TÉCNICO LISBOA

Space Based ADS-B receiver for ISTsat-1

Luís Filipe Brás Pinto Pereira

Thesis to obtain the Master of Science Degree in

Telecommunications and Computer Engineering

Supervisors: Prof. Alberto Manuel Ramos da Cunha
Prof. Rui Manuel Rodrigues Rocha

Examination Committee

Chairperson: Prof. Ricardo Jorge Fernandes Chaves
Supervisor: Prof. Alberto Manuel Ramos da Cunha
Member of the Committee: Dr. Rui António Policarpo Duarte

October 2019

Acknowledgments

A vida é feita de metas e todas elas vão tendo barreiras pelo caminho que devem ser ultrapassadas e nos ajudam a evoluir e a ser melhores. Não posso deixar de agradecer a esta instituição por todo o conhecimento transmitido aos longo dos últimos anos que me fez ultrapassar todas essas barreiras e chegar aqui.

Não posso deixar de agradecer aos meus pais, avós e namorada por tudo o que foram passando comigo e por todos os ensinamentos e apoios que foram dando ao longos destes anos nesta instituição.

Não posso deixar de realçar o grupo N1Z que foi criado entre amigos nesta insituição. Todos eles sabem o que representaram, representam e representarão na minha vida e a eles um muito obrigado pelo acompanhamento. Destacar ainda uma pessoa ainda mais importante para mim neste grupo, João Pinto que foi como um orientador para mim nesta tese e que sem ele teria desistido de realizar este projecto.

Agradecer ainda a todos os meus outros amigos que de uma forma ou de outra contribuiram para o meu desenvolvimento como pessoa mas que acima de tudo me apoiaram ao longo destes anos.

Abstract

A CubeSat is a 10 cm cube low-cost satellite with a mass up to 1.33 kg commonly launched into a Low Earth Orbit (LEO), which extends until an altitude of 2000 Km. The ISTsat-1 is the first nanosatellite developed by the ISTnanosat team. This satellite is intended to test the reliability of an aircraft tracking system namely the Automatic Dependent Surveillance - Broadcast (ADS-B) system, being its scientific payload.

The major requirement for the payload mission is related to the acquisition of signals while decoding ADS-B messages at the same time. This master thesis focuses on the description of the ADS-B messages that are relevant to the algorithm developed in order to receive and decode them. Alongside, the possible architectures to overcome the mission requirements are described with particular attention to the ADS-B algorithm necessary to detect and decode ADS-B messages and the LPC4370 peripherals that will help to receive and sample the respective messages. Therefore, the several results acquired with the final solution are provided and a performance evaluation is given regarding the CPU usage and the power consumption.

Finally, a real test case is provided in order to understand the feasibility of the system in the earth's surface and to guarantee that the system works as required.

Keywords: ISTSat-1, ADS-B, Mode-S message, LPC4370, algorithm.

Resumo

Um CubeSat é um satélite de baixo custo que tem a forma de um cubo de 10 cm de aresta e massa até 1,33 kg, lançado em LEO, que se estende até uma altitude de 2000 km. O ISTsat-1 é o primeiro nanossatélite desenvolvido pela equipa do ISTnanosat e com o qual se pretende testar a confiabilidade de um sistema de rastreamento de aviões, nomeadamente o sistema ADS-B.

O principal requisito para a missão está relacionado com o facto de a aquisição de sinais ter de coexistir com a decodificação de mensagens ADS-B. Esta dissertação de mestrado está focada na descrição das mensagens ADS-B relevantes para o algoritmo desenvolvido, nomeadamente, para as detectar e descodificar. Seguidamente, as possíveis arquiteturas para atingir os requisitos da missão são descritas dando particular ênfase ao algoritmo ADS-B necessário para detectar e decodificar as respectivas mensagens e aos periféricos do microcontrolador LPC4370 que irão ser fundamentais para receber e detectar as respectivas mensagens. Posteriormente, são fornecidos os vários resultados obtidos com a solução final e uma avaliação de desempenho é feita com base na carga de CPU e no consumo de energia.

Em suma, é fornecido um caso de teste real para dar a conhecer a viabilidade do sistema na superfície da Terra e garantir que o sistema funciona conforme desejado.

Palavras-chave: ISTSat-1, ADS-B, LPC4370, Mensagem Mode-S, Algoritmo.

Contents

List of Figures

List of Tables

List of Acronyms

ACAS Airborne Collision Avoidance System

ADCHS High Speed Analog-to-Digital Converter

ADP Aerion Processing and Distribution

ADS-B Automatic Dependent Surveillance - Broadcast

AFSK Audio Frequency-shift Keying

AHB Advanced High-Performance Bus

ALERT Aireon Aircraft Locating and Emergency Response Tracking

ANSP Air Navigation Service Provider

ASR Airport Surveillance Radar

ATC Air Traffic Controller

ATS Air Traffic Service

CA Capability Session

COM Communication Processor Board

CRC Cyclic Redundancy Check

CPR Compact Position Reporting

DF Downlink Format

DLR German Aerospace Center

DMA Direct Memory Access

EC European Commission

EPS Electrical Power System

ESA European Space Agency

FAA Federal Aviation Administration

FIR Flight Information Regions

FIFO First In First Out

FYS Fly Your Satellite!

FPGA Field-programmable Gate Array

GCC GNU Compiler

GNSS Global Navigation Satellite System

GMSK Gaussian Minimum Shift Keying

GPDMA General Purpose Direct Memory Access

GS Ground Station

IAA Irish Aviation Authority

ICAO International Civil Aviation Organization

I2C Inter-integrated Circuit

INCP ISTnanosat Control Protocol

IPC Internal Processor Communication

IST Instituto Superior Técnico

LEO Low Earth Orbit

LLI Linked List Item

NAA National Aviation Authority

OBC On-Board Computer

PL Payload

POD Probability of Detection

POI Probability of Identification

PPM Pulse Position Modulation

PTA Probability of Target Acquisition

PTI Probability of Target Identification

RAM Random Access Memory

RF Radio Frequency

SDP Service Delivery Point

SDR Software Defined Radio

SSR Secondary Surveillance Radar

SGPIO Serial General Purpose I/O

SRAM Static Random Access Memory

US United States

TC Type Code

TCAS Traffic Collision Avoidance System

TPN Teleport Network

TTC Tracking, Telemetry and Control

Chapter 1

Introduction

A CubeSat is a 10 cm cube low-cost satellite with a mass up to 1.33 kg commonly launched into a LEO, which extends until an altitude of 2000 Km. Professors Jordi Puig-Suari from California Polytechnic State University and Bob Twiggs from Stanford University developed the first CubeSat back in 1999, enabling graduate university students to design, build, test, and operate artificial satellites. CubeSats were first deployed by a military organization in 2002 and by a university organization in 2003 [1].

The ISTSat-1 is the first Cubesat developed in Portugal. This project consists of a standard dimension of 10 cm x 10 cm x 10 cm (1U) CubeSat [2] being developed by the ISTnanosat team at Instituto Superior Técnico (IST). This project is developed under the Fly Your Satellite! (FYS) program supervised by the European Space Agency (ESA) Education office, allowing to complement the academic education of students and prepare them for future work on the space sector [3]. The ISTnanosat team is one of the six teams chosen for the second edition of the FYS program.

In the ISTSat-1, the idea was to design and build all subsystems in-house to increase the educational purpose of the FYS program. Moreover, the mission of the ISTsat-1 will serve to demonstrate in orbit the Automatic Dependent Surveillance - Broadcast (ADS-B) technology, where the aircraft tracking data collected and transmitted to the ground shall be analyzed and correlated with ground segment datasets to characterize the ”Cone of Silence” and measure the performance parameters of the receiver chain [4]. Besides, the payload mission was proposed to test this new technology using a new in-house designed receiver and patch antenna whose reception results shall be compared with the ones already acquired by a commercial one.

1.1 Motivation and Goals

Nowadays, important technologies used for detecting aircraft and monitoring the airspace are radars and voice based Air Traffic Service (ATS) systems. However, the ADS-B system is the most studied technology that will improve the monitoring of aircraft not only in remote areas and oceanic routes but also on landing and take-off operations. In the ADS-B system, an aircraft periodically broadcasts its position (gathered via Global Navigation Satellite System (GNSS)), its unique identification (ICAO

address) and other information regarding its speed and altitude, enabling the aircraft to be easily tracked.

The International Civil Aviation Organization believes that ADS-B will eventually become the preferred surveillance technology worldwide in the next few years, as it will be mandatory the use of this system by all aircraft in 2020 [5]. Although new aircraft are already being manufactured respecting the necessary regulations to comply with the ADS-B system, the older ones need to install or upgrade the respective equipment until 2020.

The major disadvantage of using radar technology is the accuracy. Radar accuracy is a measure of the ability of a radar system to determine the correct range, bearing, and height of an object. One of the most significant issues regarding the correct determining of these measures is the atmospheric conditions, as these change the results of a radar accuracy and give wrong feedback to the end-users. Another disadvantage of the radar is related to the price of its equipment. Radar is very expensive to install and maintain, being the installation of this device not feasible in some remote areas like the desert due to the low traffic. However, ADS-B provides a complete coverage of many of those remote areas at a smaller price. Even in remote areas where traffic is relatively high, the price to pay for an ADS-B can be justified in opposition of installing a radar.

Nowadays, if any emergency occurs, the only way to communicate between the aircraft and the control tower is through the voice based ATS system. Also, during ocean routes, pilots are not able to contact any control tower during hours, meaning that the control tower will not have the correct information about the aircraft and can only predict the route of the plane. Actual tracking systems only allow aircraft to fly with a high separation between them to avoid aircraft from crashing into each other. However, if the precise position of the aircraft could be determined, the separation could be diminished, and the density of aircraft could be increased. ADS-B can help on this matter as the update rates are much higher than radar.

In the last couple of years, projects regarding the Space-based ADS-B system have been developed in small satellites to verify the operation of this system which can solve the problem of coverage and availability that current tracking technologies are facing. The ADS-B system works independently, and neither pilots nor controllers have to do any actions. By 2020, all aircraft must have the capability to send ADS-B messages downwards (to the earth surface) and upwards, which enables satellites and other aircraft to receive them. Satellites are much faster to do a roundtrip to the earth than an aircraft travelling in an oceanic route, where aircraft can not find any ground station to downlink messages about their current location, velocity or altitude. With the use of satellites, controllers can have access to recent information of an aircraft in a much smaller time frame. Aireon LLC company[6] is already testing a new solution to provide full coverage with a constellation of 66 satellites providing the first live tracking system for the aviation industry. This new strategy can also provide several advantages to aircraft companies such as low fuel consumption and reduced costs. This system will allow a smaller separation between aircraft, as more planes can share the same wind conditions and travel with the best possible conditions, which can lead to a decrease in the cost of the tickets.

The main goal of this dissertation is the reception of ADS-B messages on the payload system. However, the mission of the ISTsat-1 is to detect, store and downlink the received messages and compare

them to ground segment datasets of messages on the Ground Station (GS). To accomplish this goal, it is important to understand the specific requirements defined by the ISTnanosat team for the payload mission, which are related with the acquisition of signals and decoding of messages.

1.2 Document Structure

This document describes the research and work developed to accomplish a fully functional ADS-B payload, being organized in six chapters.

Chapter 1 introduces this report, with the motivation and objectives of the master thesis.

Chapter 2 describes the ADS-B system, its components, the state of the art of space-based ADS-B and several tracking software.

Chapter 3 details the mission requirements, constraints, and what shall be accomplished, describing the architecture of the solution as well as giving an overview of the different tests that shall be done.

Chapter 4 describes the implementation of the chosen architecture giving particular details regarding the algorithm choices, the peripherals usage and the test bench.

Chapter 5 describes the validation tests that have been done to prove this proposed solution. It also describes an evaluation of the performance of the algorithm responsible for detecting, decoding, and filtering the message as well as the power consumption of the subsystem developed.

Chapter 6 summarizes the developed work and the future one that shall be done in the field.

Chapter 2

State of the Art

The first Radio Detection Ranging (Radar) was discovered in the 19th century by a German Scientist named Heinrich Hertz, where he recognised that radio waves could be read and reflected off metallic objects. This object detection became an essential method in Radar systems. By the 1970s, new technologies especially the discovery of the satellites changed the way people used Radar systems. As more advanced and precise technologies became available, flight tracking itself became more accurate. The constant development of technology is responsible for the appearing of the ADS-B system.

ADS-B is one of the most recent technologies designed to help controllers and pilots in the aviation industry. Europe and United States (US) share now the same deadline date (January 1, 2020) to have all aircraft equipped with the ADS-B system [7].

Along with the usual ADS-B system, several studies and investigations have been developed during the last few years regarding the Space-Based ADS-B system implementation. The ISTSat-1 is intended to be part of the studies and investigations regarding this space-based technology.

2.1 ADS-B System

The ADS-B system was designed to complement the conventional radar, allowing to control aircraft with greater precision and over a substantial percentage of the earth's surface, being also much cheaper to build and implement. Therefore, with the usage of satellites, aircraft transmitters and ground station receivers, the goal of this system is to help both controllers and flight crews and to cover 99% of the earth.

According to [8], ADS-B means:

- Automatic, because position and speed information is automatically transmitted (at least once every second) without flight crew or operator inputs.
- Dependent, because transmission is dependent on proper operation of on-board equipment that determines position and speed.
- Surveillance, because information about aircraft is in continuous observation.

- Broadcast, because the information is broadcasted to any aircraft or ground station equipped with ADS-B receiver.

As illustrated in Figure 2.1, aircraft obtain their position through the GNSS and altitude and velocity from the on-board systems, which are then broadcasted. This feature allows the reception of signals on a satellite, other aircraft or Ground Station. This is only possible because aircraft have one antenna mounted on the bottom of the fuselage and another on the top of the fuselage.

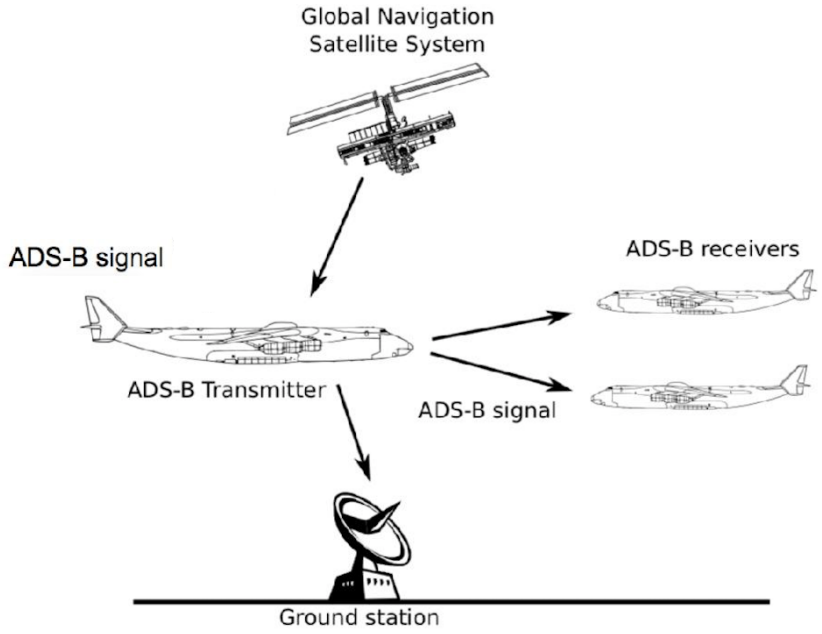

Figure 2.1: ADS-B system overview

2.1.1 ADS-B Components

The ADS-B system has three components: airborne components for ADS-B OUT, airborne components for ADS-B IN and ground components.

ADS-B OUT, represented as the ADS-B Transmitter in figure 2.1, is responsible for broadcasting the speed, altitude, identification and location of the aircraft to other aircraft and ground stations, being its equipment required in all aircraft by the Federal Aviation Administration (FAA) and European Commission (EC). Commonly, a GNSS receiver processes the GNSS satellite signals to produce the aircraft position and velocity that will be broadcasted.

FAA and EC do not currently require ADS-B IN equipment. However, if this equipment is mounted on an aircraft, it will have the ability to provide pilot-usable flight information such as traffic and weather information. This equipment is responsible to receive and process messages and whose messages can come from ADS-B OUT system of another aircraft or a GS. The ADS-IN equipment is represented as the ADS-B Signal in figure 2.1. This information is then passed to the cockpit display of traffic information to update the crew knowledge about nearest aircraft environment and weather conditions.

Ground components include an ADS-B antenna receiver with an unobstructed view towards the horizon responsible to capture the signals emitted by an ADS-B OUT system. This component is responsible for receiving the messages used by the control towers to have more accurate locations of the planes. However, since this method does not allow to cover the entire earth's surface, the space component is being considering as a new ADS-B component for the future.

Space component is intended to help the aviation industry, especially the tracking systems to accomplish approximately 100% coverage of the globe. This component is intended to include an ADS-B IN component as it needs to receive messages from aircraft properly, being usually a big satellite that is responsible for the acquisition of messages in areas where radar and ADS-B ground segments have no coverage. Several investigations on the field will be described in section 2.2, where ISTsat-1 is included as a way to test the reception of ADS-B messages with a small cost.

2.1.2 ADS-B Messages

Every aircraft equipped with the ADS-B OUT system can send messages that are received through the ADS-B IN receiver mounted on the other plane or by a ground segment, needing a rule-compliant ADS-B OUT solution to transmit those messages and a 1090 MHz extended squitter unit to be compliant with international rules. The word squitter refers to a periodic burst or broadcast of aircraft-tracking data that is periodically transmitted without interrogation from the controller's radar. Mode-S is a Secondary Surveillance Radar (SSR) process that allows selective interrogation of aircraft using the 24 bit ICAO address.

This extended squitter is a subsystem of the current Mode-S where a transponder squit only sends the most necessary information about aircraft identification, system status and pressure altitude information. The extended Mode-S allows sending 49 individual parameters compared to the 7 for the normal Mode-S.

The structure of an ADS-B message always has two segments: Preamble and Data block. The former lasts for 8μs and is used to synchronize the receiver with the transmitter, while the latter is used to transport the specific information of an aircraft like its position, speed, altitude and identification. Therefore, ADS-B has a data rate of 1 Mbit/sec.

Figure 2.2 gives an overview of the whole preamble pattern. This preamble has to be matched when the signal is acquired on the receiver to start reconstructing a message correctly.

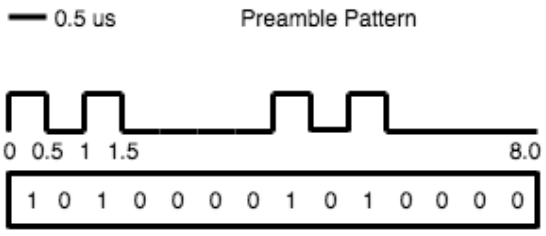

Figure 2.2: Preamble pattern of an ADS-B message

Only the data block is encoded using Pulse Position Modulation (PPM) and Manchester Encoding

while the preamble part only has spikes. The spikes are the ”1’s” on the figure 2.2 as here only the value of the samples matter to determine if its a low(0) or a high(1). The receiver uses Manchester decoding to determine the high and low bits only inside the data block. In this block, in the time allocated for transmitting one bit, the pulse may occupy the earlier (left) or later (right) half of the timeslot encoding either 1 or 0. For a logic ”0”, a transition from 0 to 1 is needed. For a logic ”1”, a transition from a 1 to 0 is needed [9]. Figure 2.3 gives an overview of the reconstruction patterns.

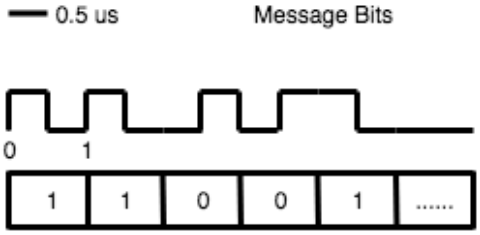

Figure 2.3: ADS-B message with Manchester Code

Each data block contains an ADS-B message with a defined format of 112 bits, as shown in Figure 2.4:

- The Downlink Format (DF) (5 bits) - Identifies the type of message. The correspondent decimal value should be equal to 17 or 18 to identify an ADS-B message. There is no difference between both.
- The Capability Session (CA) (3 bits) - Additional identifier that is intended to give users more information regarding each message of the Mode-S messages. However, for the ADS-B purpose, this field is not relevant.
- The International Civil Aviation Organization (ICAO) (24 bits) - Carries the unique address of the aircraft, being also called as Mode-S ”hex code”, and is part of the Certificate of Registration of a single aircraft. The Certificate of Registration, which is the same as a personal identification card but for aircraft, is maintained by the National Aviation Authority (NAA), a government authority responsible for the approval and regulation of the civil aviation. Some rules should be taken into account for the ICAO addresses where these addresses shall not be changed during a flight, and those with 24 ZEROs and 24 ONEs shall not be assigned to any aircraft as a security measure.
- The DATA (56 bits) - Carries information regarding the position, velocity, location and altitude depending on the message sent by the aircraft.
- The InterrogatorID/Parity (24 bits) - Based on a Cyclic Redundancy Check (CRC) which allows validating the correctness of the received message [10].

To correctly decode the DATA segment, it is necessary to understand what is the type of information carried by each one of the possible messages. Decoding the first 5 bits of the 56 bits of the segment is useful to understand the type of message. Those bits specify the Typecode(TC) of the message. Each different TC is responsible for bringing various information about the aircraft.

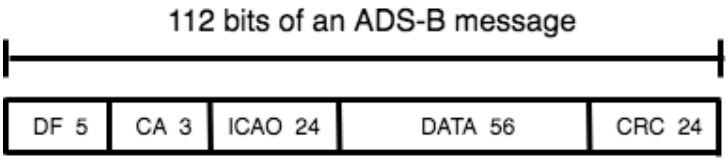

Figure 2.4: Main ADS-B message fields

For each one of those messages, the remaining bits of the DATA segment carries information to calculate or to directly retrieve the values of velocity, location, altitude, speed or identification according to the particular type of message. Annexes describe all the fields carried by the Aircraft Identification, Airborne Position and Airborne Position (w/ GNSS Height).

2.1.3 Broadcast Times

Each ADS-B message is broadcasted with a random period that ranges between 0.4 and 0.6 seconds. This randomization is designed to prevent aircraft from having synchronized transmissions on the same frequency, and thus overlapping each other's communications. However, the overlapping in communications is still usual and can lead to errors, as several aircraft can send their data at the same. For example, if any of the 112 bits could not be correctly decoded and the CRC (last 24 bits of the message) is wrong, the message will not be successfully received.

Table 2.1 shows the broadcast periods for low and high rates of the airborne position, airborne velocity and aircraft identification messages.

Message Type	Low Rate Broadcast Period	High Rate Broadcast Period
Airborne Position	-	0.4s to 0.6s
Airborne Velocity	0.4s to 0.6s	0.4s to 0.6s
Surface Position	4.8s to 5.2s	0.4s to 0.6s
Aircraft Identification	4.8s to 5.2s	9.8s to 10.2s

Table 2.1: Broadcast periods for each message type.

Values for the Low Rate Broadcast Period are used if the navigation source position data has not changed more than 10m in a 30 second sampling interval[11].

2.1.4 Radar vs ADS-B

There are two types of radars: primary and secondary. The former transmit electromagnetic radiation that is reflected by metallic objects in the air allowing to detect aircraft. This system is entirely independent, not requiring any cooperation from aircraft. The latter are based on interrogations and responses. Those radars are intended to question all aircraft in their range and wait for the respective responses.

Severe weather conditions can affect radars and lead to shutting down airports due to the problems of the surveillance systems. This problem can be solved with the reliability of the ADS-B system that is not affected by weather conditions.

With an ADS-B system, it will be possible to improve several aspects compared to the radar system, namely:

- Less separation between aircraft;
- More efficient routes, which leads to fuel savings.
- No need for 2-way communications (request and reply like in the secondary surveillance radars);

ADS-B system will provide less separation between aircraft due to the higher update rates compared to radars. Radars have to wait for a response or a reflected signal, and short-range radars only update once every 4 seconds while slower rotating long-range radars only update once every 13 seconds. ADS-B system messages have much smaller broadcast periods as described in Table 2.2 which will help air traffic controllers and pilots to have faster access to new data.

Airport Surveillance Radar (ASR) that uses radar systems will soon start to use the ADS-B system, even for airport purposes, as it will help even with the grounding aircraft that are on the runway. The smaller update period will also allow aircraft to travel with less separation between them, at least in oceanic routes. It will also allow pilots to share the same routes, enabling companies to save fuel. Actual longitudinal separation is fixed at approximately 50 Nautical Miles or 5 minutes, but with the ADS-B system, this separation can be reduced to about 14 Nautical Miles [12].

In comparison to radar, ADS-B system is entirely independent and do not need any response from the aircraft or traffic controllers. However, placing all these new pieces of equipment, especially in remote areas, may bring problems like vandalism or theft of some material, as occurred in Nigeria in November 2018 [13]. The best solution in the long term is to develop the space component of this system.

2.2 Space-Based ADS-B Systems

Space-Based ADS-B systems are being developed over the last years attempting to fill the gaps of the conventional ADS-B system. The constant evolution in technology is helping to complement the ADS-B system. One of the most significant achievements of this evolution will allow that even if ground stations do not exist to receive the ADS-B signal, air traffic controllers will have access to the correct location of the aircraft. This system will have great utility in remote areas or oceanic routes where there is no available Ground Segment equipment. An overview of some testing campaigns that occurred over the past years is further provided, which are the basis for the Payload (PL) system studied in this report. Furthermore, the first case of a global live tracking ADS-B system is described in section 2.2.3.

2.2.1 DLR

From 2009 to 2013, German Aerospace Center (DLR) and SES Techcom Services developed the first project of ADS-B over Satellite or Space-based ADS-B [14]. The PROBA-V mission was the first in-orbit demonstration of this surveillance system. This demonstration was made in the frame of ESA's PROBA-V mission (PROBA Vegetation [15]) on 23 May 2013, where, in just two hours, at an altitude of 820 km it

recorded more than 12,000 ADS-B and Mode-S messages [16]. Proba-V was the first experiment of this kind with proved feasibility of a Space-based ADS-B system and is responsible for the basis of future developments in this area. A Field-programmable Gate Array (FPGA) with a 32-bit embedded processor was used combining the complete data processing as well as the communications with their On-Board Computer (OBC) [17].

The mission goals of the PROBA-V are the foundations of the ISTsat-1 mission and almost all measures defined by the PROBA-V plan will be used to characterize the ISTsat-1 ADS-B receiver. The PROBA-V measures were:

- Probability of Target Acquisition (PTA): defined as the percentage ratio between the actual number of targets detected and the expected number of targets to be detected within a certain area or time of observation;
- Probability of Detection (POD): defined as the percentage ratio between the actual number of position messages received and the expected number of position messages;
- Probability of Identification (POI): defined as the percentage ratio between the actual number of identification messages received and the expected number of identification messages.
- Probability of Target Identification (PTI): defined as the percentage ratio between the actual number of aircraft identified and the expected number of aircraft which should have been identified.

In all cases, detection means the provision of positional data which will correspond to the reception of airborne position messages. Also, targets mean the detection of aircraft (at least receiving one message from that specific target/aircraft).

2.2.2 GOMSpace

GOMspace [18] is a globally leading designer, integrator and manufacturer of high-end nano-satellites for customers in the academic, government and commercial markets. GOMspace was created in 2007 with roots from Aalborg University, Denmark.

The GOMspace launched the 2kg GOMX-1[19] satellite in late 2013 with a payload for air traffic monitoring from space. Their mission was to demonstrate that ADS-B signals broadcasted by aircraft can be received on a nano-satellite. The conclusion of GOMX-1 Mission says that data reception was perfect and accurate. However, the specific results of the mission are not accessible, so there is no way to compare our results with the Proba-V mission.

In 2015, the GOMX-3[20] was developed in order to update the ADS-B receiver and add a particular capability to the satellite: high-speed data downlink. The new receiver performed very well during the mission collecting ADS-B data globally except in polar areas because of the satellite route. In this mission, the data received by the GOMX-3 payload was downloaded immediately after recording, autonomously post-processed and send to the Flightradar24 platform to display on their flight-tracking website.

Finally, GOMX-4[21] started in 2019 and is their latest mission in two 6U CubeSats whose objective is to use the same ADS-B receiver as GOMX-3 and test the inter-satellite communication at different distances up to 4500km. The focus here is to allow ground segments to have available information as soon as possible because one of the satellites always has a connection to a ground station to transfer the corresponding data.

Nowadays, GOMSpace company has a commercial solution that is available upon request called NanoCom ADS-B using less than 1W of power and built over an FPGA for Space Based ADS-B system to include on a nano-satellite. It can process up to 800 ADS-B messages per second [22].

2.2.3 Iridium

Iridium is set to be one of the future companies regarding real-time tracking ADS-B systems [23]. Iridium and NAV CANADA signed a contract for a joint venture to be run under the company Aireon LLC [6] with support from the US FAA. The ADS-B receiver payloads, to be mounted on each Iridium NEXT satellite, will operate independently and perform the air traffic surveillance function separately from the primary mission of the spacecraft. The Iridium NEXT LEO constellation is the world's largest with 66 operational satellites providing global coverage, system availability, redundancy and flexibility [24]. It is the first system that will not need any specific GS as Aireon company will provide the complete service. From receiving the message in the satellite, downlink it to their Teleport Network (TPN), processing by their Aerion Processing and Distribution (ADP) and deliver the final result to the several Air Navigation Service Provider Service Delivery Point (SDP) systems that have requested them [25]. The Iridium TPN consists of 32 tracking antennas positioned all over the globe and a vast network interconnecting ground stations, operation centre, and satellites.

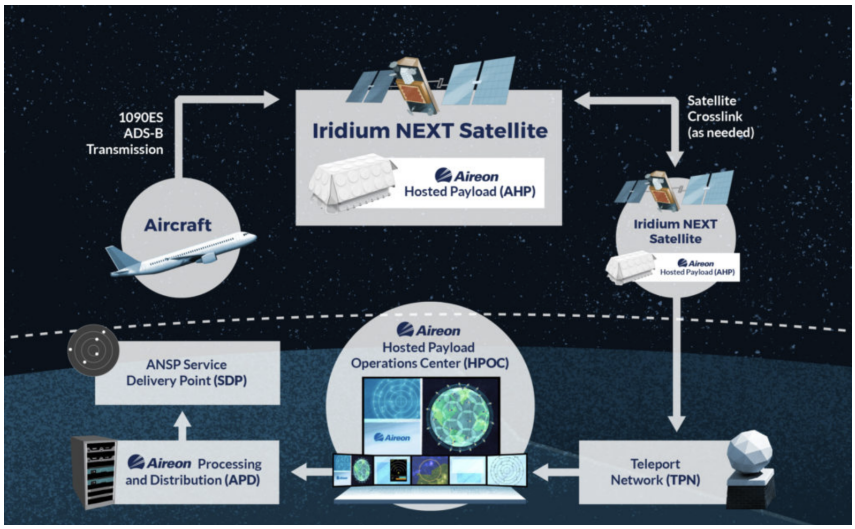

Figure 2.5: Architecture of Space based ADS-B implemented by Aerion company [Extracted from [25]]

This system is operational since 2017 and was created to offer new capabilities and benefits to the global aviation community. January 2019 marked the last launch of satellites to the Iridium NEXT LEO constellation. The 66 satellites are now on their final locations and all operational.

By November 2018, they already had 11 Air Navigation Service Provider's customers. On 25 March 2019 NAV CANADA was the first customer to receive this service on their Edmonton Flight Information Regions (FIR). The first operation was done on 27 March 2019 in the North Atlantic Ocean, which is the world's busiest oceanic airspace, and no results are already available.

Besides, Aerion is offering a free alert system called Aireon Aircraft Locating and Emergency Response Tracking (ALERT) being the only free global emergency aircraft location service available nowadays. According to [26] Irish Aviation Authority (IAA) will be responsible to operate the Aireon ALERT system.

Iridium offers the most exceptional solution so far for a perfect real-time aircraft tracking system. However, the ISTSat-1 is intended to show that a small satellite with a low cost of production can still be very accurate and also receive messages from aircraft. The idea is to compare the results of the ISTSat-1 mission with one of these solutions to prove that our in-house system is cheaper but can have similar results.

2.3 Tracking softwares

Nowadays, there are plenty of choices when thinking about live tracking services for aircraft. Some of them also have free services that are available for all interested people. This is a topic that is gaining new users every day as people are becoming more interested in the aviation industry.

Along the last couple of years, tracking-software has been developed to verify the courses of planes. There are several choices regarding the tracking software to use. However, I will only focus on 3 of them, namely FlightRadar, RadarBox and FlightAware.

FlightRadar [27] is one of the most used software tracking systems. It gives the users the ability to check every detail of the plane and also to filter them by several categories. It provides statistics to users regarding the number of aircraft and how they are acquiring the information from those aircraft. Here it is clear to understand that a significant part of the planes already has the ADS-B system. From the 14,000 planes that are the average number of planes active along the day, 11,000 of them are sharing their information over the ADS-B technology.

AirNav Systems owns the only Worldwide Real-Time ADS-B Flight Position Network. RadarBox [28] is the free or premium service for tracking the aircraft. The available information is almost the same as the FlightRadar.

FlightAware [29] is the last one. It offers the same services as the above tracking systems with an upgrade: the coverage of their operation is shown for a better understanding. The system is fed by other tracking software, namely PlanePlotter, FlightFeeder, PiAware and RadarCape. Figure 2.6 shows the coverage area given by these tracking services.

All this platforms have external collaborators. Those collaborators have a significant role in these tracking systems as they set up several different receivers and help to improve the coverage area of the ADS-B system. This system will be used to check the details of the aircraft and compare with the messages received on the payload system on a testing basis during the development process of the

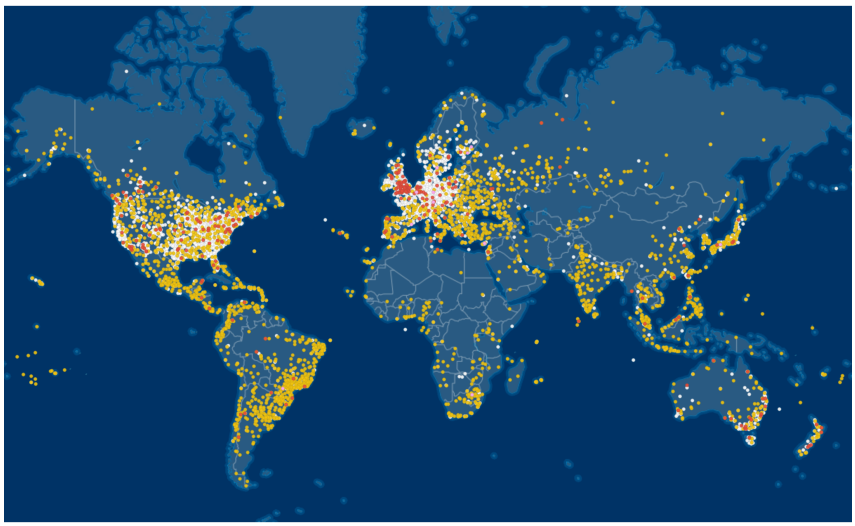

Figure 2.6: Coverage of the FlightAware tracking software system. [Extracted from [30]]

payload.

2.4 ISTSat-1 System Overview

A brief overview of the ISTsat-1 system is done to contextualize the satellite for which the Payload subsystem is developed. The general structure of the ISTsat-1 follows a typical system structure of a CubeSat with four major components: On-Board Computer, Electrical Power System, Tracking, Telemetry and Control and Communication Processor Board. The Payload , described in this report, is added and is related to this CubeSat mission.

Inside the ISTSat-1 all subsystems are interconnected through a common and standard bus system, the PC/104 [31], which allows the stacking of each subsystem board on top of the other.

Payload
OBC
TTC
COM
EPS

Figure 2.7: Structure of the ISTsat-1

In the case of ISTsat-1, the aircraft tracking data collected and transmitted to the ground shall be analyzed and correlated with ground segment datasets to characterize the ”Cone of Silence” and measure the performance parameters of the receiver chain, namely Probability of Target Acquisition (PTA), Probability of Detection (POD) and Probability of Identification (POI) [4].

After acquiring the ADS-B messages, the Payload subsystem shall send these to the COM subsystem, responsible for the satellite data storage. In this subsystem, the messages are compressed

and sent afterwards to the Ground Segment through the Tracking, Telemetry and Control (TTC), when a space link is available. Here, one of the most conditioning factors on the ISTSat-1 is related to the limited bandwidth of communications between the satellite and the GS. Three different data rates may be used: 1.2 kbit/s, 9.6 kbit/s and 48 kbit/s. Depending on the space link and signal quality, these communication bandwidths may affect the performance of the ISTsat-1 mission, as the received ADS-B messages may take longer to be downloaded to the GS.

Additionally, it is essential to characterize the antenna responsible for the capture of ADS-B OUT signals, to demonstrate the ADS-B technology and evaluate the performance of the receiver. The performance parameters will serve to accurately compare the ISTSat-1 mission with other missions already done in the Space-based ADS-B segment. Furthermore, it should allow to identify a possible ”Cone of Silence” on the reception of the ADS-B messages. This problem is related to the antenna pattern, and the way aircraft antennas propagate the signals to the airspace, which produces a null on the z-axis.

Chapter 3

Architecture of the solution

To understand what shall be necessary to create an architecture for the solution, it is mandatory to determine the purpose of the project. Section 3.1 details the mission of this project and the requirements to accomplish the mission goals. After a clear understanding of the mission and the particular requirements, a brief explanation is given about the necessary software to fulfil them.

3.1 Mission

The mission of the ISTsat-1 is to characterise its ADS-B OUT receiver. To accomplish that characterization, it is necessary to collect the messages broadcasted by commercial aircraft. The aircraft data collected in orbit and transmitted to the ground station shall be analysed and correlated with ground-based datasets of messages to demonstrate the ADS-B detection technology. The results of this correlation will confirm if the system is working correctly (reception of signals and software processing of messages).

Aircraft typically use a blade antenna mounted on top of the fuselage and another at the bottom, alternating the transmission between the two antennas, at 1090 MHz. Its linearly polarized radiation pattern generally presents a null along the z-axis which is responsible for a “Cone of Silence” around the aircraft zenith and nadir, which can extend over a range between 60km and 100km at an altitude of 400km. This means that, when a aircraft are between this range, the satellite antenna will not be able to detect messages broadcast by them, which will probably cause interruption on the reception of messages from a specific aircraft during a period. The ISTSat-1 will use an ADS-B antenna with circular polarisation that will be responsible for receiving ADS-B OUT messages sent by the top-mounted antenna of an aircraft. Figure 3.1 represents a ”Cone of Silence” that may occur on the ISTsat-1 mission[32].

The ”Cone of Silence” will be studied in our mission. The idea is to verify in the ground station if a breakdown in communications occurred for a particular aircraft, being necessary to confirm if the plane was passing directly above the satellite afterwards. However, this situation can be tricky since messages can collide and thus leading to a ”fake” time interval of non-received messages. In addition, the time interval of non-received messages shall be higher and constant during a small period.

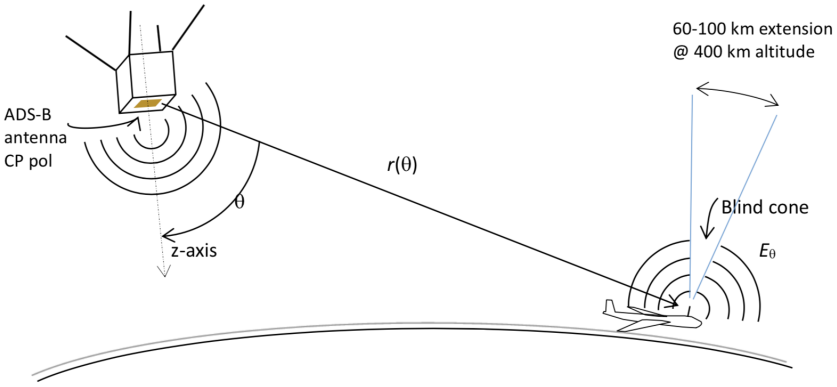

Figure 3.1: Cone of silence representation. Source: [32]

3.1.1 Mission requirements

As described in [33], there are several requirements approved by ESA specialists that need to be met in this mission. It is mandatory to refer to those requirements as they are the basis of the software architecture developed for the Payload system. The most important to refer regarding the PL and mission are:

- Requirement PLS-FKT-180: The ADS-B receiver shall be able to receive samples and decode messages simultaneously;
- Requirement PLS-FKT-150: The ADS-B receiver shall allow filtering the ADS-B messages sent for downlink using filters;
- Requirement PLS-DCP-030: The ADS-B receiver shall be able to communicate with the Communication Processor Board (COM) subsystem through a digital interface;
- Requirement PLS-ITF-110: The generated Message Statistics shall include: number of preambles detected and number of messages decoded;
- Requirement PLS-CTR-360: The ADS-B receiver power consumption shall not exceed 0.8W in Normal mode;
- Requirement MIS-DCP-030: The mission is considering that in a time frame of 10 minutes, it should collect around 40.000 messages from 20 different aircraft.

3.1.2 Goals of the mission

To verify the ISTsat-1 solution regarding the ADS-B receiver, only three of the four parameters defined in PROBA-V mission will be used: the PTA, the POI and the POD. These measures will always be made over the entire mission time. Furthermore, to characterize the receiver chain and understand the performance parameters, it is mandatory to understand the messages exchanged, the respective formats and their broadcast periods as described in chapter 2.

In the Annexes, the three most significant ADS-B messages indispensable for the mission are described in detail, allowing to understand that filtering will play a key role when obtaining the correct PTA, POD and POI measures. The following expressions define those measures:

$$PTA = \frac{\text{Actual number of targets detected}}{\text{Expected number of targets to be detected}} * 100$$

Targets detected means the reception of a single message with the correspondent ICAO address (unique ID of aircraft).

$$POD = \frac{\text{Actual number of position messages received}}{\text{Expected number of positions messages to be received}} * 100$$

Position messages mean the reception of ADS-B messages with a TC decimal value between 9 and 18 or between 20 and 22.

$$POI = \frac{\text{Actual number of identification messages received}}{\text{Expected number of identification messages to be received}} * 100$$

Identification messages mean the reception of ADS-B messages with a TC decimal value between 1 and 4.

These measures will allow understanding the accuracy of the ISTsat-1 ADS-B receptor and compare with other successful missions. However, the main goal is to make sure the mission can be carried out in the payload as this subsystem needs to host a signal processing and software to decode messages and both processes shall be running at the same time.

3.1.3 Evaluation

The receiver that needs to be done for the acquisition of samples shall work at 400km of altitude. However, it's not easy to have a testing facility capable of simulating the same signals like the ones expected to receive at that altitude. Also, the bandwidth of downlink can be a problem if the satellite is unable to properly communicate with ground stations, as it will not be possible to have access to the mission data.

If everything works as expected, there should be mission results available on the GS, meaning that software capable of processing samples must be created to fulfil the mission requirements. The evaluation of the mission will be determined by the performance parameters already described above. However, to properly evaluate the mission parameters, it will be necessary to correctly choose an area where our receiver can handle the messages. If a defined region is not set, the receiver can have problems receiving the signals and can even saturate. In these conditions, it will be complicated to compare results with ground station datasets and thus verifying the correct reception of messages.

Finally, the final evaluation will take place during the mission. However, other performance tests can be done on earth's surface that shall give an idea of what is expected to receive and detect at 400km of altitude, like placing the antenna directly under a route of aircraft and check if there is a time interval of non-received messages.

3.2 Architecture

The proposed architecture of the subsystem relates to the mission requirements described in section 3.1.1. Figure 3.2 describes in detail the proposed solution, as a description of each block inside this architecture is detailed further.

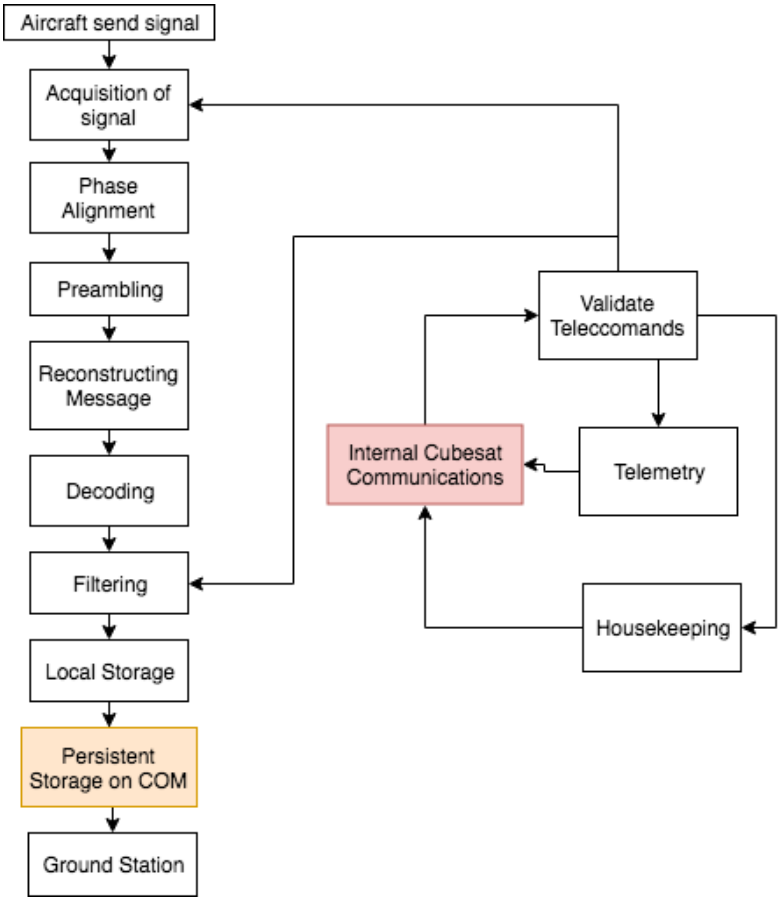

Figure 3.2: Functions representation to accomplish the mission

The left part of this architecture is called the receiver chain, which is intended to be a straightforward process. The process always starts by detecting signals from a particular aircraft through the hardware ADS-B RF frontend. After that, all the functions of the receiver chain are software-based and will allow the system to detect, decode and filter the correct messages. The right side of the figure represents other small functions necessary for the Housekeeping, Telemetry and command validation. In the middle, the coloured block demonstrates the internal communications done inside the ISTSat-1. The following sections describe in detail each block, giving particular attention to what are their roles inside this architecture.

3.3 Acquisition of Signal

The signal acquisition block is responsible for the reception of ADS-B signals and converts them into samples. The antenna mounted on the ISTSat-1 is responsible for capturing the ADS-B OUT signals

sent from the aircraft. The receiver then uses a 12-bit ADCHS that is responsible to convert into samples, which digital values corresponding to a specific voltage in a defined bit range. After this conversion, made only recurring to hardware, the software will start to be fully used to process each one of the samples. As described earlier, ADS-B messages are encoded using Manchester code with a bit rate of 1 Mbit/s. This must be taken into account when choosing the sampling rate necessary as Manchester code needs an oversampling for the decoding process. The software will allow to clearly define the sampling rate, which will be used to detect the high and low samples that are specific characteristic of the Manchester code.

3.4 Preambling and phase alignment

The phase alignment is intended to align the start of the preamble, being responsible of identifying the first sample of the preamble sequence so that no propagation errors occur. The preamble is a distinctive pattern or sequence of pulses that can be easily detected, allowing the receiver to unambiguously determine the position of the high and low bits within a message (synchronization method). As described in section 3.3, high and low samples created by the ADCHS are used for preamble detection purposes.

The number of samples to use in this step is related to the sampling rate. For a better understanding, a sampling rate of 12 MSamples/s is assumed, where each preamble step occupies half the time bit, meaning that only six samples are used to determine each step. With a small number of samples, determining the preamble spikes is much more difficult as a lot of errors appear, as consecutive samples may have energy from the previous ones. Therefore, better results are obtained with higher sampling rates. In addition, the preamble does not use the Manchester code, which means that for a transition, it is necessary to compare each step value of six sample values. The objective of the preambling phase is thus to verify the several spikes represented as ”1” in the preamble pattern, as this verification will correspond only on the difference between consecutive samples.

Along with this, a variable is used to maintain the number of preambles detected that will correspond to all Mode-S messages received. ADS-B messages will be a reduced number of all this preamble messages detected since all Mode-S messages share the same preamble pattern.

3.5 Reconstructing Message

The following stage starts the reconstruction of the ADS-B message, as the transmitter and receiver are correctly aligned due to the preamble pattern. The samples acquired in the acquisition phase are again used to reconstruct each bit of the message, which are encoded in Manchester code. The Manchester code for each bit creation always displays a transition on the bit time in one of two possible ways: from high to low or from low to high. The transition occurs in the middle of the bit time. Through this method, the number of samples to evaluate is always equal to the Sampling rate divided by the message bit rate. For example, at 12 MSamples/s, it is necessary to analyse 12 samples to construct one bit. It is possible to reconstruct each bit in two different cases. The first one is to have the first six samples at a high level

and the next six at a low level, creating a bit ”0”, and the other case is to have the first six samples at a low level and the next six at a high level, creating a bit ”1”. This process is repeatedly used until the 112 bits of the message are reconstructed.

3.6 Decoding Message

The message decoding is mainly based on the dump1090 code, which is an OpenSource code available to all people interested in the reception of ADS-B messages and other Mode-S messages[34]. For the purpose of this work, only the specific decoder for the ADS-B messages is considered, as other Mode-S messages are not relevant for this mission. This code allows any user to use a simple Software Defined Radio (SDR) connected to their computer to instantly start seeing live messages sent by the aircraft on a terminal or web page.

At this step and according to the type of ADS-B message, every 112 bits are decoded. Here it is essential to verify if the reconstructed message has a good CRC, meaning that a correct message has been received.

A couple of statistical measures regarding the number of messages decoded with good CRC and the number of messages decoded shall be maintained during this process.

3.7 Message Filtering

Filtering will play a vital role to restrict the number of messages stored on the COM subsystem. They will also be essential to calculate the different measures of the mission accurately. A couple of filters have been defined for this purpose on the payload, namely:

- Filter by Location, used to create a virtual rectangular surface area where it is supposed to accept only the messages coming from aircraft inside that area. This filter is one of the most important and should always be used to reduce the number of messages sent to the GS. The mission is supposed to be done inside a virtual rectangle with an average of 20 aircraft at the same time, where A total of 40.000 messages is expected to be received using this filter in a 10-minute mission.

- Filter by Time, used to accept messages between a defined start and end time. This filter is essential to compare the messages with ground segment datasets, as it is crucial to verify these datasets to understand if the messages received on the ISTsat-1 are correct.

- Filter by Altitude, used to accept messages above or under an altitude defined in the filter. This can be useful to determine the vertical separation between aircraft sharing the same routes, like oceanic routes.

- Filter by Velocity, used to accept messages above or under a velocity defined in the filter. This can be useful to determine planes approaching an airport while landing, or during their take-off manoeuvre.

- Filter by ICAO, used to accept messages from a particular ICAO. This will be very useful to determine a possible ”Cone of Silence” that can occur with our antenna, as explained before.
- Filter by Type Code Number, used to accept messages with a specific Type Code. This filter is required to identify the different messages and thus to complete the performance measures.
- Filter by Aircraft Identification, used most of the times with the Type Code Number filter, being another way to retrieve only messages that bring the identification of the aircraft. This will allow to determine the actual identification of an aircraft and follow its route.

Payload software must handle with one or multiple active filters at the same time. The active filters will always be available for a particular mission, so that the mission parameters described in section 3.1.1. can be properly updated. Also, some filters are not strictly connected to the mission measures but will provide some extra testings on the reception of the different messages.

3.8 Internal Cubesat Communications

For the communications between two subsystems, the ISTnanosat Control Protocol (INCP) shall be used to correctly format the messages exchanged by the subsystems [35]. Internally, between subsystems, some fields of the INCP protocol shall be used along with the I2C protocol.

In the send function, an INCP encoding and an I2C frame formatting to send the information from the payload to COM or OBC shall be done. The send function will be executed whenever the payload needs to respond to a command, report an error or transmit the mission data.

The inverse process shall be done in the receiver function while receiving new messages from those two subsystems. The I2C handler can check the integrity and reliability of the message.

In both functions, statistics regarding I2C messages transmitted and I2C messages received will be done during the entire time payload is active.

3.9 Telecomands

Internally, some telecomands have been defined for the payload subsystem. This function will be responsible of dealing with all the cases defined and trigger the respective action. A list of the different telecomands that shall be available is given:

- A command to setup each one of the filters;
- A command to reset filters;
- A command to activate or deactivate a filter;
- A command to get the filters list;
- A command to reset statistics;

- A command to enable or disable the Housekeeping and Telemetry.

This will be essential to receive orders from other subsystems and trigger one of the desired actions without any mistake.

3.10 Telemetry

Telemetry function is going to collect private information regarding the system and the mission. This information shall then be sent to the OBC subsystem after being requested. A list of the telemetries that shall be collected inside the payload is given:

- Number of active filters;
- Total number of filters;
- Number of decoded messages;
- Number of detected preambles;
- Last ADS-B message received;
- Number of I2C messages transmitted;
- Number of I2C messages received;

3.11 Housekeeping

The housekeeping block must be triggered in a defined interval of time to understand the operation of the different functions. This functions will be responsible for testing each small part of the software of the payload providing information about every process running on the Payload system. Receiver chain shall be verified by inserting a good preamble followed by a message and understand if it can process the preamble, reconstruct the message, decode and filter. This information shall be stored on a report that will be sent downwards to the Ground Station.

Chapter 4

System Implementation

The purpose of this chapter is to address all the implementation details. Starting with the solutions proposed to contain the architecture described in section 3.2 and moving on to the functions inside the system, the algorithm for the detection and reconstruction of messages and finally the testing bench. The proposed implementation summarizes the different approaches and all the development processes done during the implementation of this system.

4.1 Design of the solution

At the microprocessor level, there was no flexibility to choose a microprocessor. The proposed solution for the Payload and sent to the ESA included the use of the LPC4370 microprocessor. So, the base of this design solution is linked to this microprocessor.

LPC4370 is an ARM Cortex-M4 based microcontroller for embedded applications which include an ARM Cortex-M0 co-processor and an ARM Cortex-M0 subsystem. In this microcontroller, all cores can manage peripherals such as ADCHS, GPDMA and Serial General Purpose I/O (SGPIO). Additionally, 64kB of ROM can contain boot code and on-chip software drivers.

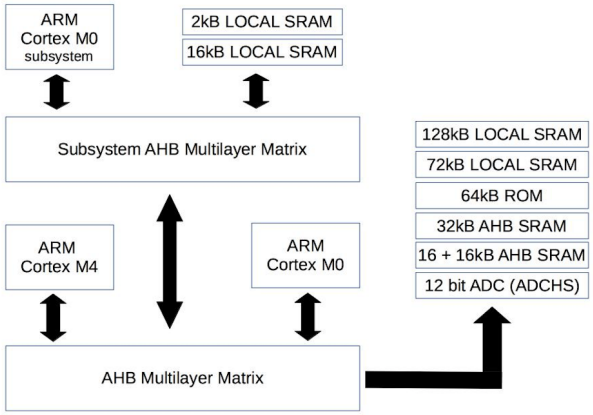

Figure 4.1: LPC4370 memory and processors

Figure 4.1 gives an overview of the interactions between processors and shared memory. A triple

core architecture can be found on this microprocessor. Besides, it is clear to understand that ARM Cortex M0 and ARM Cortex M4 share the same AHB matrix, which means that if both access a memory address through that matrix, the access time of both is going to be the same. However, with the ARM Cortex M0 subsystem, this will not occur because of the subsystem AHB multilayer matrix as this introduces an access latency when crossing to the Main AHB domain. This is a fundamental question when thinking about the location of the different functions necessary, especially to detect and decode the messages.

After the clear understanding of the triple-core processor and the different access times to the peripherals from all of them, the first thought solution is described in figure 4.2.

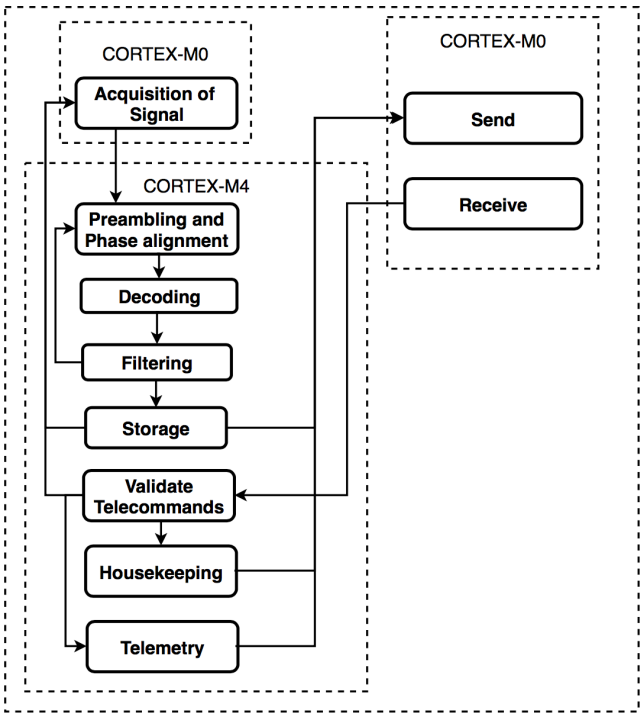

Figure 4.2: First software design iteration for Payload

In this first solution, the idea is that the Cortex-M0 only contains the signal acquisition function, which corresponds to the acquisition of the signals and thus creating the samples. Upon the reception of samples inside the Cortex-M0, they must be used by the Cortex-M4 to attempt to detect, reconstruct, decode and filter the different messages. On the other hand, the Cortex-M0 SUB deals with internal communications with the Cubesat. After the reception of any command message, it should be directed to the validate telecommand function, that will be responsible for triggering the respective action within the payload after the interpretation of the content of the command message. The Cortex-M4 must handle the functions of telemetry and housekeeping that will only be called when receiving a specific command message that will trigger them. In this solution the idea was to use one of the smaller cores to give more processing time to the ARM Cortex-M4 as it could not have time to detect and decode the ADS-B messages.

To test this first solution, the most crucial feature to verify is the maximum sampling rate to receive several signals from aircraft. The idea was to have a minimum fixed sampling rate at 10M Samples/s.

The test scenario was to have the ARM Cortex-M0 sampling while the ARM Cortex-M4 will handle will all the logical part. However, the results were very far from what was expected as the test result only allowed the sampling rate to reach the 3M Samples/s, which was not close to the fixed minimum sampling rate required. The ARM Cortex-M0 can't handle with the required amount of data and thus the solution was to implement the sampling over GPDMA.

This microprocessor has the ability to select which cores are active (Power on) or inactive (Power Off). The M4 is still needed as the main core while the other two processors can be powered off or powered on by the main core. This feature allows to set up a second approach where the ARM Cortex-M0 that is connected to the main Advanced High-Performance Bus (AHB) matrix is powered off. The decision to power off this ARM Cortex-M0 is related to the fact that it can be sharing the same buses as the ARM Cortex-M4 and thus leading to longer access times in vital functions of this system such as the signal acquisition function. Figure 4.3 illustrates the final distribution of functions inside the LPC4370 multicore system without one of the ARM Cortex-M0.

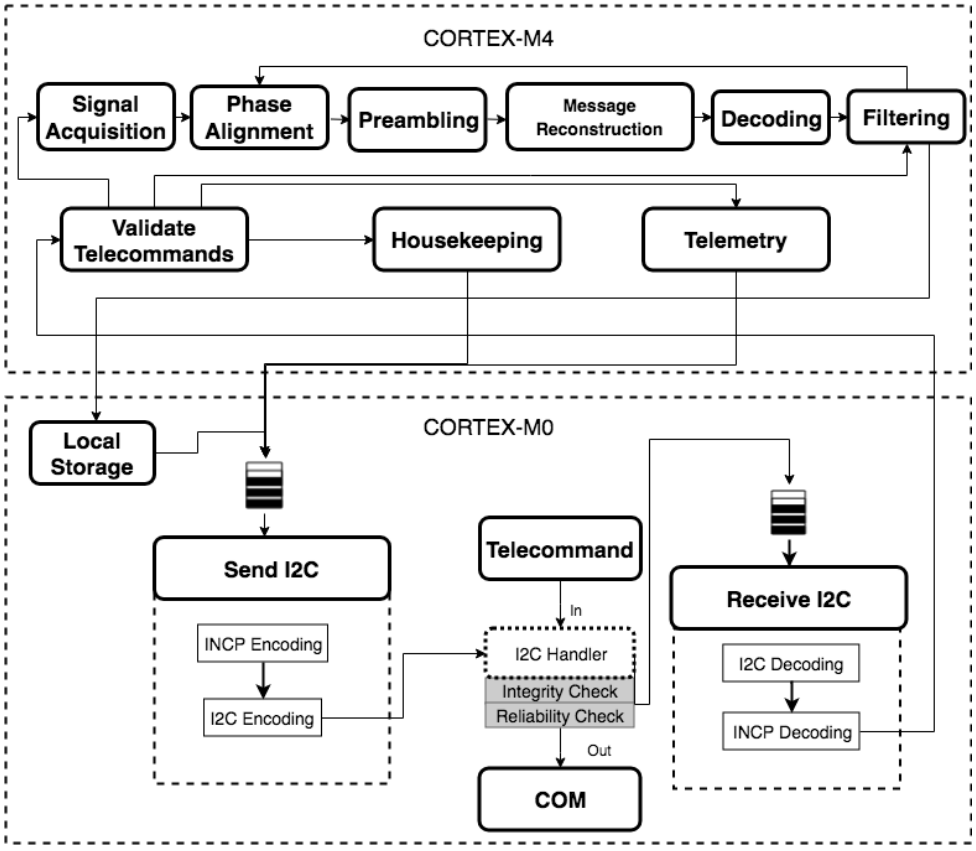

Figure 4.3: Distribution of the main functions in each core of the LPC4370

The ARM Cortex-M4 is responsible for the logical part of the system as this processor is the most powerful in terms of processing. ARM Cortex-M0 is only responsible for sending or receiving messages through I2C using the INCP to create or decode a message. However, this was not the final architecture.

The final architecture is shown in figure 4.4 and only has the ARM Cortex-M4 active. After the implementation of the receiver chain, it was possible to understand that this core can handle with all functions of the system. On the next sections, the development of each one of the functions will be

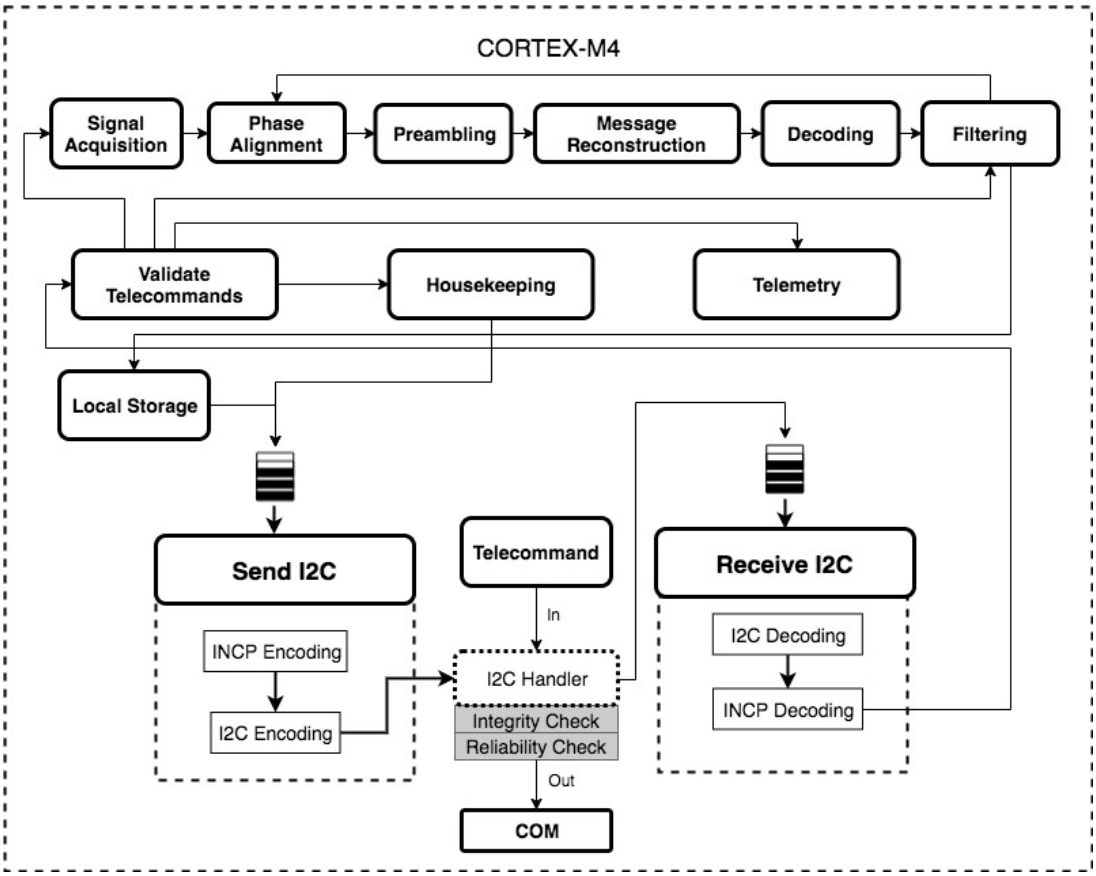

Figure 4.4: Final architecture design of the LPC4370

provided in detail. Moreover, special attention will be given to the receiver chain that contains the signal acquisition, phase alignment, preambling, message reconstruction, decoding, filtering and local storage. This is a straight-forward process where each function receives the output of the last one. Phase alignment will receive the samples retrieved in the signal acquisition. Preambling will receive the starting point of the preamble gathered on the phase alignment. Message reconstruction will take place as soon as preamble conditions are met. Decoding can only start after the correct reconstruction of the 112 bits of the message. Filtering can only be used after the correct decoding of the message. Local storage will only occur if the message has passed the active filters at the moment. If all these conditions are met, the message may be sent through the I2C formatted with the INCP to the COM subsystem for persistent storage. Telemetry, Housekeeping and Validate telecomands functions will only occur sporadically or on demand from another subsystem.

4.2 High Speed ADC and GPDMA

The LPC4370 was chosen mainly because of the ADCHS peripheral and because of the triple-core architecture that allows detecting and decoding messages simultaneously. This peripheral is responsible for converting voltages from the RF frontend into samples at a specific sampling rate, which are transferred to a particular location on the Static Random Access Memory (SRAM) memory using the

GPDMA. The usage of the GPDMA peripheral is intended to save processing power from the M4-Core as the converting is done by hardware and not software-based. Therefore, the samples required for the Payload algorithm are provided through a combination of the ADCHS and GPDMA peripherals, as shown in Figure 4.5.

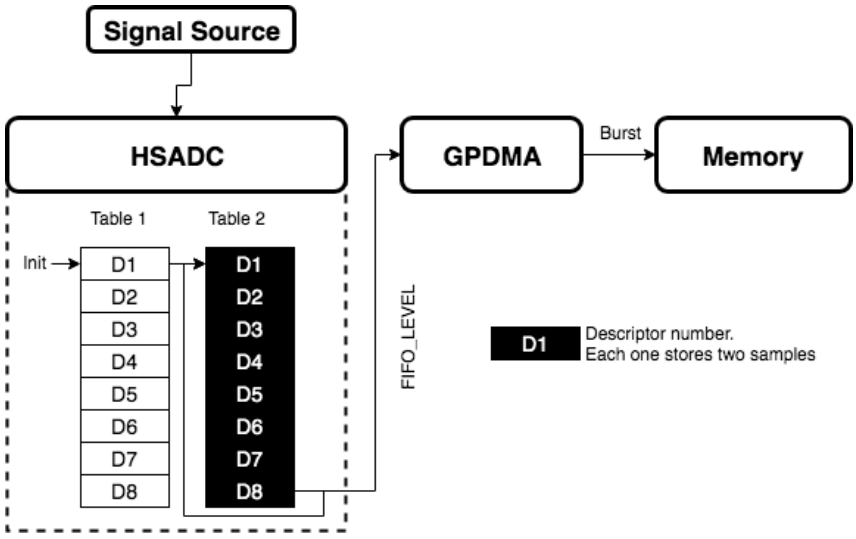

Figure 4.5: Process to put ADCHS samples directly to memory using GPDMA

4.2.1 High Speed ADC Configuration

The first thing considered for the ADCHS peripheral is the configuration of its sampling rate. Each conversion takes only one clock cycle of the clock of this peripheral, whose maximum value is 80 MHz. However, to provide a maximum sampling rate of 12 MSamples/s, a clock of 12 MHz is selected, as this lower clock peripheral saves power. This clock is different than the processor one whose maximum value is 204 MHz.

The ADCHS works from 0 to 1.2V and the expectation is that the signal comes between the 300mV and 1.1V. It has also a feature to detect High and Low thresholds by software. However, this setup can't be done for this project as the software generates an interruption each time the condition has been met and the number of interrupts multiplied by the number of samples created will not satisfy the condition to detect and decode simultaneously.

The acquired samples are stored into a 32-bit FIFO buffer with 8 positions. Additionally, every position of this FIFO may store two consecutive samples through the packed feature, as shown in figure 4.6. This means that this FIFO is capable of holding 16 samples before transferring them through the DMA.

Recalling Figure 4.5, the ADCHS is configured with the use of two descriptor tables, which may be considered as two internal state machines with 8 descriptors each. The ADCHS clock feeds an internal peripheral timer, and this state machine is verified at each increment, meaning that each descriptor may be configured to be called and start a conversion based on a match with a certain value of this timer, providing different acquisition rates. Furthermore, interchange between each descriptor table may be configured on each descriptor.

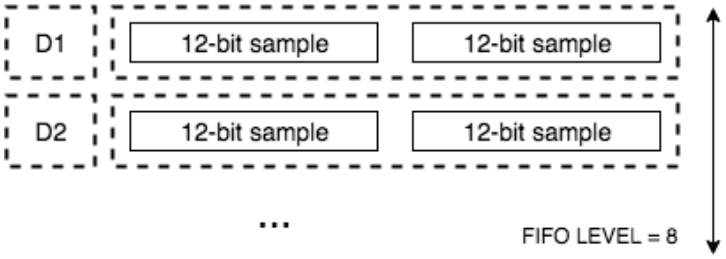

Figure 4.6: Packed samples inside each descriptor

The first table is only used to provide an initial startup delay, required for the ADCHS peripheral to start its acquisitions correctly. For that, its first descriptor is set with a startup delay required to be higher than 12 μs, according to the manual of the processor. After that, the state machine jumps to the next descriptor table and stays there for the rest of the acquisition time.

Since a stable sampling rate of 12 MSamples/s is used, every descriptor of the second table is configured to reset the aforementioned timer and all match values are set to 0, meaning that two samples are acquired at every successive descriptor call (two samples due to the aforementioned packed feature). A round robin configuration of the second table is done so that the ADCHS is continuously running, meaning that the state machine returns to D1 from D8.

All these configurations must be done prior to start the ADCHS acquisitions, which is triggered or stopped through its dedicated software trigger register.

4.2.2 GPDMA

At this point, it is known how the ADCHS is configured, running continuously and providing two packed 12-bit samples in a 8 position FIFO buffer that must be constantly emptied. Therefore, when this buffer is filled with the resultant 16 consecutive samples, a DMA request is issued to transfer these samples directly to another memory region without using the processor cores.

The DMA is configured so that these transfers of 32 bytes (2 bytes per sample) are done simultaneously, being incrementally repeated to two consecutive DMA buffers with a fixed size of (4088 * 4) = 16352 bytes. Theoretically, with this size, the chosen DMA buffer takes approximately 681 μs to fill up at a sampling frequency of 12 MSamples/s. After this time, a DMA interrupt is called to inform the ADS-B algorithm that a new buffer can be analysed. Furthermore, the two buffers are configured as two linked lists, meaning that the next buffer is filled up without the interruption of the ADCHS acquisitions and following Direct Memory Access (DMA) transfers after the aforementioned DMA interrupt is called.

4.3 ADS-B Algorithm

The ADS-B algorithm is based on the Dump1090 code[10], which is used to receive ADS-B messages in a computer using a simple RTL-SDR antenna. The implementation of Dump1090 is based on a Linux implementation which can not be directly ported for the Payload microcontroller. Regarding this issue,

the first step was to understand the necessary changes so that the Dump1090 could run on the Payload, giving special attention to the receive, reconstruct and decode functions and grab their logical parts.

In the developed algorithm, all the logical part of the Dump1090 remains the same, checking for the preamble, reconstructing a message and decoding the fields of each message. However, a couple of functions have been adapted to run on an embedded system. Other functions were created from scratch because of the new implementation characteristics.

The actual implementation of the Dump1090 only uses 2 MSamples/s, which is not even close to an ideal situation. Higher sampling rates means that the creation of bits using samples decisions are made with more precision. If a higher sampling rate is used, there should be more samples available at the same level, which will help to find the different Manchester code levels. To overcome this problem, Dump1090 uses an interpolation method which allows it to sample at 10 MSamples/s mathematically. However, this process was not considered for the Payload as it would require a lot of processing power which is not available in a low power microcontroller.

Instead, the approach was to set up a higher sampling rate, accomplished with the aforementioned ADCHS and DMA. The sampling rate is one of the most critical parts of the implementation, especially when dealing with the Manchester code and the correct reconstruction of bits, as a higher sampling rate will mean more samples on the same level and therefore a smaller probability of a wrong detection of a transition between samples.

The Dump1090 contains two primary functions to receive a message: Detect message and Decode message. The former is intended to detect an ADS-B message which contains the preamble checking pattern. The latter is related to the reconstruction of an ADS-B message. In this process, samples are used to decide which bit is created.

Due to the design of the Payload RF frontend, the ADS-B signals arriving to the ADCHS input are inverted. This has to be considered in all algorithm phases that are further described.

The developed ADS-B algorithm contains a couple more significant functions compared to the Dump1090 implementation, namely, FindFirstLow phase, Preambling phase, Reconstruct phase and Decoding. The Find First Low function has been inspired by the Detect Message function of the Dump1090. However, Dump1090 uses a different way of checking the samples, as the algorithm firstly fills a buffer to process it afterwards, losing new samples while this processing is done. In this new algorithm, it is required to detect and decode messages simultaneously. DMA is used to retrieve the samples as fast as possible, which allows saving processing power of the ARM Cortex-M4 that will handle the algorithm processing. Figure 4.7 represents the state machine that is used inside the DecodeBuffer function that is responsible to control the flow of the receiver chain.

4.3.1 Dynamic Threshold Calculation

Before entering the DecodeBuffer state machine, the system will be responsible to calculate the value of the dynamic threshold. The calculation is based on processing several buffers and checking the sample values to find the highest and lowest values. This is a crucial step in order to correctly define the dynamic

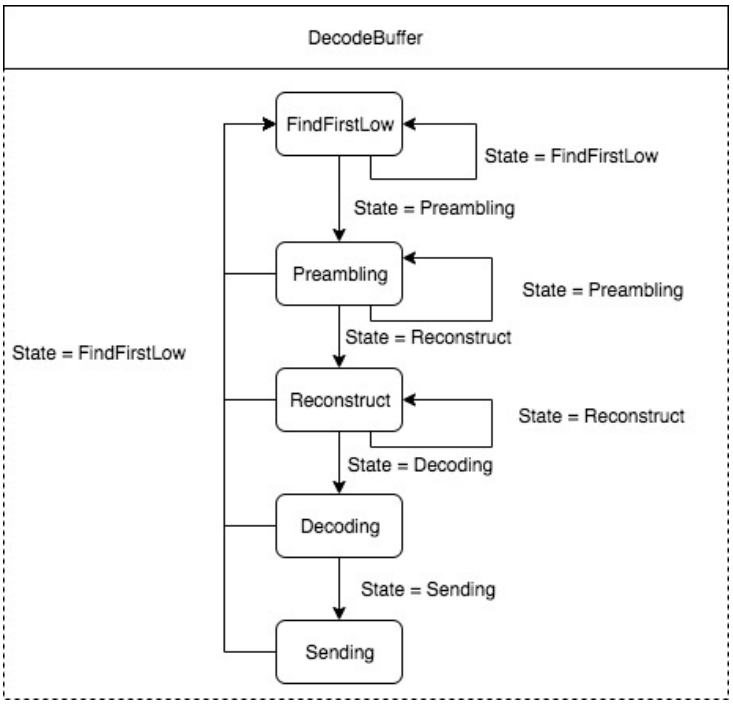

Figure 4.7: Machine States of DecodeBuffer Function

threshold that shall be used to reconstruct the ADS-B messages. Equation 4.1 demonstrates how the calculation of the medium threshold is done.

Noise Case

3583	3417	3445	3811	3765	3436	3524	3867	3747	3449	3634	3860	3649	3451	3704	4018	4095	4025	4004	3950	3800	3763	3650	3827
4087	4002	3857	3560	3445	3598	3734	3899	4009	4090	3978	3875	3679	3524	3413	3380	3545	3711	3928	4030	3954	3901	3823	3671

Medium Noise Threshold Value: 3740

Possible Message Case

2359	1749	1492	1528	1652	2131	2798	3241	3265	3140	3300	3219	2709	2075	1581	1269	1471	2040	2763	3378	3583	3417	3445	3811
3765	3436	3524	3867	3747	3449	3634	3860	3649	3451	3704	4018	4095	4004	3655	3365	3241	3245	2952	2350	1754	1423	1553	1970

Highest Value: 4095	Lowest Value: 1269	Medium Threshold Value: 2682

Figure 4.8: Dynamic Threshold and Noise Medium Value

$$Medium\ Threshold\ Value = \frac{High\ Value - Low\ Value}{2} + Low\ Value \tag{4.1}$$

Figure 4.8 represents a smaller portion of the samples inside a buffer, where sample values will vary between 0 and 4095 due to the ADCHS 12 bit range. All values are real samples retrieved from the ADCHS and will be very useful for all future functions of the receiver chain. The figure also represents the two possibilities when creating the medium threshold. The noise case represents a medium threshold

where no message has been found and thus no samples at a low level appear. The possible message case represents a possible good threshold that can be used to find ADS-B messages.

4.3.2 FindFirstLow Phase

In this phase, the idea is to process the samples and find the first spike of the preamble pattern using the capabilities of ADCHS and DMA. There are two processes that can be used here. The first one processes the 96 samples correspondent to a preamble each time. if the preamble is not matched, the first sample shall be dropped and the next one on the buffer shall be used. For the construction of each ”bit” of the Preamble, six samples shall be used. The second one is intended to save processing time. It always uses 12 samples and the idea is to detect a transition higher than the Fixed Threshold. The dynamic threshold shall be updated when the Payload system is powered on but also from time to time because this threshold is not constant due to different noise levels.

After the possible first spike is found using this process, the idea is to unambiguously detect the first sample under the dynamic threshold of the dynamic range. The idea is to iterate one by one all the 12 samples where the spike has been found and clearly identify the first one smaller than the medium threshold value. When the first sample lower than the threshold value is found, the synchronisation is done, which means that the algorithm is clearly aligned with the first sample corresponding to the preamble and the algorithm moves to the Preambling phase.

Figure 4.9 demonstrates how the dynamic threshold is used and what are the different use cases inside the FindFirstLoW phase. The idea is always to dynamically set up a medium threshold for the message and another for the noise value so that both can be used on these two functions.

4.3.3 Preambling Phase

Regarding the Preambling phase, the new algorithm uses the same logic as the Dump1090, using a specific pattern that needs to be matched. Here, the Manchester code is not used, leading to process six samples each time instead of the 12 used with Manchester to reconstruct one bit of the message. The sum of each six samples is stored in a buffer with the size of 16 positions which is the size of the Preamble pattern. The comparison is then made using the values inside that buffer.

In Figure 4.11, each P[number] corresponds to the sum of 6 consecutive samples that shall be stored inside the small buffer with 16 positions (P[16]). Those comparisons are always made step by step, and if a condition does not match, the program will directly go to the FindFirstLow state, which means that no valid preamble has been detected. Furthermore, it will be directed to the DecodeBuffer state machine, restarting the whole process of checking for a preamble starting from the first sample of the last six samples that did not pass the condition. Otherwise, if all preamble comparisons passed, the DATA block of the message is intended to be on the next 1344 samples (112 bits * 12 samples). However, this number of samples is not fixed and will vary because synchronization deviations may occur.

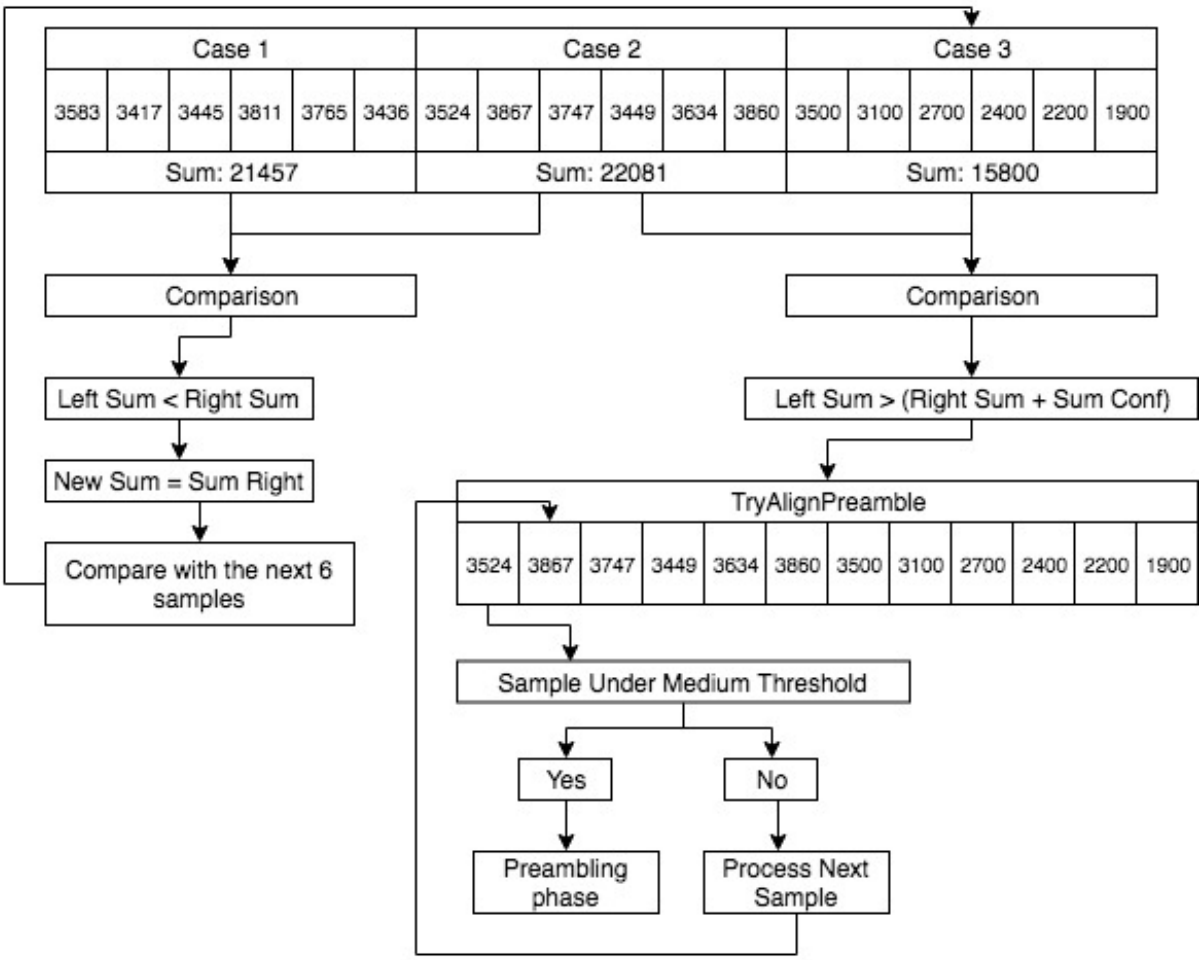

Figure 4.9: FindFirstLow and Preamble Align

4.3.4 Reconstruct Phase

This phase is responsible for converting samples into bits using Manchester Code. The first thing to determine is the level of the samples. The algorithm grabs the second and third samples and does an average value of both to determine if the level is high or low. High and low samples are determined comparing the average value with the dynamic threshold that has been calculated. After the decision, it is necessary to determine where the transition occurs. Ideally, if all samples come perfectly aligned, twelve samples are needed to construct one bit. However, some of the edge samples can have energy from the previous one and thus leading to one more sample on a certain level. If the first part of the bit construction is being checked, it is not necessary to count until 6, being only necessary to detect the transition from low to high or high to low between consecutive samples. However, if the second part of the bit construction is being checked, it is necessary to have a counter which should count 6 samples and then stop if the transition does not occur before the six samples. If the transition occurs, the bit creation should stop on the last sample of the same level. This process allows us to identify the several transitions that occur in Manchester and to synchronize the transitions.

In Figure 4.11 an arrow has been placed in sample number 7 of the second bit to understand the problem of the alignment. In this case, the correct transition occurs on the sample number 7, meaning

P[0] < P[1]
P[1] > P[2]
P[2] < P[3]
P[3] > P[0]
P[4] > P[0]
P[5] > P[0]
P[6] > P[0]
P[7] > P[8]
P[8] < P[9]
P[9] > P[6]
P[10] > P[6]
P[11] > P[6]
P[12] > P[6]
P[13] > P[6]
P[14] > P[6]
P[15] > P[6]

Figure 4.10: Preamble comparison pattern

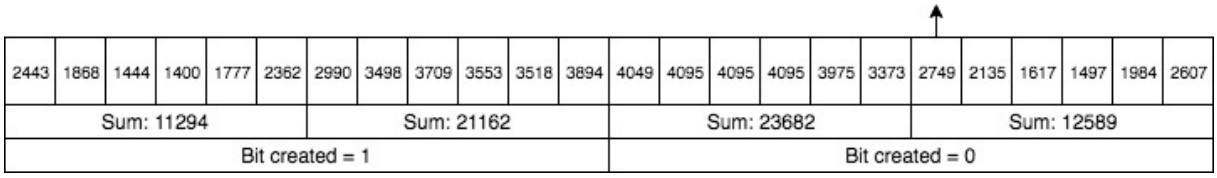

Figure 4.11: Reconstruct function

that a propagation error would occur if six samples were always processed, as this sample 7 would have the incorrect level.

Another aspect to take into account is that the algorithm needs to handle the DMA buffer jumps. If it is processing the second buffer, it needs to grab samples from the next buffer, which will be the first one again due to the circular buffer in order to complete the bit creation, and vice-versa. As the ADCHS is always running, it is not a problem to automatically access the buffer that is being filled by the ADCHS and grab the remaining samples to finish up the work.

Another important aspect here is that the reconstruct algorithm is capable of understanding if an ADS-B message is present or not. Mode-S messages short and extended (ADS-B ones) share the same preamble pattern. A way to save processing time is to detect which is the size of the message coming right after the preamble. After the construction of the bit number 8, the algorithm checks if the first 5 message bits correspondent to the DF are equal to 17 or 18 (in decimal values). If they are not equal to those numbers but are one of the numbers described on the table 4.1, the algorithm has the ability to jump the equivalent to (56 bits * 6) samples or (112 bits * 6) samples depending on the case of short or extended Mode-S messages. This allows to save a lot of processing time and will be useful only to reconstruct ADS-B messages in the least possible time.

Downlink Format	Content	Message Lenght
DF0	Short Air to Air ACAS	56 bits
DF4	Surveillance (roll call) Altitude	56 bits
DF5	Surveillance (roll call) IDENT Reply	56 bits
DF11	Mode S Only All Call Reply	56 bits
DF16	Long Air to Air ACAS	112 bits
DF17	1090 Extended Squitter	112 bits
DF18	1090 Extended Squitter, suplemmentary	112 bits
DF19	Military Extended Squitter	Undefined
DF20/DF21	Comm. B Altitude, IDENT reply	112 bits
DF22	Military use only	Undefined
DF24	Comm. D Extended Lenght Message (ELM)	112 bits

Table 4.1: All Mode-S messages, respective lenght and content

4.3.5 Decoding Phase

Finally, after the reconstruction of the message, a reduced decoder of the Dump1090 is used. This decoder will only contain information regarding the ADS-B message. Besides, the first thing to check is the CRC of the message. If the CRC is not correct, the message should be automatically dropped so that the processor can save resources for other functions like Housekeeping, Communications or Telecommands. The rest of the decoding phase is based on the Dump1090 and will allow to gather the necessary information for mission purposes. This decoder can identify and decode each one of the fields that come inside each ADS-B message described in the several annexes.

For the mission purposes, filters will be applied to understand if a message has to be stored on the COM subsystem or not. If filters allow the message to pass, the respective message shall be stored on the COM subsystem for further compressing and transmission to the GS. Otherwise, the message shall be dropped and will not be saved.

The mission shall produce a large quantity of data, as the ESA POCKET Housekeeping Telemetry Compression Algorithm will be used in the COM subsystem to reduce this data. To correctly use the POCKET Algorithm, it was necessary to define the size and format of the message that shall be sent to the COM. The final message will not include the CRC and but will contain all the other bits of the ADS-B message. Furthermore, a timestamp shall be added to the end of the raw ADS-B message. This will be done so that messages can be appropriately compared with ground datasets of ADS-B messages and know the time at which those messages have been received. The final size of the message being sent to the COM subsystem should be equal to 120 bits, correspondent to the ADS-B message without the CRC field (88) plus the aforementioned timestamp (32).

4.4 Filter's Planning

Filters were planned according to the mission of the Payload. Those filters will allow determining the performance measures already described, namely the POD, POI and PTA. For the correct planning of the filters, the analysis of the different messages and different metrics is essential. The list of filters that have been defined and the respective parameters were already described in section 3.7. The idea is to understand how to interconnect the different filters to get the several performance parameters of the mission.

Firstly, it shall be clear that a filter by time and a filter by location will always need to be present. Those filters will allow to reduce the number of received messages, making it easier to identify the "Cone of Silence" of the antenna. Also, for the POD measure, a database of ICAO addresses shall be used. This will allow storing the new ICAO addresses coming inside the ADS-B messages to understand how many different aircraft have sent an ADS-B message. Besides the mission parameters that will only need those three filters, other filters can be added to verify various aspects such as all aircraft on a downward trajectory or all aircraft with a velocity higher than a specific value.

The payload system needs to have a couple of functions that allow users in the ground to manage the different filters. For this purpose, it is crucial to have a defined API so that users on the ground station can set up the new filters without any confusion. From the ground station, it shall be possible to send telecommands to get the list of filters, to add a new filter or to reset the filters.

The default filters that will be available on the Payload are described in figure 4.12.

Filter Name	Filter Parameters
Filter By Location	Latitude Top, Latitude Bottom, Longitude Left, Longitude Right
Filter By Time	Start Time, Stop Time
Filter By Typecode	Typecode Decimal Value

Figure 4.12: Default Filters for the Payload

Those filters will allow acquiring results that can be correlated with the messages received in the same time frame and over the same area by ground stations. The area of interest is already and will be around the Azores, Portugal. It is expected that in a 10-minute mission, a medium of 20 aircraft will be present simultaneously inside the respective area of interest. This will represent a number of around 40.000 ADS-B messages sent by aircraft over that period. The TC value will depend on what is the default mission parameter of the satellite at that particular time. However, at least those three filters have to be present to accomplish any of the mission measures.

The filters shall not be erased when a shut down to the system occurs, and thus a non-volatile memory shall be used. In the case of the payload, the Flash memory shall always be used to retain the active filters so that on a power down, the data is stored.

4.5 Operating System

The payload can use or not an operating system. For the receiver chain, there is no need to add any operating system as the system has time to process samples and decode messages. However, to better manage the system, while waiting for the next buffer, a real-time operating system like FreeRTOS may be used. This operating system will allow to schedule the several tasks and create better management when the system is dealing with the functions that are not under the receiver chain scope. When the system is processing the buffers from the DMA and a request arrives through the I2C for the Validate of telecommands function for example, the use of an operating system will help managing the interchange between functions.

4.6 Testing bench

To correctly test this architecture, a chain of tests have been thought for this long term project. The receiver of the payload expects to have an extensive signal reception. So, the first idea to test the reception of signals is related to the use of an ADALM-PLUTO with a GNU-Radio software to produce an ADS-B message.

ADALM-PLUTO is a SDR active learning module that is capable of simulating the transmission of an ADS-B message being sent from an aircraft. GNU-Radio is a free software development toolkit that provides signal processing blocks to implement software-defined radios and signal-processing systems that will be used to generate the respective ADS-B message. This process allows to understand if the samples produced by the system are acquired inside the payload system. Furthermore, with a well formatted ADS-B message, whose output is already known, it should be possible to validate all the receiver chain process.

When the first test is fully functional, a second test shall be done with the antenna and the payload board, which includes the digital and receiver modules shall be used. The idea is to correctly set up the antenna and receiver and check the results of receiving real messages from several aircraft. The mockup for the Ground station should also be used to receive the message that shall be decoded during the mission. However, this should not be the final test.

In the earth surface, the ”Cone of Silence” generated by the antenna is not going to be the same as the one expected in the LEO. Here, the simulation of the real test in the earth surface means that the satellite is not moving and only the velocity of the plane will be the constraint for determining the ”Cone of Silence”. However, in normal mission conditions, satellite and aircraft velocities will have an influence on the determination of the ”Cone of Silence”, where the worst case to be considered is when a plane is travelling in the opposite direction of the satellite. Also, the pointing of the satellite will assume a huge importance on this problem. If the satellite is pointing to a different direction that the earth surface at that moment, the probability of receiving message will be null. Furthermore, the final testing can only take place with the satellite in place at 400km above the Earth’s surface. However, the deadlines for this project will only allow this test to occur in 2021. In that year, the student responsible for the

Ground station data handling will be able to determine the ”Cone of Silence” during the mission. This will be possible after the comparison of the downloaded messages from the satellite with the datasets of messages received on the Earth’s surface.

4.7 Development Board and Payload Board

The first testing approach was to use the LPC-Link 2 to explore some of the capabilities of the LPC4370 microcontroller. This development board was used as an evaluation board for the NXP LPC4370 triple-core MCU[36], allowing to produce the first testing purposes regarding the ADCHS and the DMA.

After that, a final payload board has been used with both digital and RF part for the further developments. Figure 4.13 shows the final Payload board with RF and digital parts.

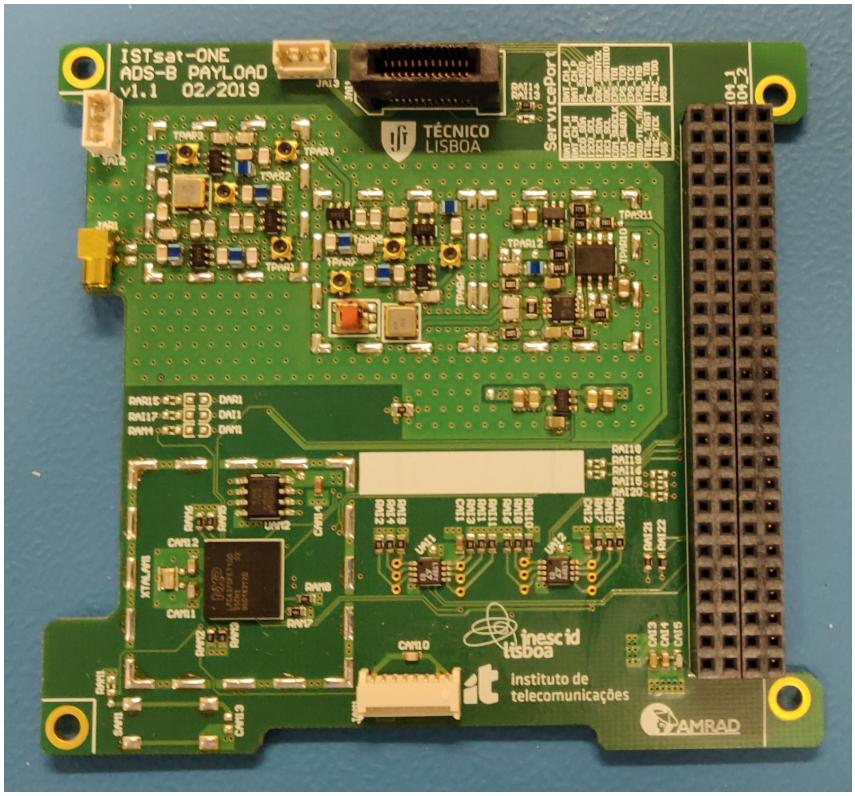

Figure 4.13: Final board of the Payload system with RF and digital parts

The board is divided into two areas. The top one is the complete RF part that is responsible for receiving the ADS-B signals. The bottom part is the digital part that has the LPC4370 microcontroller and the respective peripherals. This board has already the final components and is supposed to be the final ISTsat-1 flight model board.

Chapter 5

Performance Evaluation

A performance evaluation is used to determine how a system performs in terms of responsiveness and stability under a particular workload. It also validates, measures, investigates and verifies other quality attributes of the system, such as scalability, reliability and resource usage.

This chapter contains the validation of the algorithm developed to acquire the ADS-B messages. Furthermore, all tests performed on the algorithm are fully demonstrated by the respective results. It contains the unitary tests required to verify if the algorithm was doing precisely what it is supposed to do before moving to the ”real tests”. Alongside, the performance of the algorithm is shown regarding the speed of processing a single buffer while the DMA is filling another one. Also, an evaluation of the workload of the Payload system is provided. Finally, an evaluation of the energy consumption is done regarding the power budget available for the payload system.

5.1 Algorithm Validation

Algorithm validations are usually present during software developments. These validations will allow to clearly understand if the algorithm is performing as it is supposed, as different tests may be done during these validations, mainly related to the unitary tests, the ADALM-PLUTO tests and the real test that shall be done during the mission of the satellite.

5.1.1 Unit Testing

Unit Testing is known as a software testing method that allows the test of small functions or procedures to verify if the output is the expected one. The software of the payload was not an exception, and it was also necessary to verify particularly well the different functions of the receiver chain.

For this testing purposes, it was used the Munit framework [37], which is used by all users of the ISTSat-1 project. A couple of tests have been implemented, such as:

1. Test Init
2. Test General Functions

3. Test FindFirstLow
4. Test TryAlignPreamble
5. Test Preambling
6. Test ReconstructingMessage
7. Test Decoding

Inside each one of these main tests, several small tests have been done to eliminate any doubts that may occur about the correct results.

The purpose of the Init test is to verify if the GPDMA, the ADCHS and the Inter-integrated Circuit (I2C) are being correctly initialized as they are the most important peripherals to guarantee the perfect functioning of the payload system.

For the general functions test, smaller tests have been created. The first one is the Samples Left test, intended to determine the number of samples left on a buffer. The second one is the Finish Buffer test, used to verify if the program has the capacity to the change from the last sample of a buffer to the first sample of another buffer. Most of these small functions will be used inside the big functions already described, such as Preambling, Reconstructing or Decoding.

For the acquisition (FindFirstLow test), it was necessary to test if the sum of the six consecutive samples was correct. Then, a test was done to understand when the actual buffer has no more samples to process. Next, a couple of tests were done to determine if a preamble could be found, which focused on understanding if the first sample of the preamble corresponds to the correct one, verifying if the alignment of the preamble is properly done. The next step was to create some testings for the Preamble function, confirming that all comparisons are correct and that the samples are being checked correctly and consecutively.

For the other primary functions, the testings were to check if the message was reconstructed correctly, meaning that all bits are equal to the known message being tested and that its CRC also validates it.

The whole project is maintained over the gitlab repository. To properly test this, the Linux was used instead of the LPC4370. All these tests use a couple of buffers with real ADS-B samples. These samples have been produced with the help of an ADALM-PLUTO that is a software-defined radio capable of generating a complete ADS-B message.

All tests performed well and clearly give the feedback that everything was running correctly with a defined buffer.

5.1.2 ADALM-PLUTO testings

A couple of tests using the ADALM-PLUTO have been thought as they are done under a controlled environment. These tests serve to verify the correct functioning of the receiver chain. The first test consisted of sending a message from the PLUTO and decode it correctly on the Payload. To correctly

set up this test, it has been necessary to use an RTL-SDR to receive a couple of real ADS-B messages that serve as test messages to be sent from the ADALM-PLUTO to the payload system. Furthermore, all messages used for these testing purposes are real messages sent by an aircraft.

After the correct reception of those messages, an Octave script has been used to transform the Hexadecimal code of the message (Ex: 8D49528899045E9CE84E0F3460B3) into a .dat file that will be read by the ADALM-PLUTO. However, the last six hexadecimal characters corresponding to the CRC of the message shall not be used to create the ADS-B message, as this Octave script calculates the CRC and then produces the .dat file with the correct information. The algorithm was developed by a professor of the IST to test the RF part and check if the Payload could receive messages from an aircraft, being then used to build test messages from real messages detected on the earth surface.

To correctly send ADS-B messages and the respective noise between two consecutive messages, a new file only with noise values has been created. A python script has been used to read the ADS-B and noise files, being the ADALM-PLUTO responsible for reading these files and sending the correspondent signals, as they are the source for the ADS-B message and noise. However, the ADALM-PLUTO system needs a brief period to start sending the values that will correspond to the ADS-B message. Because of that, extra noise must be sent first. The python system has a defined number of messages that will be sent, and the time frame of noise samples can be adjusted.

The first test consisted of sending the example message 8D49528899045E9CE84E0F3460B3 and check if the Payload can detect and decode the message correctly. The produced result has revealed that the received message was equal to the test message sent by the ADALM-PLUTO. Moreover, this was the first time that the receiver chain has revealed to be correctly working.

The TX output of the ADALM-PLUTO is directly connected to the receiver of the payload. Between these, a 40 dB signal attenuator is used to reduce the signal power, as the Radio Frequency (RF) receiver frontend is designed for signals with less power, as the received power signal at 400 km of orbit distance is not equal to the ones received in the Earth's surface from the aircraft that are located at 10 km of altitude.

Some performance issues were detected on the ADALM-PLUTO transmission during its testings. Since it only has a USB 2.0 interface, which is a problem when dealing with vast amounts of data on a small time frame, an error may occur on the transmission of some messages. For example, some messages did not include the Preamble pattern, others were just compressed in size, and others appeared with the preamble but only showing 5 or 6 bits of data. These results were verified through an oscilloscope connected to the input of the ADCHS of the microcontroller. The percentage error introduced by this problem is about 5 % on all results, which is clearly identified in each one of the following results.

The first test performed was based on sending equal messages with a defined time interval and check how many the Payload was capable of receiving and decoding, confirming if the time frame period between messages is relevant to the test purpose. The results of this test are shown in table 5.1. These results serve as a basis comparison for the evolution of the algorithm.

Several improvements mainly related to the reconstruction phase and buffer transitions allowed the number of received messages to increase approximately 40 %, as may be seen in the following test

	Test 1	Test 2	Test 3	Test 4
# Messages Sent	45	45	45	45
# Correct Messages Sent	43	43	43	43
# Reconstructed Messages	32	31	29	31
# Received Messages	24	25	21	26
Percentage of Messages Decoded	56%	58%	49%	60%

Table 5.1: First test results with ADALM-Pluto sending ADS-B messages

results.

Then, the next step was to add other types of messages to the Python program so that the Pluto software can read them. Different types of messages were added for the same aircraft, like Position, Location and Identification. A couple of messages and the corresponding hexadecimal value are:

- Position Message: 8D495288601F255D49AB05F32AC6
- Velocity Message: 8D49528899045E9CC8490F5B3C1A
- Identification Message: 8D49528820501437E333A0A9E974
- Another Position Message: 8D495288601F11CB239DB7AEB333

Figure 2.4 shows how to divide each one of the messages and determine the respective Downlink Format, ICAO, DATA frame and CRC. It is clear to understand that they are all ADS-B messages as 8D is represented by 10001101 in binary format. Looking to the first 5 bits, 10001 corresponds to 17 in the decimal value, which is the ADS-B message code. Also, it is clear to understand that for all of those messages, the ICAO address is the same (495288). In a close look to the Data frame of the message it is possible to identify the several types of messages. For the first case and last messages, it is clear that the data frame starts after the ICAO address. For both, it is necessary to look at the first 5 bits, which corresponds to 01100 in binary (60 in hexadecimal). The decimal value for both is 12. Table 2.1 allows an understanding that a position message can have the decimal value between 9-18 and 20-22. For the other ones, the process is the same. It is easy to guess that for the velocity message, the TC of the message is 19 (10011 in binary) and for the identification message is 4 (00100 in decimal). The several fields of each one of the messages can be found on the appendix pages of this work.

The algorithm decoded all fields, and the CRC was correct, which leads to good reception of the messages. The duration of each one of the tests is around 2 minutes. The separation time frame between message is always the same and is around 0,4 seconds. This is almost half the time frame used by aircraft to send the Velocity and Position messages (1 second between messages from the same type). Table 5.2 demonstrates the several tests done to understand how well the payload system can deal with the different types of messages.

Regarding table 5.2, the number of reconstructed messages is smaller than the correct number of messages as the algorithm was not capable to identify the preamble and thus not reconstructing the message.

	Test 1	Test 2	Test 3	Test 4
# Messages Sent	300	300	300	300
# Correct Messages Sent	285	285	285	285
# Reconstructed Messages	213	216	209	213
# Messages Decoded	209	213	204	213
Percentage of Messages Decoded	73%	75%	72%	75%

Table 5.2: Tests acquiring several different messages from the same aircraft with a time interval of 0.4s

Test sending different types of message (Position, location, identification) from various aircraft. For this test, two new messages have been added from another aircraft, namely:

- Velocity Message: 8D49528F99046A9B08270D248FA3
- Another Velocity Message: 8D49528F681BA5590DAB2C2AD128

For both messages, the same process as described above can be done to determine the Downlink Format, ICAO, Data frame and CRC of the respective ADS-B message. The several results acquired are shown in table 5.4.

	Test 1	Test 2	Test 3	Test 4
# Messages Sent	450	450	450	450
# Correct Messages Sent	427	427	427	427
# Reconstructed Messages	295	311	304	292
# Messages Decoded	289	303	297	287
Percentage of Messages Decoded	68%	71%	70%	67%

Table 5.3: Tests acquiring several different messages from multiple aircraft

The next couple of tests performed are related to the filtering part of the receiver chain. In those tests, the idea was to understand how the system deals with several active filters and several messages being received by the Payload.

Active filters during the first test:

- Location
- TC (Velocity)

The filters used allowed to understand if the system can detect the velocity messages(TC value of the filter).

Table 5.4 gives an overview of the results of using the Location filter and the TC filter (only looking for velocity messages). During the second test, the location filter was kept, but the TC has been changed to look only for the Identification messages. The results of table 5.5 allowed to understand that the system is capable to only detect identification messages from all the ones that have been decoded.

	Test 1	Test 2	Test 3	Test 4
# Total Messages Sent	450	450	450	450
# Total Velocity Messages Sent	213	213	213	213
# Messages Decoded	287	284	299	287
# Filtered Messages by Location	96	99	101	92
# Filtered Messages by TC	96	99	101	92

Table 5.4: Results from filtering the messages by Location and TC(Velocity)

	Test 1	Test 2	Test 3	Test 4
# Total Messages Sent	450	450	450	450
# Total Identification Messages Sent	71	71	71	71
# Messages Decoded	287	293	306	288
# Filtered Messages by Location	46	48	52	45
# Filtered Messages by TC	46	48	52	45

Table 5.5: Results from filtering the messages by Location and TC(Identification)

The next test performed only used a single filter. The filter used is the ICAO one which will be used to follow a path from a specific aircraft. Here, the idea is to understand if the system can only store the messages from that single ICAO. Table 5.6 shows the results acquired in four different tests.

	Test 1	Test 2	Test 3	Test 4
# Total Messages Sent	450	450	450	450
# Total Effective Messages Sent	427	427	427	427
# Total Messages sent with same ICAO	150	150	150	150
# Messages Decoded	315	283	305	303
# Stored Messages after filtering by ICAO	98	97	98	101

Table 5.6: Results from filtering the messages by ICAO(49528F)

The results provided give an idea that around 60 to 70% of the messages are received on the Payload system. Even though the test only has two different ICAO addresses on all messages, the filter allowed to reduce the stored messages.

For the last test, the idea was to define an area outside the range and understand if the system can't receive any messages.

Table 5.7 provides the results of this test. It is possible to determine that no messages have been received on the payload. It is clear that the system is ready to deal with this kind of filters. Furthermore, the mission measures have been calculated for the new four tests using the same amount of messages. For those tests, the active filters are:

- Location (Portugal zone)
- ICAO

	Test 1	Test 2	Test 3	Test 4
# Total Messages Sent	450	450	450	450
# Total Velocity Messages Sent	427	427	427	427
# Messages Decoded	310	293	297	301
# Stored Messages after filtering by Location	0	0	0	0

Table 5.7: Results from filtering the messages by Location

- TC (according to the mission)

Table 5.8 shows the results for the mission measures with the received messages.

	Test 1	Test 2
# PTA	100%	100%
# POD	67%	67%
# PTI	75%	69%

Table 5.8: Mission measures for the same aircraft

With these results, it is possible to understand that the system was capable of detecting all different aircraft. Moreover, the percentages of the POD and PTI are always on the 60-70% range, which is a good starting point to receive ADS-B messages. Comparing with the results of the Proba-V mission, for example, the POD measure for the Australian area was 22,4% and was their best result in terms of area of interest for that measure. Furthermore, the POI measure was 17% for the same reference area. Of course, the comparison of results from our 60-70% to their 17-22% is not correct but can give good perspectives for the real mission test. The final idea is to compare our results with those obtained from the Proba-V mission.

5.1.3 Real Test simulation

The real test simulation is intended to demonstrate if the payload system can receive messages from real aircraft. The tests done with the ADALM-Pluto revealed that the Payload could receive ADS-B messages. However, in a real scenario, not only ADS-B messages but also Mode-S messages will be detected by the Payload RF Frontend, being necessary to verify how the system deals with other messages. Figure 5.1 shows a real message representation to understand the test performed. This message has been captured using the antenna and the payload system.

It is possible to detect the preamble and the data block that is associated with an ADS-B message. In a real scenario, Mode-S messages will be detected by the Payload as they have the same preamble as an ADS-B message, as shown in Figure 2.2. The idea here is to verify if the Payload can distinguish the several different messages to save processing time. Furthermore, the Payload should detect and decode correct ADS-B messages.

This test was performed on IST-Taguspark campus. The antenna was placed outside of the campus connected to the Payload system, pointing directly to one of the common routes of departure and arrival

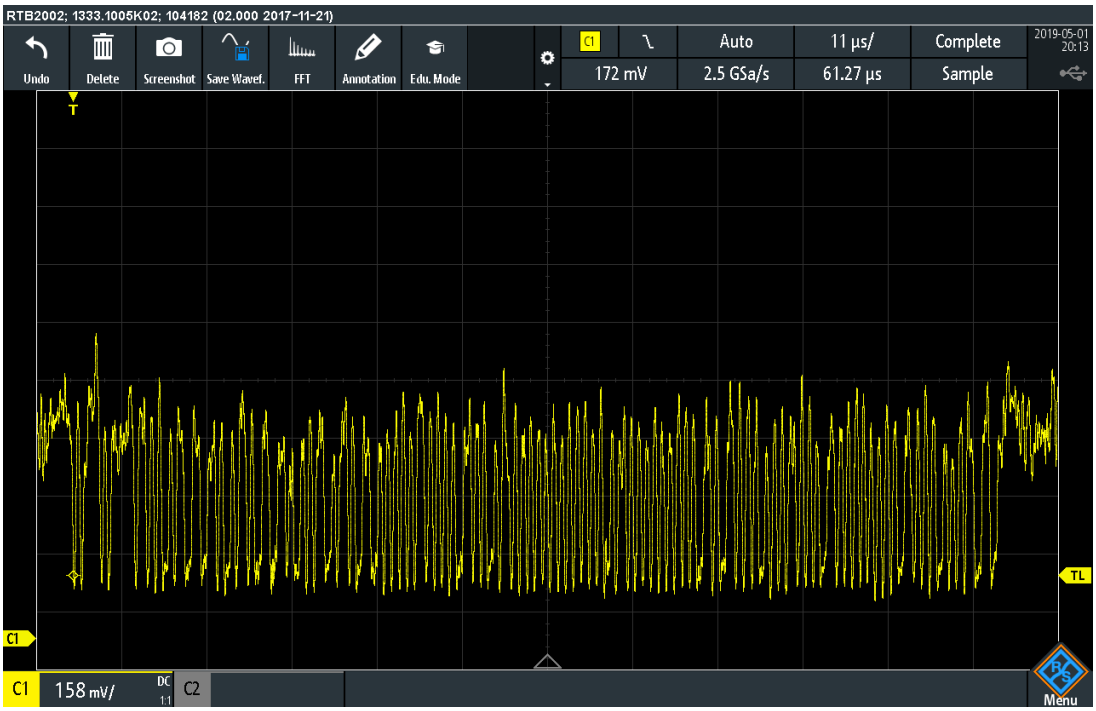

Figure 5.1: Representation of a real ADS-B message received in the Payload

from the Lisbon airport, where a couple of aircraft routes from central Europe cross this path on flights to South America. Figure 5.2 demonstrates the paths of two planes, one departing from the Lisbon airport and another one travelling to Brazil from Amsterdam.

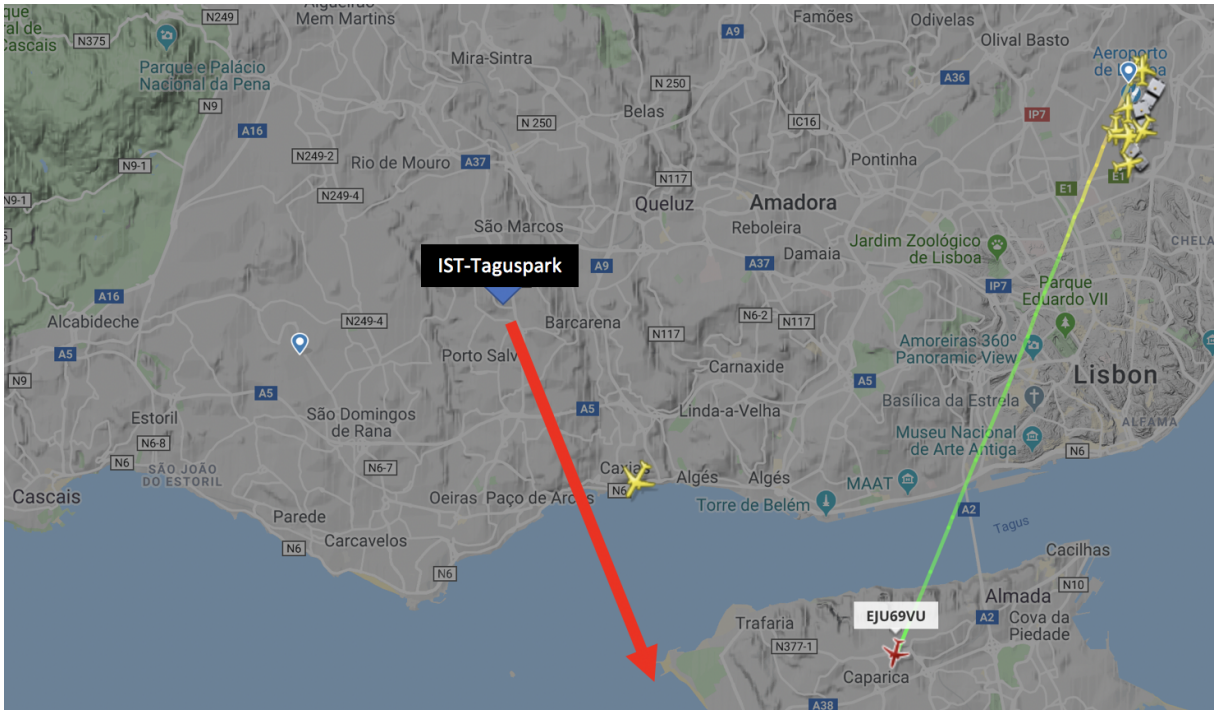

Figure 5.2: Aviation paths crossing IST-Taguspark

The red line demonstrates the pointing of the antenna. The ”Cone of Silence” test will not be possible as aircraft do not travel above the IST-Taguspark where this test was performed. However, a lot of planes

travel daily in these paths, which is good for testing purposes. An average value of 680 aircraft depart from or arrive at the Lisbon airport per day, meaning that in a one hour time frame, there should be at least 20/30 planes crossing this path. Moreover, the Cascais Municipal Aerodrome is responsible for having a lot of small duration flights from pilots that are taking the pilot course. These pilots need to make flight hours and most of these small aircraft already transmit flight information through the ADS-B system as well. Those messages are also received on the payload system. The time of activity is small as the aircraft are not visible for a long time and bigger activity times would not improve the measures as no more ADS-B messages would be received on the payload system.

	Test 1	Test 2
# Time of activity	30s	30s
# Possible Preambles	1500	693
# Reconstructed Messages	94	60
# Messages Received	99	65
# of Other Mode-S Messages	270	211
# of Wrong Messages Decoded	17	6
# Position Messages	37	23
# Identification Messages	3	3
# Velocity Messages	33	25

Table 5.9: First test results with ADALM-Pluto sending ADS-B messages

Here, other Mode-S messages have been received, and thus the results are different compared with the ADALM-Pluto ones. Here, the number of received ADS-B messages is clearly above the ones obtained with the ADALM-Pluto because there is a lot of other mode-s messages being received by the Payload system. Although the system has the ability to detect the other ones and skip the respective number of samples, the overlapping of messages is a constant and will lead to less ADS-B messages being received.

Finally, this two tests allow to prove the functionality of the algorithm under real conditions and thus giving good perspectives on what should be the real in-orbit mission.

5.1.4 Cone of Silence Real Test

The characterization of the ”Cone of Silence” is not feasible with the satellite placed on the earth’s surface as aircraft is very close to the satellite. In IST-Taguspark facility where these tests were done, it is possible to see that most aircraft are on approach to land or started to take off a few seconds ago, which lead to a 2/3km altitude. At an altitude of 400Km, the antenna should produce a cone of silence with 60-80km width. At 3km, the ”Cone of Silence” has only a 1.5-2km width, which is clearly a very small area to identify the ”Cone of Silence” produced by the antenna of the aircraft. Moreover, the time frame of not receiving messages from that plane can also be an overlapping of messages from other aircraft.

The conclusion is that this test can only be done when the satellite is active and in the LEO orbit. Therefore, the characterization of the satellite can only occur on the time frame of the mission.

5.2 Algorithm Performance

To test the time that the algorithm takes to process each function and thus understanding if it is possible to acquire and decode at the same time, one microcontroller timer has been used. This timer is useful to clearly understand the time that a function or a couple of functions are taking to run.

The first thing to do is to clarify the tools used to compile and obtain the several results. The GNU Tools for ARM Embedded Processors namely the GNU Compiler (GCC) compiler is used to compile the payload code. The GNU Tools are open source tools that allow programming ARM-Cortex targets directly. Additionally, it was necessary to set up the JLINK GDB SERVER that allows debugging each core properly, which is run inside a Raspberry Pi that is directly connected to the Flatsat system. This system is a way to properly have the satellite boards connected over the PC/104 connector but placed side by side instead of being mounted one at the top of the other just like the final configuration. This implementation brings a lot of advantages in the development and testing phases as it is possible to access and debug specific signals on the several hardware components. In the testing phase, it should be needed to have access to the various signals inside each board, and this architecture will allow improving that access. For the payload, the advantage was to check the input signal of the ADCHS, after the payload receiver. This allowed checking the messages using an oscilloscope as already shown in Figure 5.1.

To properly compile over the Raspberry Pi, an SSH tunnel is used to directly access the Raspberry pi and compile over the Payload board. With only the ARM Cortex-M4 and ARM Cortex-M0 connected, the algorithm have gone through several improvements.

During the algorithm performance testings, the samples come from two buffers, where one of the buffers does not include any message and the other one included one message.

The time DMA needs to fill out a buffer is 681 μs. This should be the maximum available time to work with an entire buffer, as the payload system must process one entire buffer in less than this time.

Table 5.10 shows the time of the Dump1090 code compared with the new algorithm that has been created. The several times where retrieved using a timer was setup with the help of the clock of the processor at 204MHz.

For the first algorithm on the list, the dump1090 code with the changes needed to detect and decode messages at the same time has been done. The algorithm at this moment was not dealing with the double buffer, which if used, would cause the loss of messages. With the conditions already described, the times retrieved from the timer served as a reference to start improving the algorithm and adding extra functionalities.

The next step was to test the different number of messages inside the same test buffer. The result here is that each new message inside the buffer will add approximately 20 μs of processing time.

For the double buffer strategies, two tests have been performed in order to understand the extra time

	Message in Buffer	No Message in Buffer
Dump1090 algorithm	200μs	146μs
Dump1090 algorithm with 2 messages in same buffer	211μs	146μs
Dump1090 algorithm with 3 messages in same buffer	231μs	146μs
Dump1090 algorithm with 4 messages in same buffer	251μs	146μs
First iteration of the new Algorithm	276μs	260μs
First iteration with Message occupying both buffers	282μs	260μs
Double buffer with improvements in functions	345μs	268μs

Table 5.10: Times on several improvements in algorithm

necessary to process a message that has samples on both buffers and another just with samples on one buffer. For the double buffer usage with only one message inside the buffer, the CPU usage was about 40%. For the double buffer usage with a message that contains samples in both buffers, the CPU usage was about 41%. The improvements did not add much CPU usage and should lead to much better results as messages can now be found between buffers.

The final iteration that also deals with the double buffer usage and message processing revealed that the medium usage time of the buffer with message inside is 345 μs. This test revealed that 51 % of the CPU is being used to process the entire buffer and the respective message.

In terms of particular functions, the time of reconstructing a message over one buffer is 63 μs. The FindFirstLow function and a couple of tries on the TryAlignPreamble function for the entire buffer take 268 μs. The available free time for the CPU to do other functions, like communications, telecommands, housekeeping or telemetry, ranges from 330 to 410 μs. This available time will be essential to execute the other functions and thus not stopping the acquisition and reception of messages.

Furthermore, all the optimization levels have been tested. The best results occurred using the GCC -03 optimization level that takes more time to compile but allows to reduce the processing time. Also, static variables are being used whenever it is possible to reduce the processing times. Furthermore, the correct sizes of the variables have been done in order also to reduce the memory usage.

The test for the Double buffer with improvements in functions has been done for all the different available optimization levels. Figure 5.3 shows the average time to process a buffer, the number of decoded messages and the number of received messages for each one of the optimization levels.

The result of the tests revealed that -Og and -02 optimization could not be used to meet the buffer requirements, as there is no available time using this optimization levels to process an entire buffer as they need more than 681 μs. For the -01 optimization, there is available time to process the buffer. However, with this optimization, there is no time to do any other tasks or functions. The optimization levels -03 and -04 produced the same results, and the optimization -03 is the one that will be used in this project. This optimization level allows to vectorize loops and uses function inlining, which increases the speed of the executable but also increases its size. With the memory available in the system, the size of the executable will not be a problem.

The last improvement that has been done is related to the real test. After the reconstruction of the

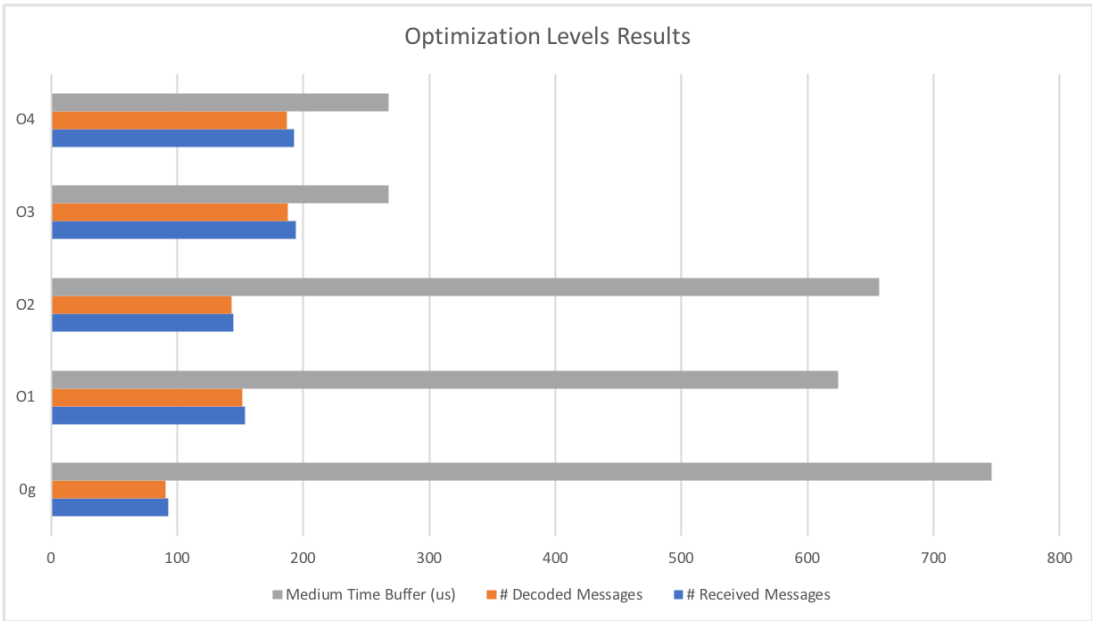

Figure 5.3: Several Optimization levels for the ADS-B receiver

first 5 bits of a message, a condition to verify the type of message exists. If the message is an ADS-B the algorithm shall continue. However, if the type of the message is not ADS-B it should move the number of samples corresponding to the bits (112*6 or 52*6 samples). This allows saving processing time in the real testings.

Table 5.11 shows the final values of the CPU usage on the payload system.

	Medium CPU Usage	Available CPU
Messages in Buffer	51%	49%
No Message in Buffer	42%	58%

Table 5.11: Time available after the receiver chain

In terms of CPU usage it is possible to understand that the payload is only using half of the capacity dealing with the receiver chain. Even with only a single operation on the filtering and missing the house-keeping and telemetry values it is possible to understand that the ARM Cortex-M4 has the capacity to deal with this minor functions that shall only occur when the processor finish the processing of the algorithm.

5.3 Energy Consumption

Energy consumption was an important metric to verify during the entire development, as the Payload system should not consume more than 0.8 W as described in section 3.1.1. Regarding this requirement, it was necessary to verify if the payload can accomplish this value. The power budget of the satellite has been defined on an internal ISTsat-1 document [38].

For that purpose, tests have been performed to verify the different consumption of energy during the

development process of the software. These tests are divided into:

1. Only ARM Cortex-M4 running;
2. ARM Cortex-M4 + ARM Cortex-M0_APP running;
3. Triple Core running;

The first three tests served as default values. The measures were done using a while loop to test energy consumption. In all processors, the frequency was set to 204Mhz. There was also a test to understand the power consumption added by connecting the DMA and the ADCHS on the M4 core. The ARM Cortex-M4 with DMA powered on showed a total consumption of 673mW. Furthermore, the ADCHS powered on showed an extra consumption of 12mW. Table 5.12 shows the consumption for each one of the tests.

	Consumption
ARM Cortex-M4 running	627mW
ARM Cortex-M4 running + DMA	673mW
ARM Cortex-M4 running + DMA + ADCHS	685mW
ARM Cortex-M4 + ARM Cortex-M0_APP running	674mW
Triple-Core running	687mW

Table 5.12: Energy consumption for the several cores and peripherals

In terms of energy consumption, the power budget document declares that the average consumption of energy while working on about 15% of the orbit time shall not exceed 150mW. On the peak, it should not be above the 1000mW. The orbit of the ISTSat-1 will take around 92.7 minutes with 36 minutes in eclipse. In a standard orbit, the payload can be working on a maximum of 57 minutes. Furthermore, the payload mission will only take a maximum of 10 minutes which allows an understanding that it will only work in maximum consumption for about 11% of the working time. To reduce power consumption, it will be possible to power off the ADCHS and the DMA allowing to reduce significantly the power consumption. Also, the frequency of the payload cores shall be decreased as it is not necessary to be in the 204Mhz frequency. The decrease in the frequency will allow saving energy consumption on the payload system. However, the decision of the mission time will depend on the status of the payload.

Chapter 6

Conclusions

The presented work proposes a design solution in order to overcome the problem of receiving and decoding samples at the same time in the Payload subsystem of the ISTsat-1. The proposed solution presents a couple of tests that proves the feasibility of the code and thus leading to understand that the chosen ARM Cortex-M4 shall have the ability to perform the decoding and acquisition of samples and still have time for other processing. There are good perspectives of receiving and decoding the several messages in the LEO orbit which are the expectations.

6.1 Achievements

The major lessons learned regarding this thesis are related with the complete understanding of the ADS-B system and the understanding of the LPC4370 processor. The final solution has allowed to prove that testing is very important to verify if the system can handle the different requirements. Furthermore, the several tests performed revealed that messages can be received and decoded by the receiver chain.

6.2 Future Work

The future work on this for this project is related with the acquisition of messages. There is always room for improvements and new ideas that can help achieving better results on the process of receiving ADS-B messages. Additionally, GS segment shall be prepared to receive the messages that have been downlinked from the COM subsystem. As part of another student thesis, an implementation of a small application capable of reconstructing the path of the planes shall be made according to the fields of an ADS-B message. Besides, this small application should handle with the different messages and will be a very important tool to check the mission measures.

For the mission purpose there is also future work to do, like the characterisation of the ”Cone of Silence” that will only be possible using real mission data. It will also be important to compare the received messages with the Ground based datasets of messages and verify the performance of the antenna. At the end of those results, there should be available data to compare with another space based ADS-B

systems, like the Proba-V mission, and have a notion of how well the antenna has performed.

References

[1] Erik Kulu: Overview of CubeSats. available at: http://www.nanosats.eu accessed in 2018-04-27.

[2] Brian Dunbal: CubeSats Overview. available at: https://www.nasa.gov/mission_pages/cubesats/overview (February 2018) accessed in 2019-03-20.

[3] ESA: "Fly your satellite! programme". available at: https://www.esa.int/Education/CubeSats_-_Fly_Your_Satellite/Fly_Your_Satellite!_programme#description (October 2018) accessed in 2019-03-20.

[4] José Freitas and Rui Rocha and Pedro Gameiro and Alexandre Silva: ISTsat-1 Mission Description Document (October 2017)

[5] Flight Safety Foundation: Benefits Analyses of space-based ADS-B (June 2016)

[6] Aerion: Aireon LLC company (2018) available at: https://aireon.com.

[7] FAA Government: Equip ADS-B. available at: https://www.faa.gov/nextgen/equipadsb/ (May 2019) accessed in 2019-05-02.

[8] Boeing: New Air Traffic Surveillance Technology. Aero (2010) available at: http://www.boeing.com/commercial/aeromagazine/articles/qtr_02_10/2/.

[9] Abdulrazaq Abdulaziz and Abdulmalik S. Yaro and Ashraf A. Adam and Mahmoud T. Kabir and Habeeb B. Salau: Optimum Receiver for Decoding Automatic Dependent Surveillance Broadcast (ADS-B) Signals. Technical report, Faculty of Electrical Engineering, Universiti Teknologi, Malaysia; Department of Electrical and Computer Engineering, Ahmadu Bello University, Zaria, Nigeria; School of Engineering and Engineering Technology, Federal University of Technology, Minna, Nigeria (2015)

[10] Junzi Sun: The 1090MHz Riddle. Open-access book: https://mode-s.org/decode/ accessed in 2019-03-30.

[11] PROGRAMME, E.A.T.M.: Clarification mode s transponder in an airport/a-smgcs environment (2015)

[12] Mitch Launius: The Impact of Space-Based ADS-B on International Operations available at: https://ops.group/blog/the-impact-of-space-based-ads-b-on-international-operations/ accessed in 2019-03-30.

[13] Reporter, T.D.: Obaseki flays theft of airport facilities, orders investigation. This Day (November 2018) available at: https://www.thisdaylive.com/index.php/2018/11/08/obaseki-flays-theft-of-airport-facilities-orders-investigation/.

[14] DLR: German Aerospace Center available at: https://www.dlr.de/.

[15] T. Delovski and K. Werner and J. Bredemeyer: ADS-B over Satellite - Global Air Traffic Surveillance from Space (2014)

[16] T. Delovski and K. Werner and T. Rawlik and J. Behrens and J. Bredemeyer and R. Wendel: ADS-B over Satellite The World's first ADS-B receiver in Space (2014)

[17] Herbert J. Kramer: Observation of the Earth and Its Environment: Survey of Missions and Sensors available at: https://directory.eoportal.org/web/eoportal/satellite-missions/a/ads-b.

[18] GomSpace: About GOMspace available at: https://gomspace.com.

[19] Lars K. Alminde and Karl Kaas and Morten Bisgaard and Johan Christiansen and David Gerhardt: GOMX-1 Flight Experience and Air Traffic Monitoring Results (2014) available at: https://directory.eoportal.org/web/eoportal/satellite-missions/g/gomx-1.

[20] GomSpace: GOMX-3 (GomSpace Express-3) (2014) available at: https://directory.eoportal.org/web/eoportal/satellite-missions/g/gomx-3.

[21] GomSpace: GomX-4 (GomSpace Express-4) Mission (2016) available at: https://directory.eoportal.org/web/eoportal/satellite-missions/g/gomx-4.

[22] GomSpace: NanoCom ADS-B available at: https://gomspace.com/UserFiles/Subsystems/flyer/NanoComADS-B_HIGH.pdf.

[23] Iridium: Iridium to Revolutionize Global Air Traffic Surveillance With the Launch of Aireon(SM) (June 2012) available at: http://investor.iridium.com/releasedetail.cfm?ReleaseID=684218.

[24] Iridium: Hosted payloads of other customers on Iridium NEXT available at: https://directory.eoportal.org/web/eoportal/satellite-missions/i/iridium-next.

[25] Aerion Company: Resources available at: https://aireon.com/resources/overview-materials/technical-specifications/.

[26] Aerion Company: What is Aireon ALERT available at: https://aireonalert.com/overview.html.

[27] Flight Radar: FlightRadar24 available at: https://www.flightradar24.com.

[28] AirNav: AirNav RadarBox available at: https://www.radarbox24.com/.

[29] Flight Aware: Flight Aware available at: https://pt.flightaware.com/.

[30] Flight Aware: Flight Aware ADS-B coverage available at: https://flightaware.com/adsb/coverage.

[31] RTD Embedded Technologies, Inc.: What is PC104? available at: https://www.rtdusa.com/PC104/default.htm accessed in 2019-03-25.

[32] Tomás Almeida: ADS-B Antenna Design Definition. Revision A (October 2017)

[33] José Freitas and Tomás Almeida and Rui Rocha: ADS-B Payload Design Definition. Revision A (September 2017)

[34] Salvatore Sanfilippo: Dump1090 available at: https://github.com/antirez/dump1090.

[35] Alexandre Silva: ISTsat-one Control Protocol. Revision 6 (September 2017)

[36] NXP: LPC-Link2 available at: https://www.nxp.com/support/developer-resources/software-development-tools/lpc-developer-resources-/lpc-microcontroller-utilities/lpc-link2:OM13054.

[37] Evan Nemerson: UNIT TESTING FRAMEWORK FOR C. available at: https://nemequ.github.io/munit/#about.

[38] Nuno Ramos: ISTsat-1 Power and Energy Budget. Revision 3 (January 2018)

Appendix A

Identification Messages

Figure A.1 gives the several fields that are available on an identification message and under the Data segment of an ADS-B message [10].

```
+------+------+------+------+------+------+------+------+------+------+
| TC,5 | EC,3 | C1,6 | C2,6 | C3,6 | C4,6 | C5,6 | C6,6 | C7,6 | C8,6 |
+------+------+------+------+------+------+------+------+------+------+

TC: Type code
EC: Emitter category
C*: A character
```

Figure A.1: ADS-B Identification Message fields

Each one of the characters can be decoded by the respective decimal value as shown in table A.1

Character	Decimal Value
A-Z	1-26
_	32
0-9	48-57

Table A.1: Construction of characters of Identification message

Appendix B

Velocity Ground Messages

Figure B.1 gives the several fields that are available on a velocity ground message and under the Data segment of an ADS-B message [10]. Besides, this specific message shall not be decoded on mission as they are only sent when the aircraft are on land and the satellite shall only receive this messages sporadically.

MSG Bits	Data Bits	Len	Abbr	Content
33 - 37	1 - 5	5	TC	Type code
38 - 40	6 - 8	3	ST	Subtype
41	9	1	IC	Intent change flag
42	10	1	RESV_A	Reserved-A
43 - 45	11 - 13	3	NAC	Velocity uncertainty (NAC)
46	14	1	S_ew	East-West velocity sign
47 - 56	15 - 24	10	V_ew	East-West velocity
57	25	1	S_ns	North-South velocity sign
58 - 67	26 - 35	10	V_ns	North-South velocity
68	36	1	VrSrc	Vertical rate source
69	37	1	S_vr	Vertical rate sign
70 - 78	38 - 46	9	Vr	Vertical rate
79 - 80	47 - 48	2	RESV_B	Reserved-B
81	49	1	S_Dif	Diff from baro alt, sign
82 - 88	50 - 56	7	Dif	Diff from baro alt

Figure B.1: ADS-B Velocity Message fields

Appendix C

Velocity Airspeed Messages

Figure C.1 gives the several fields that are available on a velocity airspeed message and under the Data segment of an ADS-B message [10].

DATA Bits	MSG Bits	Len	Abbr	Content
33 - 37	1 - 5	5	TC	Type code
38 - 40	6 - 8	3	ST	Subtype
41	9	1	IC	Intent change flag
42	10	1	RESV_A	Reserved-A
43 - 45	11 - 13	3	NAC	Velocity uncertainty (NAC)
46	14	1	S_hdg	Heading status
47 - 56	15 - 24	10	Hdg	Heading (proportion)
57	25	1	AS-t	Airspeed Type
58 - 67	26 - 35	10	AS	Airspeed
68	36	1	VrSrc	Vertical rate source
69	37	1	S_vr	Vertical rate sign
70 - 78	38 - 46	9	Vr	Vertical rate
79 - 80	47 - 48	2	RESV_B	Reserved-B
81	49	1	S_Dif	Difference from baro alt, sign
82 - 88	50 - 66	7	Dif	Difference from baro alt

Figure C.1: ADS-B Velocity Message fields

Appendix D

Position Messages

Figure D.1 gives the several fields that are available on a position message and under the Data segment of an ADS-B message [10]. The position messages are always sent in pairs and that's the reason the Compact Position Reporting (CPR) odd/even flag is used to determine if it is an odd or even message. CPR function is responsible to determine the correct Latitude and Longitude of the aircraft using both messages.

Data Bits	MSG Bits	N-bit	Abbr	Content
33 - 37	1 - 5	5	TC	Type code
38 - 39	6 - 7	2	SS	Surveillance status
40	8	1	NICsb	NIC supplement-B
41 - 52	9 - 20	12	ALT	Altitude
53	21	1	T	Time
54	22	1	F	CPR odd/even frame flag
55 - 71	23 - 39	17	LAT-CPR	Latitude in CPR format
72 - 88	40 - 56	17	LON-CPR	Longitude in CPR format

Figure D.1: ADS-B Position Message fields