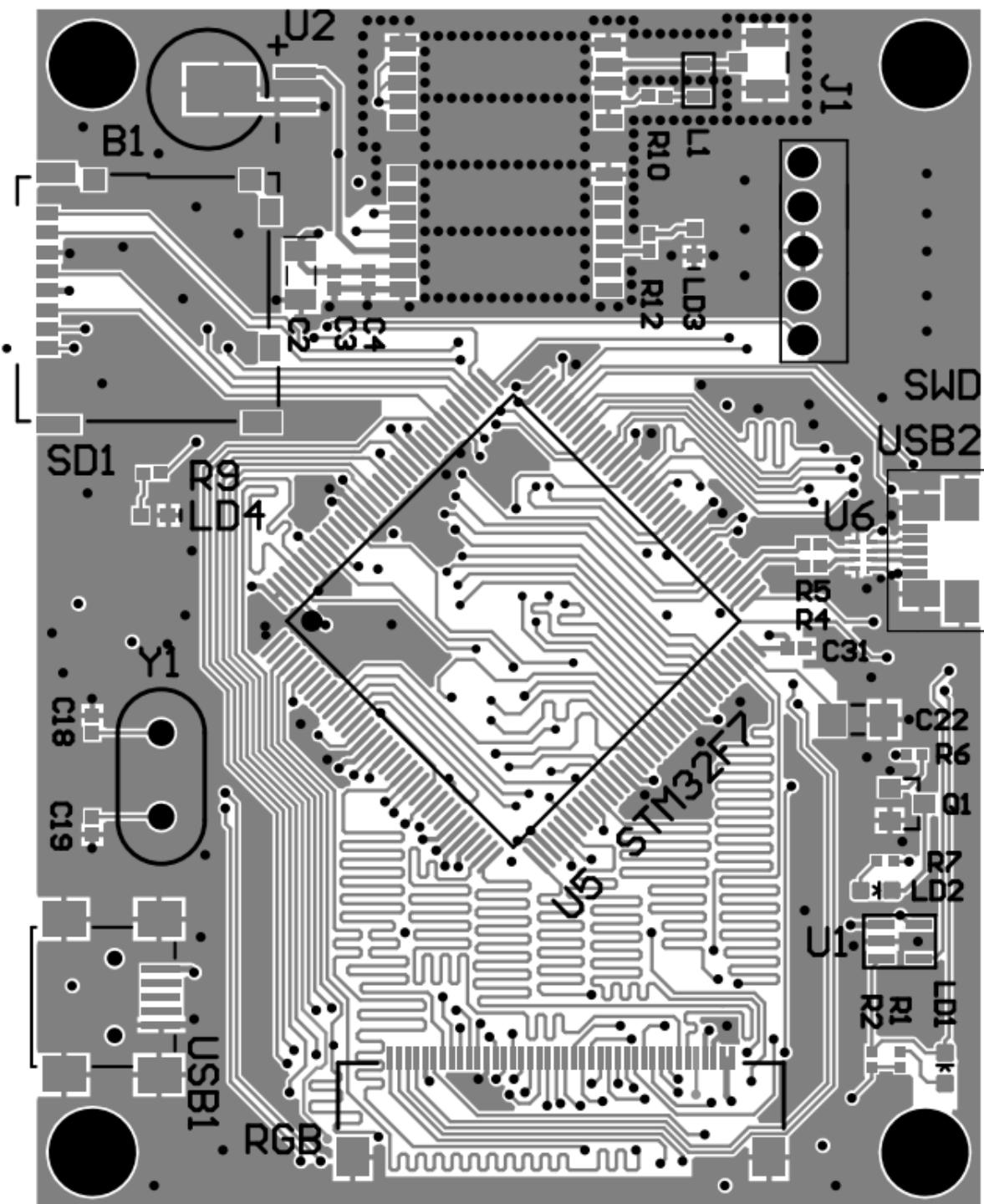
Rysunek 3.12: Przykład dopasowania długości ścieżek na PCB

Warstwa zasilania

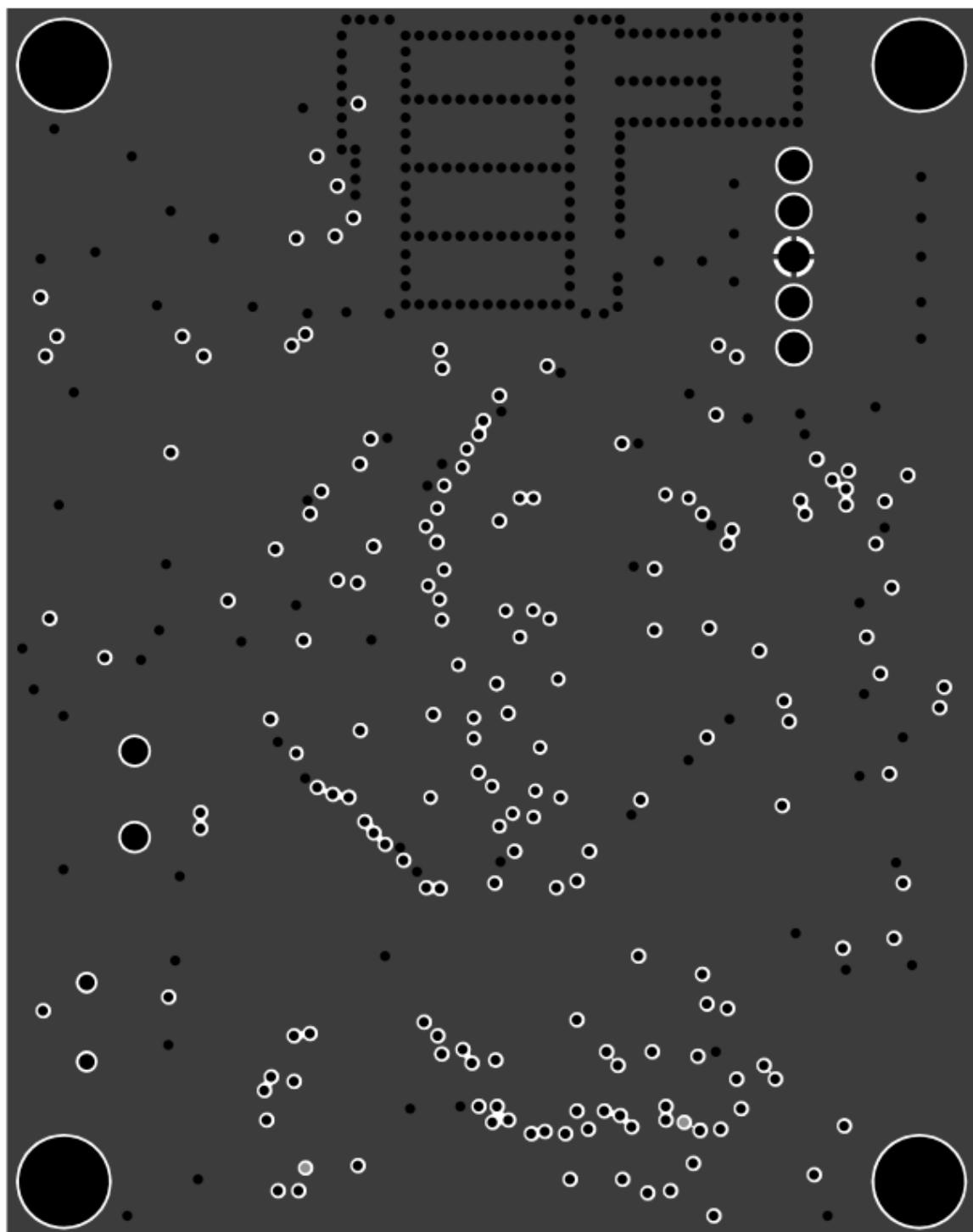
Większość układów na PCB pracuje przy napięciu 3,3V. Jednak zasilanie podłączonego urządzenia USB i podświetlenie wyświetlacza potrzebują 5V. Wymagało to odpowiedniego podzielenia warstwy trzeciej. Zadbano o to, żeby żadne ścieżki sygnałów wrażliwych na zakłócenia nie przechodziły nad przerwą pomiędzy polem zasilania 3,3V i 5V. Nieciągłość warstwy referencyjnej pod liniami powoduje powstawanie pętli prądowych [1]. Zjawisko to może powodować wzrost zakłóceń elektromagnetycznych emitowanych przez urządzenie. Warstwę przedstawia rysunek (3.14) w podrozdziale 3.2.

Prezentacja PCB

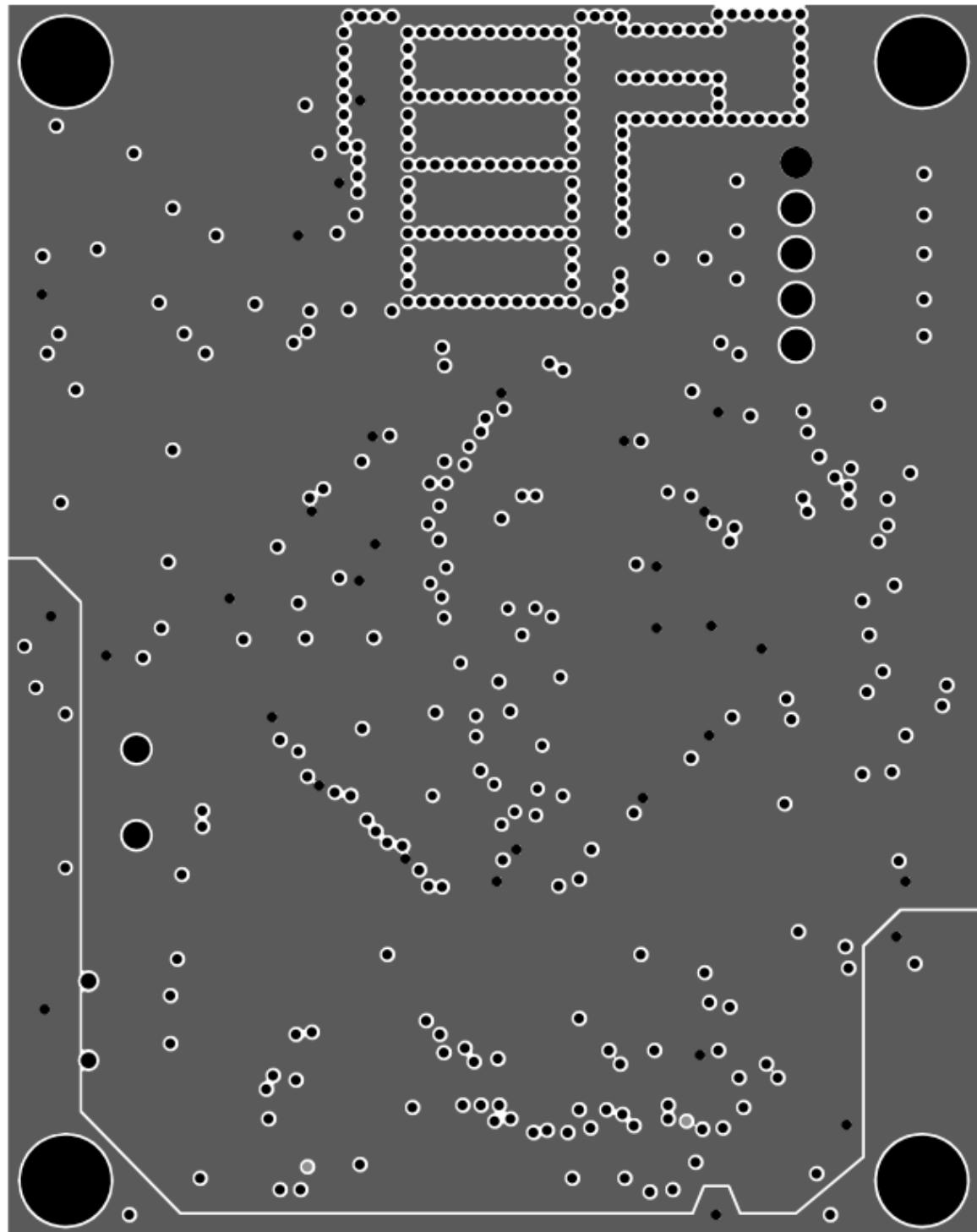
Poniżej przedstawiono wydruk wszystkich warstw z programu CircuitMaker oraz zdjęcie gotowego PCB.



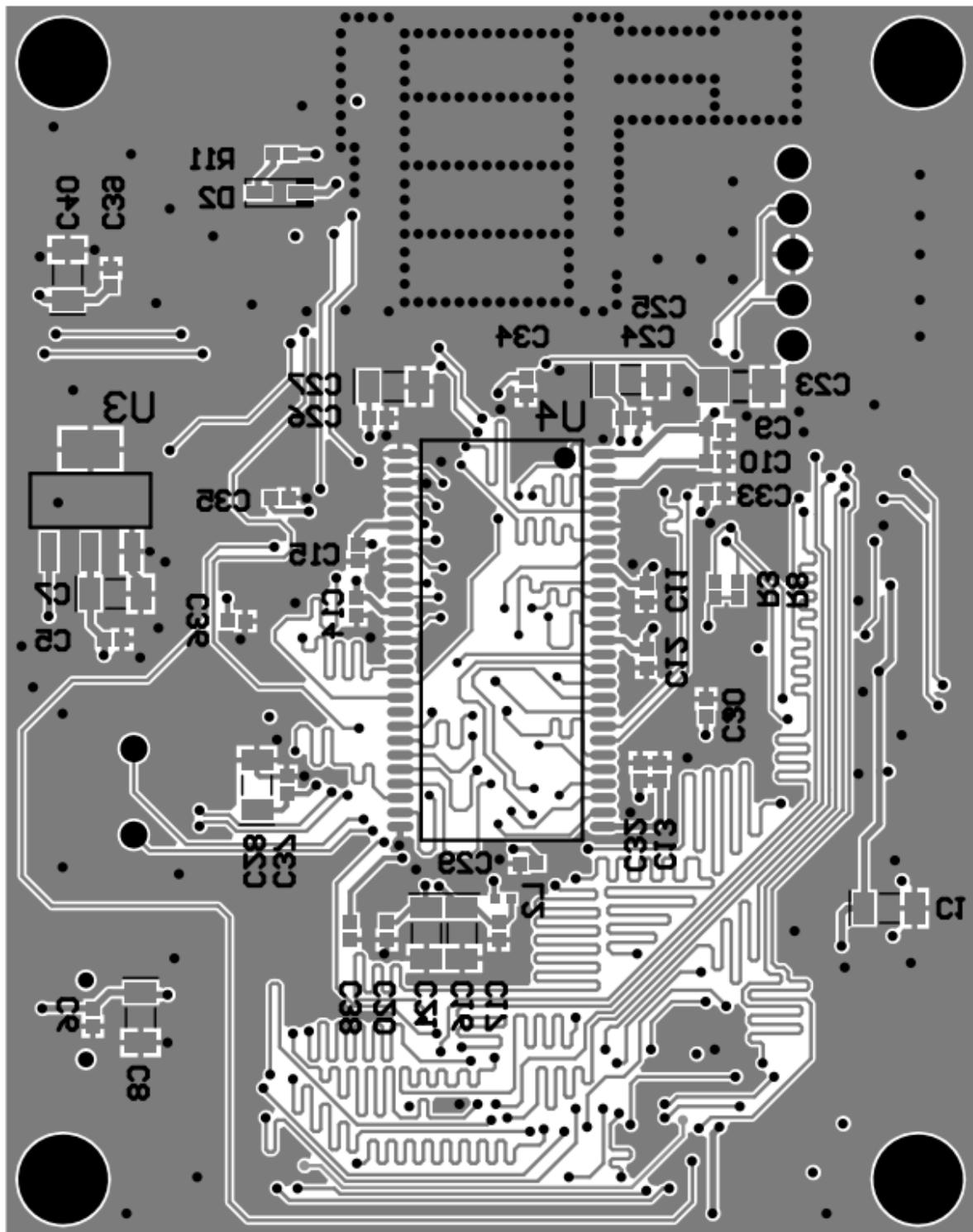
Rysunek 3.13: Układ warstwy 1.



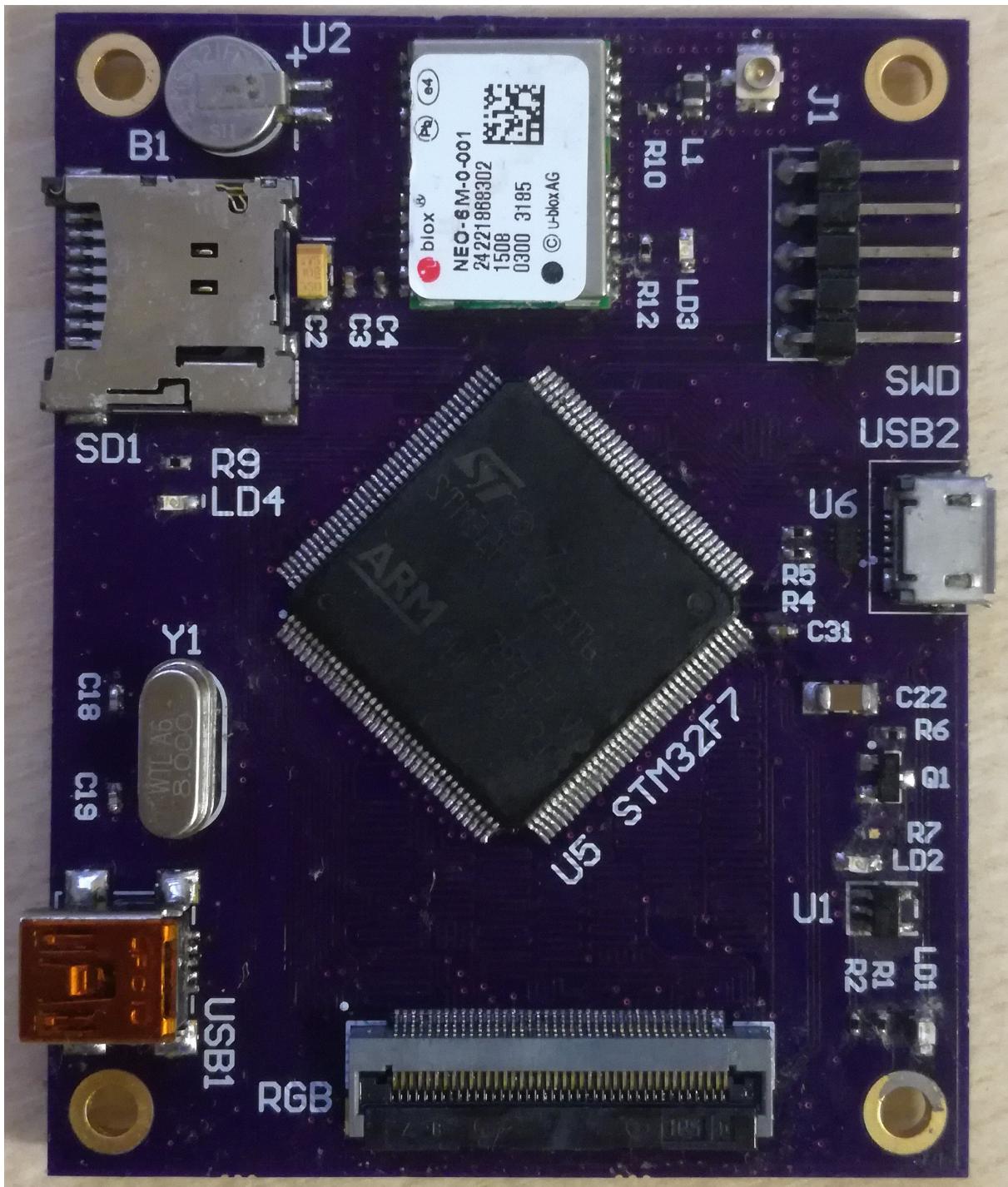
Rysunek 3.14: *Układ warstwy 2.*



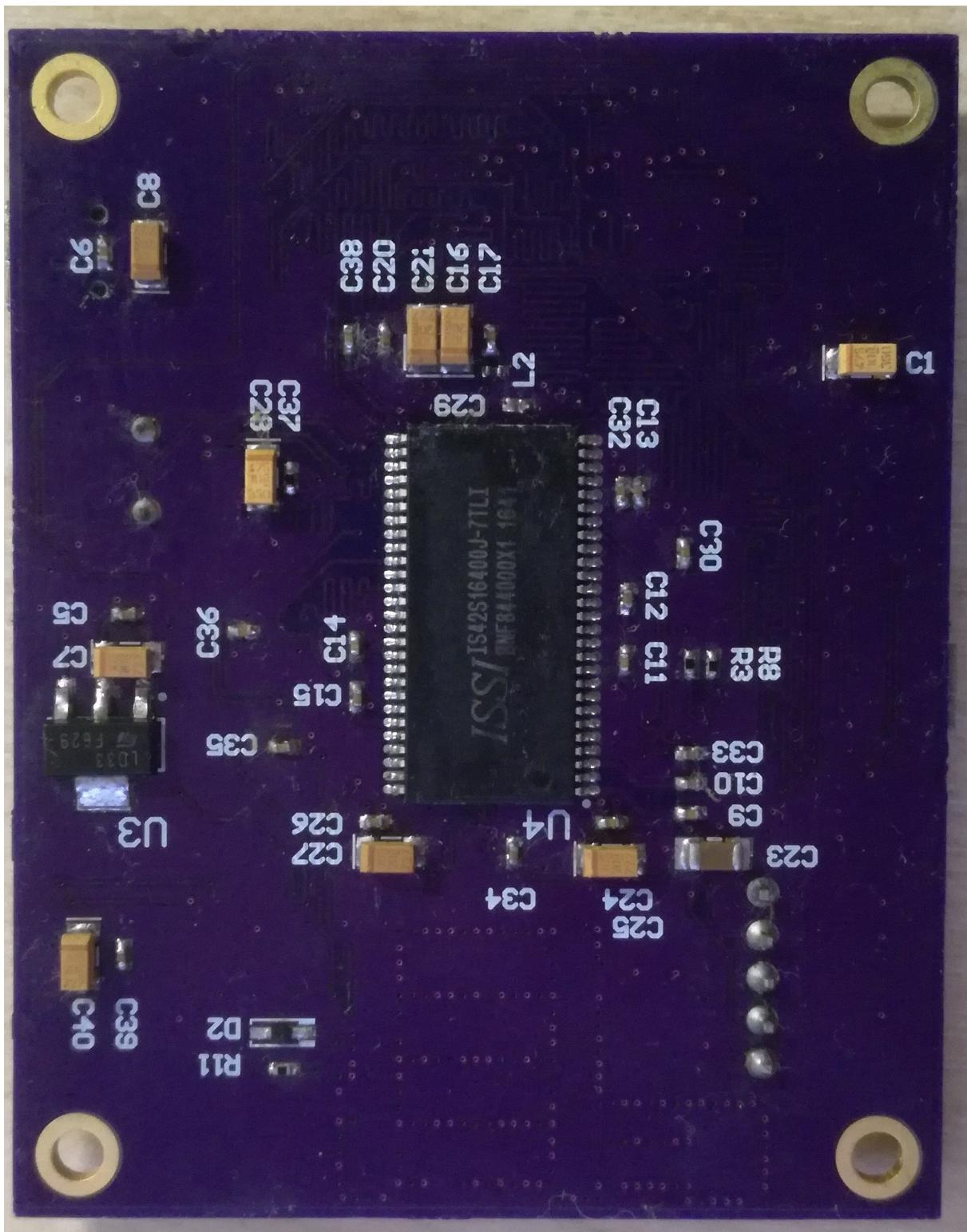
Rysunek 3.15: *Układ warstwy 3.*



Rysunek 3.16: Układ warstwy 4.



Rysunek 3.17: Zdjęcie wierzchu wykonanego PCB.



Rysunek 3.18: Zdjęcie spodu wykonanego PCB.

Rozdział 4

Architektura oprogramowania

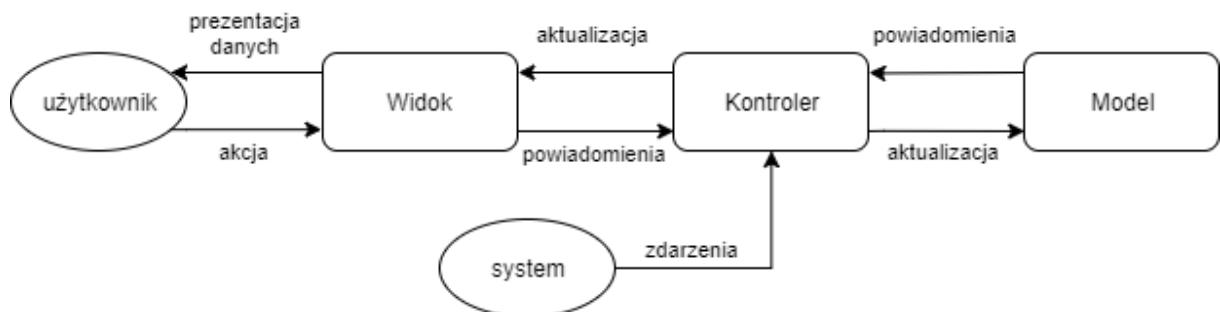
W tym rozdziale opisano strukturę programu wraz z modelem UML oraz opisem funkcjonalności poszczególnych klas. Przedstawiono wykorzystane wzorce projektowe i algorytmy.

4.1 Wzorzec projektowy MVC

Model-Widok-Kontroler, MVC (ang. Model-View-Controller) to wzorzec projektowy wykorzystywany przy tworzeniu aplikacji z interfejsem użytkownika. W MVC można wy różnić trzy zasadnicze części:

- Model - reprezentacja logiki biznesowej (struktura danych oraz wykonywane na nich operacje),
- Widok - warstwa prezentacji danych zawartych w modelu,
- Kontroler - wykonuje operacje na modelu. Zapewnia komunikację pomiędzy widokiem i modelem. Obsługuje żądania użytkownika i systemu, które dotyczą modelu.

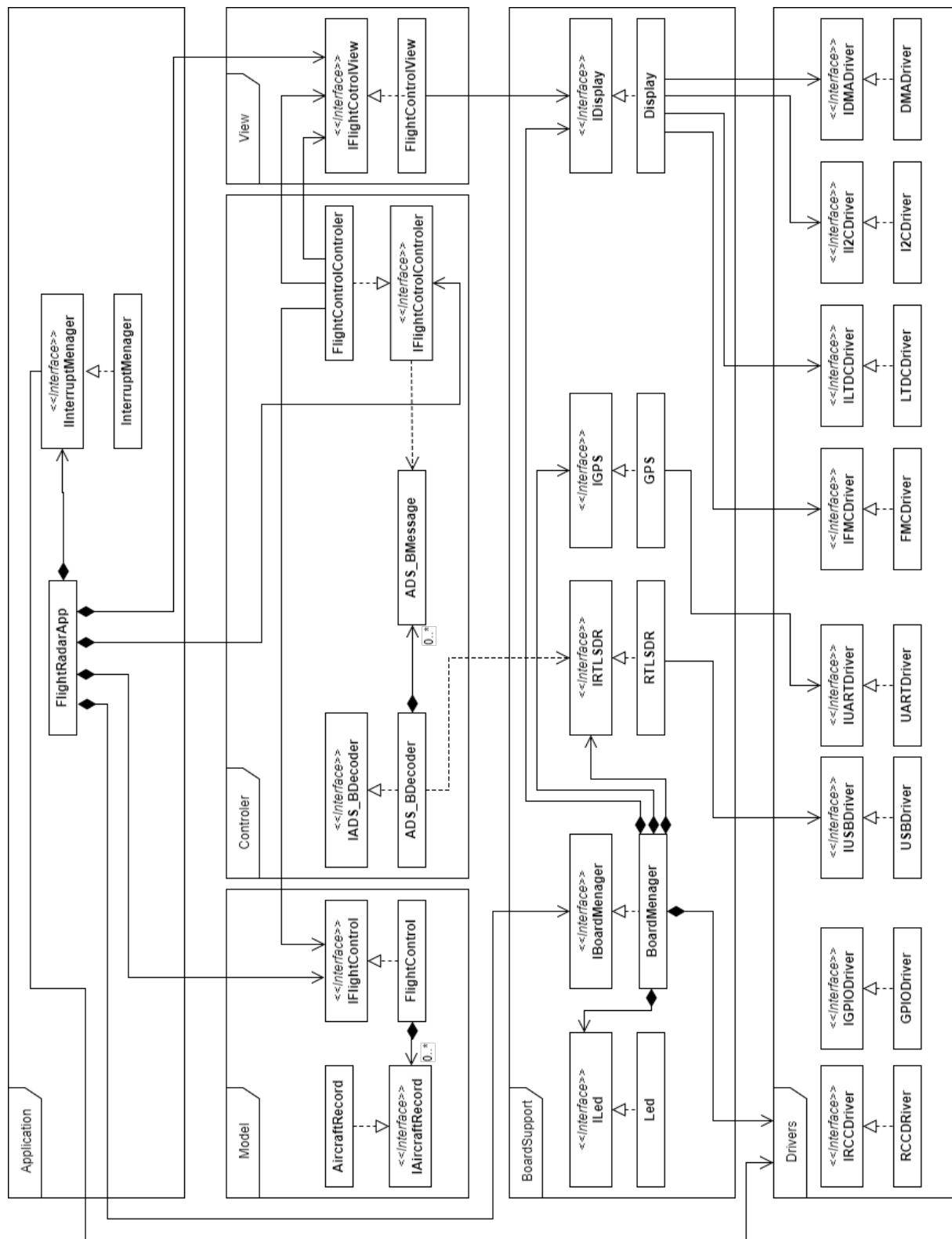
Wzorzec MVC pozwala całkowicie odseparować modelu od widoku, co ułatwia implementację obu części. Ponadto zapewnia przejrzystość projektu grupując klasy we wcześniej wymienione warstwy abstrakcji. Pozwala to ograniczyć liczbę powiązań pomiędzy komponentami. Poniżej przedstawiono diagram interakcji dla wzorca MVC.



Rysunek 4.1: Diagram interakcji pomiędzy komponentami MVC

4.2 Model UML

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



Rysunek 4.2: Diagram klas UML

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 synchronizację pomiędzy systemem operacyjnym i procesami, a odpowiednimi zdarzeniami sygnalizowanymi przez sprzęt.

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

AircraftRecord - struktura do przechowywania informacji o samolotach. Wpisy są zgrupowane w listę. Węzły wyszukuje się według adresu ICAO. Odpowiada modelowi danych z MVC.

FlightControl - implementuję logikę modelu wraz z metodami dostępu i modyfikacji poszczególnych wpisów. Kontroluje czas życia poszczególnych węzłów oraz usuwa te, które uległy przeterminowaniu. Informuje kontroler o zmianach.

Controller - zawiera klasy związane z realizacją kontrolera z MVC.

ADS_BDecoder - klasa odbierająca od obiektu RTLSDR ciągi próbek sygnału radiowego. Odpowiedzialna za wykrywanie ramek i dekodowanie ich do typu ADS_BMessage.

ADS_BMessage - struktura reprezentują odebraną wiadomość. W zależności od typu zawiera różne informacje o samolocie.

FlightControlController - reprezentacja kontrolera w MVC. Klasa jest odpowiedzialna za dodawanie i modyfikowanie wpisów w zależności od odebranych wiadomości. Zajmuje się również przetwarzaniem akcji systemu (przerwania od zegara odliczającego czas życia wpisów) i użytkownika (interakcja z panelem dotykowym). Przekazuje dane z modelu do widoku i informuje o potrzebie odświeżenia interfejsu.

View - przestrzeń zawierająca realizację funkcjonalność widoku z MVC

FlightControlView - klasa jest odpowiedzialna za prezentację danych z modelu oraz przesyłanie akcji użytkownika do kontrolera.

BoardSupport - przestrzeń z wysokopoziomowymi interfejsami do obsługi urządzeń wchodzących w skład PCB.

BoardMenager - klasa zapewniająca komunikację pomiędzy aplikacją, a sprzętem. Jest odpowiedzialna za inicializację wszystkich klas z przestrzeni BoardSupport i Drivers.

Led - interfejs do obsługi LED znajdującej się na PCB.

RTLSDR - klasa implementująca obsługę tunera DVB-T jako SDR korzystając z librtlsdr. Odpowiada za strojenie urządzenia i konfigurację układu próbującego.

GPS - zajmuje 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 klasy zapewniają interfejs do biblioteki HAL odpowiedzialnej za konfigurację i obsługę sprzętu.

RCCDriver - konfigurację 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 podłączonych urządzeń oraz nawiązywanie z nimi połączenia.

UARTDriver - zapewnia obsługę interfejsu UART.

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

LTDCTDriver - obsługa kontrolera wyświetlacza LCD-TFT wraz z konfiguracją warstw.

I2CDriver - klasa do obsługi interfejsu I2C.

DMA Driver - sterownik kontrolera DMA (ang. Direct Memory Access). Przyspiesza operację renderowania kolejnych klatek dla wyświetlacza.

Komunikacja pomiędzy komponentami odbywa się przy pomocy interfejsów. Każda klasa dziedziczy po swoim interfejsie, w którym zadeklarowano wszystkie metody niezbędne do interakcji z obiektem. Pozwala to na odseparowanie rzeczywistej implementacji. Dzięki temu, dany komponent można łatwo zmieniać bez wpływu na oddziałujące z nim elementy. Podejście to ułatwia testowanie oprogramowania, ponieważ wewnętrzna da się 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 umożliwiło podzielenie poszczególne części aplikacji na procesy. Wielozadaniowość jest potrzebna, by jednocześnie obsługiwać interfejs użytkownika, zarządzać akwizycją danych oraz wykonywać obliczenia związane z logiką programu tak, aby użytkownik miał wrażenie płynnego działania aplikacji. Mechanizm semaforów wykorzystano do synchronizacji zadań z przerwaniami sprzętowym, natomiast kolejki pozwoliły na przesyłanie pomiędzy nimi danych i zdarzeń. Ponadto użyto liczników programowych, by odliczać czasu życia poszczególnych wpisów w modelu. Element listy są usuwane usuwany po 120s od otrzymania ostatniej dotyczącej go wiadomości.

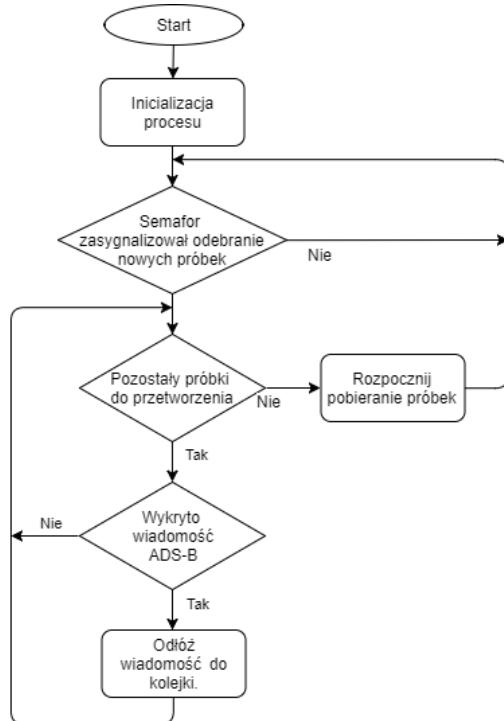
W systemie wydzielono następujące procesy:

- USBMonitorTask - odpowiedzialny za obsługę maszyny stanów USB oraz wykrywanie i konfigurację 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 FlightControlView,
- GUITask - proces obsługujący interfejs graficzny poprzez klasę FlightControlView.

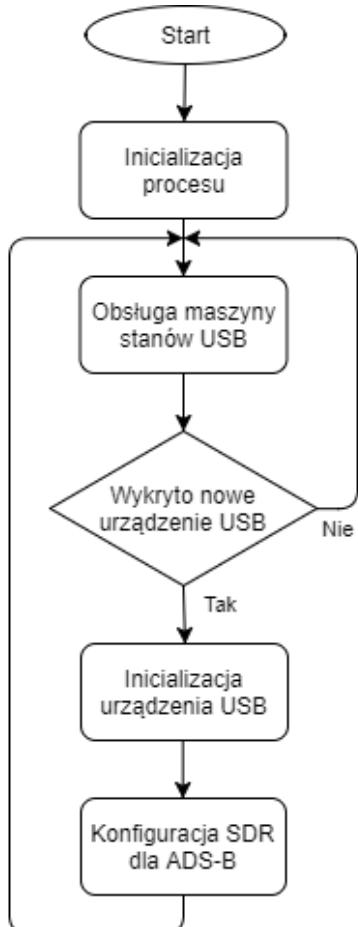
Poniżej przedstawiono schematy blokowe poszczególnych zadań.



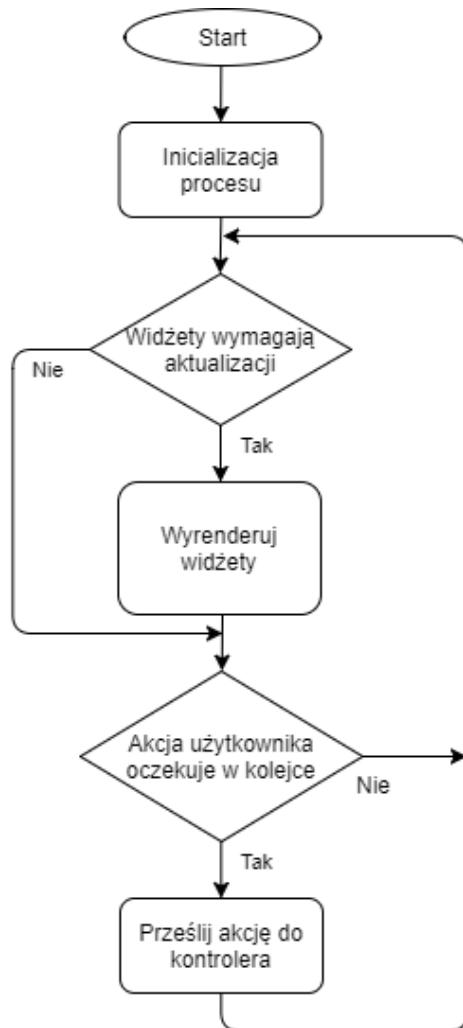
Rysunek 4.3: Schemat blokowy zadania FlightControlerTask



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

Wykrywanie Mode-S

Wiadomości Mode-S mają długość 56 lub 112 bitów w zależności od formatu. Przed wysłaniem dane są poddawane modulacji położenia impulsu z częstotliwością sygnału zegarowego 1MHz. Krótki odstęp pomiędzy stanami wysokimi oznacza logiczne 1 natomiast długi 0. Każdy bit jest kodowany co $1\mu s$. Na początku wiadomości dodawana jest preambuła o długości $8\mu s$ składająca się z czterech impulsów o długości $0.5\mu s$ w odstępach kolejno $0.5\mu s$, $2, 5\mu s$, $0, 5\mu s$. Tak przygotowany pakiet poddaje się konwersji do formatu I/Q i wysyła na częstotliwość 1090MHz.

SDR został skonfigurowany, by wydzielać sygnał z fali nośnej o częstotliwość 1090MHz. Na jego wyjściu otrzymujemy przebieg o częstotliwości $f = 2MHz$, ponieważ poszczególne impulsy trwają $0,5\mu s$. Zgodnie z twierdzeniem Kotelnikowa-Shannona, do poprawnego odtworzenia sygnału potrzebujemy częstotliwości próbkowania co najmniej $2f$. W wykorzystanym tunerze próbki są równolegle pobierane przez dwa 8 bitowe przetworniki ADC.

Przed jednym z układów następuje przesunięcie sygnału w fazie o -90° . Dzięki temu dla częstotliwości próbkowania 2MHz pasmo przenoszenie jest dwukrotnie szersze i również wynosi 2MHz [2]. Dane otrzymane z SDR są w postaci I/Q. Para próbek I/Q stanowi liczbę rzeczywistą, gdzie I to chwilowa amplituda sygnału odebranego przez przetwornik, a Q to chwilowa amplituda sygnału przesuniętego w fazie o -90° względem I. Chwilowa amplituda nadanego sygnału jest modułem liczby zespolonej I/Q obliczonym z następującego wzoru:

$$A = \sqrt{I^2 + Q^2} . \quad (4.1)$$

Aby wykryć pakiet, należy przeliczyć ciąg próbek I/Q na kolejne wartości amplitudy według wzoru (4.1). Następnie znaleźć w sygnale preambułę oraz zdekodować wiadomości przy pomocy demodulacji PPM. Poprawność ramki weryfikuje się poprzez porównanie odebranej i obliczonej sumy CRC.

Dekodowanie ADS-B

ADS-B to wiadomości Mode-S o parametrze Downlink format równym 17. Poniżej przedstawiono strukturę bloku danych ADS-B.

Tabela 4.1: Struktura bloku danych w wiadomości ADS-B

Rozmiar [bit]	Zawartość
5	Downlink format
3	Wersja systemu Mode-S
24	Adres ICAO
5	Typ wiadomości
51	Dane
24	Suma kontrola CRC

W zależności od typu, wiadomość może zawierać następujące informacje.

Tabela 4.2: Typy wiadomości ADS-B

Typ	Zawartość
1 - 4	Identyfikator samolotu i typ transpondera
5 - 8	Pozycja naziemna
9 - 18	Pozycja powietrzna i wysokość barometryczna
9	Prędkość
20-22	Pozycja powietrzna i wysokość GNSS
23-31	Zarezerwowane

Poniżej przedstawiono algorytmy dekodowania wiadomości obsługiwanych przez system.

Identyfikator samolotu (ang. callsign) jest przechowywany w bloku danych jako ciąg ośmiu 6-bitowych indeksów. Poszczególne znaki są odczytywane z przedstawionej tablicy.

```
#ABCDEFGHIJKLMNPQRSTUVWXYZ#####
_#####0123456789#####
```

gdzie:

- `#` - znak nieużywany,
- `_` - spacja.

Tabela 4.3: Struktura pakietu ADS-B typu 1-4

Rozmiar [bit]	3	48
Zawartość	Rodzaj transpondera	Identyfikator

Pozycję statku powietrznego koduję się w formacie CPR (ang. Compact Position Reporting). Wyróżniamy dwie metody dekodowania takich wiadomości. GUP (ang. Globally Unambiguous Position), która potrzebuje dwóch kolejnych wiadomości i LUP (ang. Locally Unambiguous Position), gdzie wymagana jest pozycja referencyjna, np. lokalizacja odbiornika radiowego. Jednak nie może ona być oddalona od statku powietrznego o więcej niż 180Nm. Ze względu na ograniczoną możliwość anteny tunera w projekcie wykorzystano metodę LUP. Poniżej przedstawiono strukturę pakietu ADS-B dla wiadomości z pozycją.

Tabela 4.4: Zawartość wiadomości ADS-B typu 9-18 i 20-22

Rozmiar [bit]	Nazwa	Zawartość
2	SS	Status lotu
1	NIC _b	Flaga dokładności pozycji
12	ALT	Wysokość barometryczna lub GNSS
1	T	Flaga synchronizacji z czasem UTC
1	F	Flaga parzystości ramki
17	Lat _{raw}	Zakodowana szerokość geograficzna w formacie CPR
17	Lon _{raw}	Zakodowana długość geograficzna w formacie CPR

Pozycja w formacie CPR to dwie wartości zmiennopozycyjne w zakresie od 0 do 1, zakodowane jako 17 bitowe liczby całkowite w formacie Q1.17. Konwersja z formatu Q1.17 do CPR odbywa się według następującego wzoru:

$$L_{CPR} = \frac{L_{raw}}{2^{17}} . \quad (4.2)$$

gdzie:

- L_{CPR} - zdekodowana długość lub szerokość geograficzna w formacie CPR,
- L_{raw} - długość lub szerokość geograficzna zakodowana w formacie Q1.17.

Poniżej przedstawiono algorytm dekodowania pozycji samolotu.

Znaczenie użytych symboli:

- NZ - ustalona liczba stref szerokości geograficznych wynosząca 15,
- Lat_{ref} - referencyjna szerokość geograficzna,
- Lon_{ref} - referencyjna długość geograficzna,
- Lat_{CPR} - szerokość geograficzna w formacie CPR,
- Lon_{CPR} - długość geograficzna w formacie CPR,
- Lat - zdekodowano szerokość geograficzną,
- Lon - zdekodowana długość geograficzna.

Definicje użytych funkcji:

$$\text{mod}(x, y) = x - y \left\lfloor \frac{x}{y} \right\rfloor , \quad (4.3)$$

$$NF(Lat) = \left\lfloor \frac{2\pi}{\arccos \left(1 - \frac{1 - \cos(\frac{2\pi}{2NZ})}{\cos^2(\frac{\pi}{180} \cdot Lat)} \right)} \right\rfloor . \quad (4.4)$$

Poniżej przedstawiono kolejne kroki. Najpierw wyznaczano parametr pomocniczy.

$$dLat = \begin{cases} \frac{360}{4NZ} = \frac{360}{60}, & \text{gdy } F = 0 , \\ \frac{360}{4NZ-1} = \frac{360}{59}, & \text{dgy } F = 1 , \end{cases} \quad (4.5)$$

gdzie: F - flaga parzystości wiadomości. Następnie obliczono indeks j z poniższego wzoru:

$$j = \left\lfloor \frac{\text{Lat}_{ref}}{dLat} \right\rfloor + \left\lfloor \frac{\text{mod}(\text{Lat}_{ref}, dLat)}{dLat} - \text{Lat}_{CPR} + \frac{1}{2} \right\rfloor . \quad (4.6)$$

Znalezione wartości pozwoliły na uzyskanie szerokość geograficznej według następującego równania:

$$\text{Lat} = dLat \cdot (j + \text{Lat}_{CPR}) . \quad (4.7)$$

Dalej wyznaczono:

$$dLon = \begin{cases} \frac{360}{NL(Lat)}, & \text{gdy } NL(Lat) > 0, \\ 360, & \text{gdy } NL(Lat) = 0. \end{cases} \quad (4.8)$$

Poniżej przedstawiono wzór na indeks m:

$$m = \left\lfloor \frac{Lon_{ref}}{dLon} \right\rfloor + \left\lfloor \frac{mod(Lon_{ref}, dLon)}{dLon - Lon_{CPR} + \frac{1}{2}} \right\rfloor. \quad (4.9)$$

Parametry m i $dLon$ wykorzystano do obliczenia długości geograficznej według poniższego wzoru.

$$Lon = dLon \cdot (m + Lon_{CPR}). \quad (4.10)$$

Pułap lotu jest zakodowany na polu ALT w wiadomości ADS-B. Zdekodowanie wymaga usunięcia z liczby bitu zawierającego flagę Q. Następnie posłużyono się wzorem:

$$A = \begin{cases} 100N - 1000, & \text{gdy } Q = 0, \\ 25N - 1000, & \text{gdy } Q = 1, \end{cases} \quad (4.11)$$

gdzie:

- N - wartość pola ALT po usunięciu bitu Q,
- A - zdekodowany pułap lotu w stopach.

Wiadomości typu 9 dzielimy na dwa rodzaje. Zawierające prędkością względem powierzchni ziemi (Subtype 1) lub powietrza (Subtype 3). Poniżej przedstawiono format ramki.

Tabela 4.5: *Zawartość wiadomości ADS-B z prędkością w zależności od podtypu.*

Rozmiar [bit]	Nazwa		Zawartość (Subtype 1)	Zawartość (Subtype 3)
1	IC		Flaga zmiany pułapu	
1	RES-A		Rejestr zarezerwowany	
3	NAC		Niepewność prędkości	
1	S_{EW}	S_{Hdg}	Znak prędkości Wschód-Zachód	Flaga dostępności kursu
10	V_{EW}	Hdg	Znak prędkości Wschód-Zachód	Kurs
1	S_{NS}	AS_t	Prędkości Północ-Południe	Typ prędkości powietrznej
10	V_{NS}	AS	Prędkość Wschód-Zachód	Prędkość powietrzna
1	V_{src}		Flaga typu wysokości - barometryczna lub GNSS	
1	S_{Vr}		Znak zmiany pułapu	
1	Vr		Prędkość zmiany pułapu	
2	RES-B		Rejestr zarezerwowany	
1	S_{Diff}		Znak różnicy wysokości barometrycznej i GNSS	
7	Diff		Różnica wysokości barometrycznej i GNSS	

Poniżej przedstawiono algorytmy dekodowania kursu i szybkości w zależności od typu wiadomości.

Znaczenie obliczonych parametrów:

- v - szybkość statku w węzłach,
- h - kurs statku w stopniach.

Dla ADS-B Subtype 1 algorytm wygląda następująco:

$$dV_{EW} = \begin{cases} V_{EW} - 1, & \text{gdy } S_{EW} = 0 , \\ -(V_{EW} - 1), & \text{gdy } S_{EW} = 1 , \end{cases} \quad (4.12)$$

$$dV_{SN} = \begin{cases} V_{SN} - 1, & \text{gdy } S_{SN} = 0 , \\ -(V_{SN} - 1), & \text{gdy } S_{SN} = 1 . \end{cases} \quad (4.13)$$

Następnie można wyznaczyć kurs oraz szybkość statku według poniższych wzorów:

$$v = \sqrt{dV_{EW}^2 + dV_{SN}^2} , \quad (4.14)$$

$$h = \arctan2(dV_{EW}, dV_{SN}) \cdot \frac{180}{\pi} . \quad (4.15)$$

Dla ADS-B Subtype 3 algorytm wygląd następująco:

$$h = \frac{Hdg}{1024} \cdot 360, \text{ gdy } S_{Hdg} = 1 , \quad (4.16)$$

$$v = AS . \quad (4.17)$$

Wiadomość Subtype 3 są wysyłane tylko przez samoloty starszego typu. Większość odbieranych pakietów stanowią Subtype 1.

Obliczanie położenia na płaszczyźnie

Samoloty wysyłają swoją pozycję jako współrzędne geograficzne. Obraz radaru jest płaszczyzną, zatem lokalizacja wymaga transformacji do innego układu przed wyświetleniem. System przedstawia pozycję bazując na odległości oraz kącie pomiędzy urządzeniem, a samolotem. W obliczeniach przyjęto ze ziemia jest kulą. Dystans i namiar pomiędzy punktami na sferze wyznaczono z następujących równań zaczerpniętych z [21]:

$$d = 2R \arcsin \left(\sqrt{\sin^2 \left(\frac{\varphi_2 - \varphi_1}{2} \right) + \cos(\varphi_1) \cos(\varphi_2) \sin^2 \left(\frac{\lambda_2 - \lambda_1}{2} \right)} \right) , \quad (4.18)$$

$$\theta = \arctan2 \begin{pmatrix} \sin(\lambda_2 - \lambda_1) \cos(\varphi_2) , \\ \cos(\varphi_1) \sin(\varphi_2) - \sin(\varphi_1) \cos(\varphi_2) \cos(\lambda_2 - \lambda_1) \end{pmatrix} , \quad (4.19)$$

gdzie:

- d - dystans pomiędzy punktami na sferze w metrach,
- θ - namiar pomiędzy punktami na sferze w radianach,
- φ_1 i λ_1 - szerokość i długość geograficzna odbiornika,
- φ_2 i λ_2 - szerokość i długość geograficzna samolotu.

Otrzymany punkt został przeniesiony do współrzędnych kartezjańskich według poniższej formuły:

$$\begin{aligned} x &= r \sin(\theta) , \\ y &= r \cos(\theta) , \end{aligned} \quad (4.20)$$

gdzie:

- r - dystans obliczony z równania 4.18 w milach nautycznych.

Położenie na obrazie rastrowym otrzymano według następującego algorytmu:

$$\begin{aligned} x_{img} &= x \frac{\text{range}}{\text{size}} + x_{center} , \\ y_{img} &= y \frac{\text{range}}{\text{size}} + y_{center} . \end{aligned} \quad (4.21)$$

gdzie:

- range - promień radaru w milach nautycznych,
- size - promień radaru w pikselach,
- x_{center} - współrzędna x środka radaru na obrazie,
- y_{center} - współrzędna y środka radaru na obrazie.

Rozdział 5

Interfejs użytkownika

W tym rozdziale opisano poszczególne elementy interfejsu użytkownika oraz prezentowane dane.

Opis prezentowanych danych

Interfejs graficzny użytkownika GUI (ang. Graphical User Interface) podzielono na dwie części. Z prawej strony znajduje się lista z wszystkimi wykrytymi samolotami. Poniżej opisano kolejne prezentowane kolumny:

- adres ICAO samolotu,
- pułap lotu w stopach,
- współrzędne geograficzne,
- szybkość w węzłach,
- identyfikator samolotu.

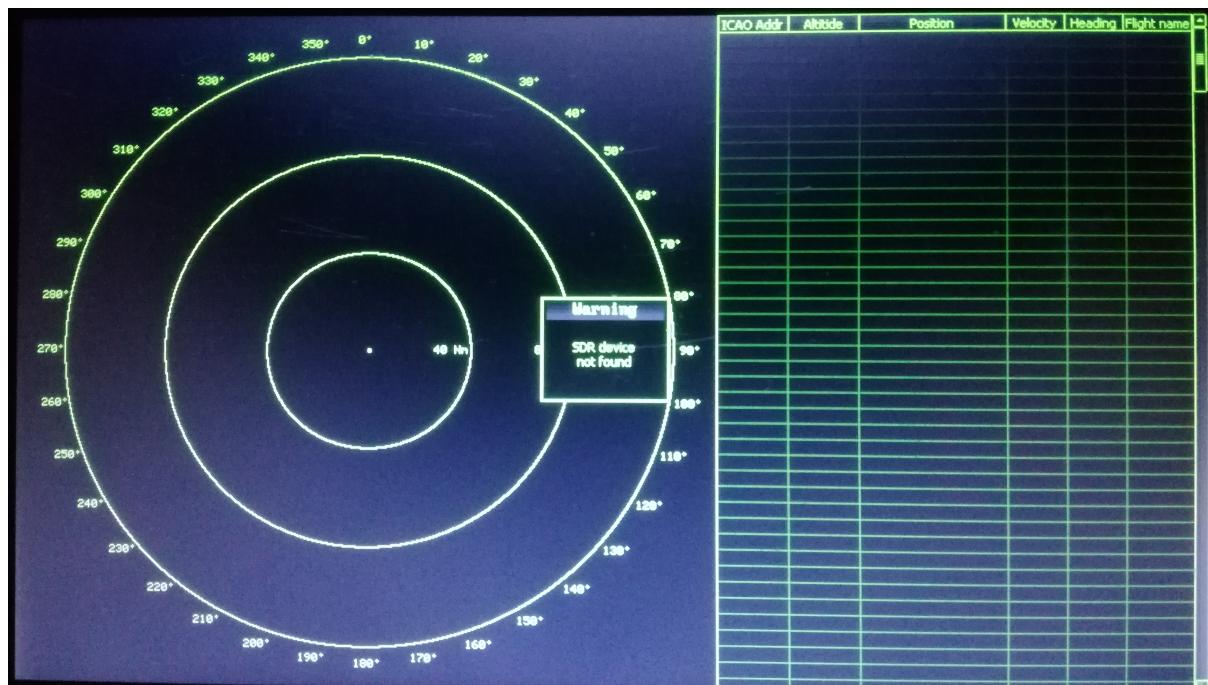
Wpis pojawia się po odebraniu pierwszej wiadomości z danym adresem i jest aktualizowany, gdy kolejne pakiety zostaną odebrane. W przypadku zapełnienia wszystkich widocznych wierszy istnieje możliwość przewijania przy pomocy paska umieszczonego przy prawej krawędzi wyświetlacza.

Po lewej stronie przedstawiono radar pokazujący położenie wykrytych statków powietrznych względem pozycji odbiornika. Kolejne okręgi sygnalizują odległość od urządzenia w milach nautycznych. Wzdłuż największego z nich naniesiono oś określającą kurs w stopniach. Widżet prezentuje następujące informacje o samolocie:

- znacznik położenia,
- adres ICAO pozwalający odnaleźć odpowiedni wiersz na liście,
- linia wskazująca kurs.

W przypadku wykrycia braku urządzenia USB zostanie wyświetlone okienko ze stosowną informacją.

Prezentacja GUI



Rysunek 5.1: Informacja o braku urządzenia SDR



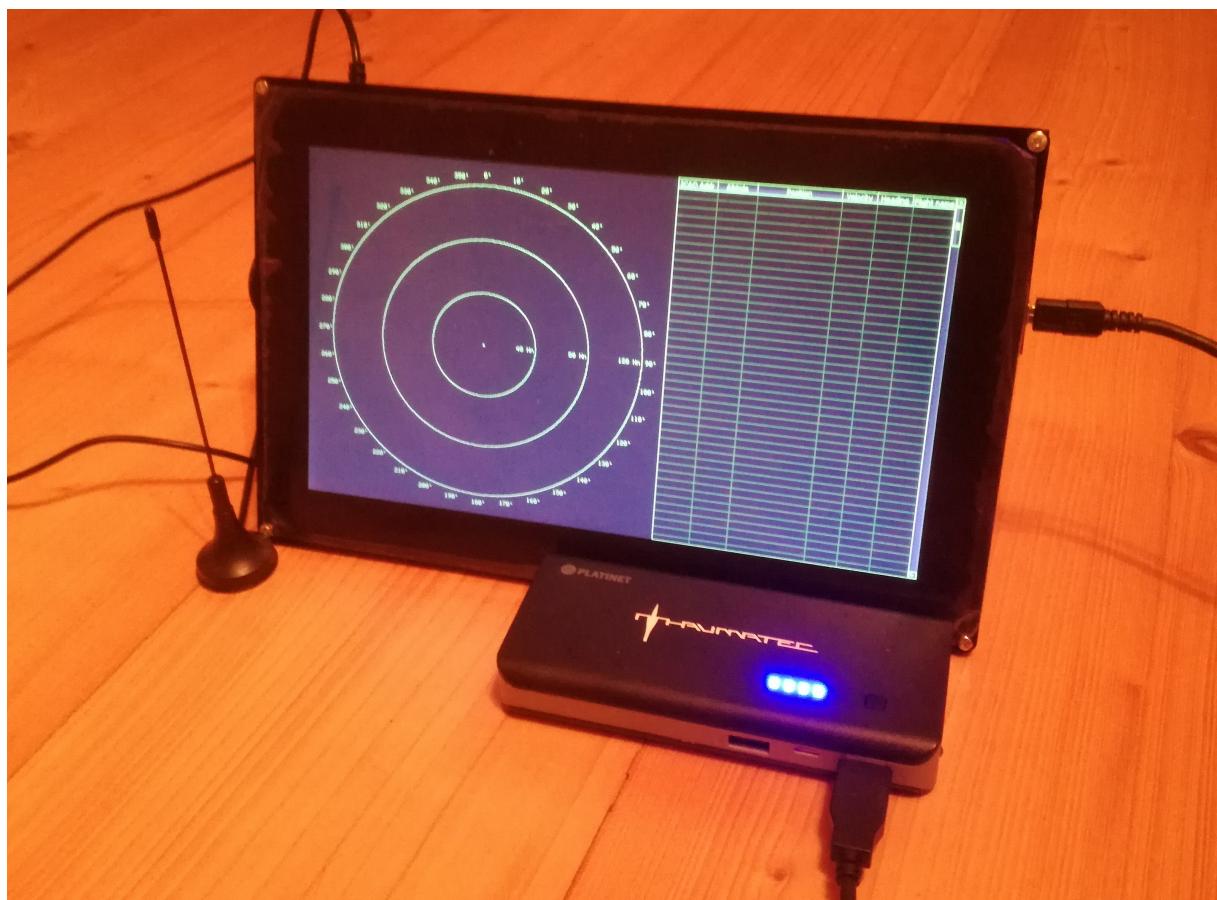
Rysunek 5.2: Prezentacja interfejsu użytkownika

Rozdział 6

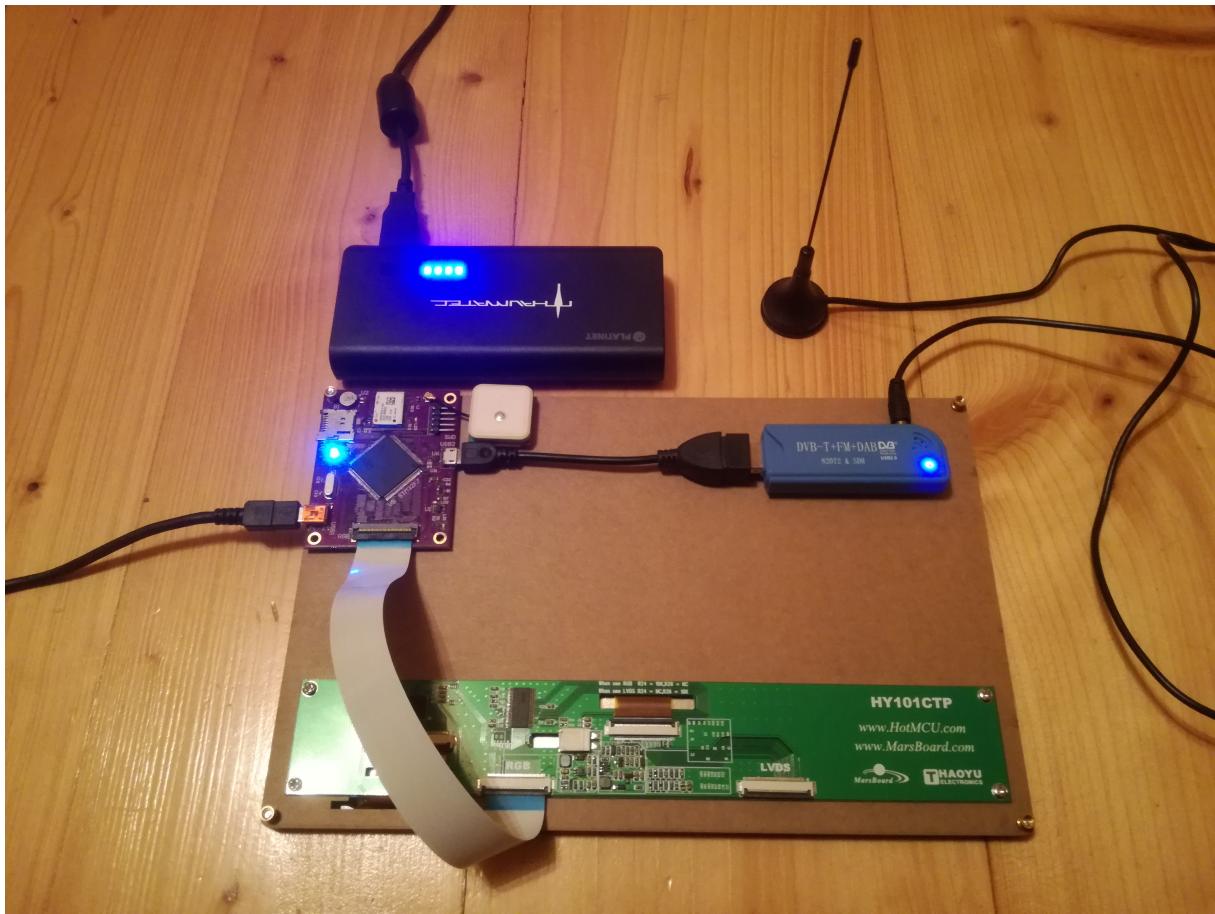
Podsumowanie

W tym rozdziale przedstawiano wyniki projektu oraz porównano działanie systemu z innym rozwiązaniem.

Poniżej przedstawiono zdjęcia gotowego urządzenia. Do zasilania wykorzystano akumulator.



Rysunek 6.1: Zdjęcie gotowego urządzenia



Rysunek 6.2: Zdjęcie tyłu urządzenia

Na rysunku (6.3) zaprezentowano porównanie danych ze strony www.flightradar24.com i prezentowanych przez system.



Rysunek 6.3: Porównanie pozycji samolotów na radarze i mapie



Rysunek 6.4: Porównanie danych zdekodowanych przez urządzenie i ze strony WWW

Można zauważyc, że pozycja na mapie i radarze nieznacznie się różni. Wynika to z formy prezentowanych informacji. Strona wykorzystuje odwzorowanie walcowe równokątne jako sposób przedstawiania globu w postaci płaszczyzny. Radar pokazuje pozycje statku we współrzędnych biegunowych (kąt i odległość). Zdekodowane dane są poprawne i zgadzają się z prezentowanymi na stronie. Występują niewielkie różnice pomiędzy wyznaczonym położeniem, które wynikają z niedokładności interpolacji położenia samolotu pomiędzy kolejnymi wiadomościami. Urządzenie nie wykryło wszystkich statków powietrznych pokazanych na stronie. Przyczyniła się do tego niewielka antena, która dodatkowo jest uniwersalna, a nie zaprojektowana specjalnie dla częstotliwości 1090MHz. Ponadto system *flightradar24* dysponuje większą liczbą odbiorników.

Wykonane PCB działało poprawnie z wszystkimi komponentami systemu. Udało się nawiązać komunikację z tunerem DVB-T poprzez interfejs USB, a następnie wykryć i zdekodować przychodzące wiadomości ADS-B. Poprawność zdekodowanych informacji potwierdzono w innym źródle. GUI poprawnie zaprezentowało wpisy z listy.

Bibliografia

- [1] Stephen H. Hal, Garrett W. Hall, and James A. McCall. *High-Speed Digital System Design - A Handbook of Interconnects Theory and Design Practices*. New York, Chichester, Weinheim, Brisbane, Singapore, Toronto, 2000.
- [2] Richard G. Lyons *Understanding Digital Signal Processing*. Eight Printing, 2001.
- [3] Scott Meyers *Effective Modern C++*. Sebastopol, 2014.
- [4] Brian W. Kernighan, Dennis M. Ritchie *The C Programming Language*. New Jersey, 1988.
- [5] Compaq, Hewlett-Packard, Intel, Lucent, Microsoft, NEC, Philips *Universal Serial Bus Specification*. Revision 2.0, 2000
- [6] Texas Instruments *High-Speed Interface Layout Guidelines*.
<http://www.ti.com/lit/an/spraar7g/spraar7g.pdf>
- [7] *FR408 High Performance Laminate and Prepreg Datasheet*.
<http://docs.oshpark.com/resources/FR408-High-Performance-Laminate-and-Prepreg-Data-Sheet.pdf>
- [8] *LEA-6 / NEO-6 / MAX-6 Hardware Integration Manual*.
https://www.u-blox.com/sites/default/files/products/documents/LEA-NEO-MAX-6_HIM_%28UBX-14054794
- [9] *STM32F76xxx and STM32F77xxx advanced Arm®-based 32-bit MCUs Reference Manual*
<http://www.st.com/en/microcontrollers/stm32f767zi.html>
- [10] *STM32F765xx STM32F767xx STM32F768Ax STM32F769xx Datasheet*
<http://www.st.com/en/microcontrollers/stm32f767zi.html>
- [11] *LCD-TFT display controller (LTDC) on STM32 MCUs*
http://www.st.com/content/ccc/resource/technical/document/application_note/group0/25/ca/f9/
- [12] *THC63LVDM83D 24bit COLOR LVDS TRANSMITTER Datasheet*
http://www.thine.co.jp/files/topics/169_ext_12_0.pdf
- [13] *IS42S16400J, IS45S16400J Datasheet*
<http://www.issi.com/WW/pdf/42-45S16400J.pdf>
- [14] *STM32F746G-DISCO schematics*
<http://www.st.com/en/evaluation-tools/32f746gdiscovery.html>

- [15] *LM1117 800-mA Low-Dropout Linear Regulator Datasheet*
<http://www.ti.com/lit/ds/symlink/lm1117.pdf>
- [16] *STMPS2141, STMPS2151, STMPS2161, STMPS2171 Enhanced single channel power switches*
<http://www.st.com/en/switches-and-multiplexers/stmps2141.html>
- [17] *ECMF02-4CMX8 Common mode filter with ESD protection for USB 2.0 interface*
<http://www.st.com/en/emi-filtering-and-signal-conditioning/ecmf02-4cmx8.html>
- [18] *HY LCD Reference Design*
http://www.hotmcu.com/101-inch-1024x600-tft-lcd-display-with-capacitive-touch-panel-p-215.html?cPath=6_16
- [19] Junzi Sun. *ADS-B Decoding Guide.*
<https://media.readthedocs.org/pdf/adsb-decode-guide/latest/adsb-decode-guide.pdf>
- [20] <http://docs.oshpark.com/services/four-layer/>
- [21] <https://www.movable-type.co.uk/scripts/latlong.html>