

SOFTWARE DEFINED AIS RECEIVER FOR AAUSAT3

GROUP 651

Troels Jessen
Jeppe Ledet-Pedersen
Hans Peter Mortensen



DEPARTMENT OF ELECTRONIC SYSTEMS
AALBORG UNIVERSITY
SPRING 2009



The Faculties of Engineering, Science and Medicine

Department of Electronic Systems

Frederik Bajers Vej 7

Phone: +45 99 40 86 00

<http://es.aau.dk>

Title: Software Defined AIS Receiver for AAUSAT3

Theme: Robust Communication

Project period:

February 2nd - June 3rd, 2009

Project group: 09gr651

Group members:

Troels Jessen

Jeppe Ledet-Pedersen

Hans Peter Mortensen

Supervisor:

Ole Kiel

Number of copies: 6

Number of pages: 60

Appended documents:

(6 appendices, 1 CD-ROM)

Total number of pages: 82

Finished: June 3rd, 2009

Abstract:

This report documents the development of a software defined AIS receiver for the AAUSAT3 student satellite. AIS is a standardized protocol designed to enhance safety at sea by automatic exchange of ship information. The AIS receiver and a prototype of AAUSAT3 will be tested on the BEXUS high-altitude balloon flight in October 2009.

The report begins with an analysis of the AIS standard with special focus on the physical and data link layers. The GMSK modulation scheme, line coding and HDLC framing are analyzed to identify the requirements for the receiver.

Based on these requirements, a non-coherent demodulator with support for packet detection and frequency estimation is designed and implemented in MATLAB. A modulator with adjustable signal-to-noise ratio is developed for testing purposes, and the MATLAB receiver is successfully tested for receiving both simulated and real-life AIS signals.

A prototype PCB with a commercial radio front-end and Analog to Digital Converter has been constructed, and a Digital Signal Processor is selected for implementation for the BEXUS flight. The mechanical design for the balloon flight is documented in a separate section.

Finally, the acceptance test shows that the implemented algorithms are suitable for receiving and decoding AIS signals. Further work is still needed to port the receiver to the DSP.

Table of Contents

Preface	v
1 Introduction	1
1.1 The AAUSAT3 Project	1
1.2 The NAVIS Project	3
1.3 Project Scope	4
2 Analysis	5
2.1 Automatic Identification System	5
2.2 Wireless Communication	12
2.3 Line Coding	13
2.4 Channel Estimation	13
2.5 Digital Modulation	14
3 Problem Description	23
3.1 Requirement Specification	23
3.2 Test Specification	24
4 Design	25
4.1 Information Source	26
4.2 AIS Data Encapsulation	26
4.3 GMSK Modulator	28
4.4 AWGN Channel	29
4.5 Packet Detection	29
4.6 Frequency Estimation	31
4.7 GMSK Demodulator	32
4.8 AIS Data Extraction	36
5 Proof of Concept	39
6 Implementation	43
6.1 VHF Antenna	43
6.2 Low Noise Amplifier	43
6.3 SAW Filter	45
6.4 Power Splitter	47
6.5 RF Front-end	47
6.6 Analog to Digital Converter	48
6.7 Digital Signal Processor	49
6.8 Analog Devices ADSP-BF537 Architecture	50
6.9 PCB Schematics and Layout	51
6.10 Experiment Setup	52

TABLE OF CONTENTS

7	Acceptance Test	57
8	Conclusion	59
8.1	Further Work	60
A	Interface Control Documents	61
B	Hardware diagrams	65
C	Component List	69
D	Work agreement (Danish)	71
E	Contents of the CD	77
F	Abbreviations	79
	Bibliography	81

Preface

This report serves as documentation for the development of a software defined radio for receiving signals from the Automatic Identification System (AIS). The project is carried out by three 6th semester Communication Systems students from Aalborg University in Denmark.

The AIS receiver will be the main payload for the AAUSAT3 student satellite, which is planned to be launched in the spring of 2011. Prior to the launch, the receiver will be tested on the BEXUS 8 (Balloon EXperiment for University Students) balloon flight in October 2009. The BEXUS project is jointly sponsored by the European Space Agency (ESA), the Swedish Space Corporation (SSC), the Swedish National Space Board (SNSB) and the German Aerospace Center (DLR) and allows European university students to test scientific experiments in high altitude.

The semester has been carried out in close cooperation with group 09gr650, who has developed an alternative AIS receiver for the satellite. The workload of the semester has been wider than expected for a normal semester project, as the whole experiment for the BEXUS flight has been carried out by these two groups. The cooperation of the groups has been named NAVIS (North Atlantic Vessel Identification System) and has been approved by the Study Board, as described in section D on page 71. An additional outcome of the NAVIS project is the SED (Student Experiment Description) which provides an elaborate description of the design and construction of the experiment. The SED is attached as a separate report and contains information about the NAVIS experiment not otherwise related to this project. The main purpose of the SED is to provide this information to the BEXUS involved partners.

Chapter Organization

Chapter 1 serves as an introduction to readers who are unfamiliar with AAUSAT3. The chapter also describes the NAVIS and BEXUS projects in further details and outlines the scope of this project.

Chapter 2 analyzes the AIS standard and provides an introduction to wireless communication systems. A number of digital modulation schemes including Gaussian Minimum Shift Keying (GMSK) used in AIS is explained in detail, with focus on the AWGN channel and bit error probability.

Chapter 3 contains the project description and requirement specification of the system.

Chapter 4 describes the design of algorithms used for packet detection, bit synchronization and the non-coherent GMSK demodulation.

Chapter 5 contains a "proof of concept" implementation of the AIS receiver using a National Instruments Sample Card and MATLAB.

Chapter 6 documents the implementation of hardware and software, including schematics, PCB layout and DSP selection for further development.

Chapter 7 lists the acceptance test of the system.

Chapter 8 contains the conclusion which sum up the results of the report.

The appendices contain hardware diagrams, component lists and Interface Control Documents for the AIS subsystem.

Conventions

References to external sources used in the project are enclosed in brackets. An example reference to a paper published in 2009 will thus be [Example 09]. The bibliography in page 81 contains a table with information about all sources used. All sources are also included on the CD. Figures, tables and code excerpts are numbered for use in cross references. In some cases, the size of the figures and tables makes it necessary to place it on the following page. The unit Nautical Mile is abbreviated nmi as recommended by IEEE [IEEE 07].

Acknowledgements

The authors would like to thank Aalborg University, the Danish Maritime Safety Administration and the rest of the AAUSAT3 sponsors. Without their help, the students satellite project would not have been possible. ESA, SSC, SNSB and DLR are thanked for selecting the NAVIS project for inclusion on the BEXUS flight and allowing the AIS receivers to be tested in high altitude conditions. Jens Dalsgaard Nielsen, Johan Christiansen and the rest of the AAUSAT3 system engineering group are thanked for providing valuable input and helping out with the preparation of the NAVIS experiment. Daniel Winther Uhrenholt has been a great help with the experiments mechanical design and construction.

The authors of the report can be contacted at {09gr650, tjessen, jledet, hp000}@es.aau.dk

Thank you for reading our report!

Troels Jessen

Jeppe Ledet-Pedersen

Hans Peter Mortensen

Chapter 1

Introduction

This chapter serves as a short introduction to readers who are unfamiliar with the AAUSAT3 and NAVIS projects. The final section of the chapter outlines the scope of this project report.

1.1 The AAUSAT3 Project



AAUSAT3 is the third student satellite developed at Aalborg University. The satellite is the successor to AAUSAT-II which was launched in 2008 and, at the time of writing, has been operational for more than a year — longer than any other Danish student satellite. The AAUSAT3 project was initiated in the fall of 2007 and the satellite is expected to be launched in the first half of 2011.

The satellite project is jointly sponsored by Aalborg University and the Danish Maritime Safety Administration (DaMSA) who is responsible for the safety in the seas around Denmark, Greenland and the Faroe Islands. In the Danish waters, DaMSA monitors the ship traffic using the Automatic Identification System (AIS) by using strategically placed ground stations and buoys. The ground stations are placed along the Danish coast line and are used to receive identification signals from ships, as well as act as electronic lighthouses.

AIS is a data exchange protocol standardized by the International Telecommunication Union (ITU) and designed to enhance safety at sea by automatic exchange of ship identification data. All ships with a gross tonnage of more than 300 tons and all ships carrying passengers is required by law to have an AIS transponder on board. For all other ships it is an optional safety feature. At regular intervals, AIS transponders broadcasts info such as registration number, position, speed, destination and name to nearby ships as well as shore based stations. The system is designed to function autonomously and allow ships to exchange information without influence of an operator. AIS defines numerous packet types, so not all information is contained in each transmitted packet. Section 2.1 describes the technical details AIS in further detail.

AIS has several advantages compared to conventional radar systems. First of all, the AIS signals can be received even if the transmitting ship is hidden by landmass, where traditional radar systems cannot detect vessels. AIS signals also contains information about more than just ship position, and the on board AIS receiver is thus often coupled directly to the ships radar monitor. This allows the ships crew to view the position, course and speed of nearby vessels in

one place. Not all ships are required to have AIS equipment on board, so the system does not give a complete picture of nearby ships. AIS is thus not intended to replace traditional radar or collision avoidance systems, but to function as an additional safety feature.

Around Greenland, which is also DaMSAs area of responsibility, a ground station based network of AIS receivers is impractical. Due to Greenland's extensive coast line, the number of ground stations required would be very high. Furthermore, the environment in Greenland is so hostile that maintenance of the ground stations will be a considerable challenge. The DaMSA, through the sponsorship of AAUSAT3, therefore wishes to investigate the possibility of receiving AIS signals in space, as a satellite based receiver will be able to cover a much wider area than conventional ground stations.

1.1.1 Satellite Structure

The primary payload of AAUSAT3 will be two student developed AIS receivers, designed to evaluate the possibility of receiving AIS in space. The reception of AIS signals in space impose a number of challenges, such as increased noise and packet collisions, which must be addressed. In addition to the AIS receivers, the satellite consists of seven subsystems that are all built by students at Aalborg University. Seven subsystems are located inside the satellite, while two are located on ground. This is illustrated in Figure 1.1.

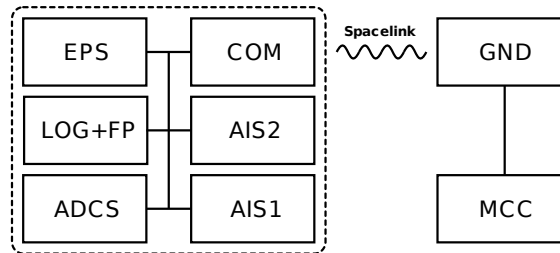


Figure 1.1: AAUSAT3 subsystem structure and internal CAN bus

The main functionalities of the individual subsystems are listed below:

EPS The Electronic Power Supply (EPS) is responsible for charging of the batteries with power from the solar panels, as well as power conversion and distribution.

LOG The Logging subsystem (LOG) is responsible for logging data and states from all subsystems.

FP The Flight Planner (FP) schedules flight operations and subsystem functionalities, allowing the operator to plan events for execution when the satellite is out of range.

ADCS The Attitude Determination and Control System (ADCS) detumbles the satellite and stabilizes the satellites attitude. This is important when communicating using directional antennas.

COM The Communication subsystems (COM) provides the space link used for transmission of telecommands and telemetry.

GND The Ground Station (GND) tracks the satellite and positions the antennas on ground. GND also controls the radio to correct for Doppler shift.

MCC The Mission Control Center allows the satellite operators to issue telecommands and presents telemetry data received from the satellite.

AIS The AIS receivers are the main payload, responsible for investigating the possibility of receiving AIS in space.

The subsystems internal to the satellite communicates using a Controller Area Network (CAN) bus, which is traditionally used for automotive applications.

1.2 The NAVIS Project



The North Atlantic Vessel Identification System (NAVIS) is a subproject of AAUSAT3. NAVIS consists of a prototype built for testing on a balloon flight in October 2009 and includes a subset of the AAUSAT3 subsystems. The balloon flight is made possible through the BEXUS (Balloon EXperiments for University Students) program, sponsored by ESA, DLR, SSC and SNSB. BEXUS allows European students to test scientific experiments in high altitude.

BEXUS experiments are launched on a balloon with a total volume of 12,000 m³ to a maximum altitude of approximately 35 km, depending on total experiment mass (40–100 kg). The flight duration is 2–5 hours. EuroLaunch, a cooperation between the Esrange Space Center of the Swedish Space Corporation (SSC) and the Mobile Rocket Base (MORABA) of DLR, is responsible for the campaign management and operations of the launch vehicles, while experts from ESA, SSC and DLR provide technical support to the student teams throughout the project. The balloons are launched from Esrange Space Center in Kiruna, Sweden.

The NAVIS experiment contains prototypes of the EPS and COM subsystems, as well as two AIS receivers based on different methods: AIS1 is based on a COTS hardware transceiver for demodulation of the AIS signals, while AIS2 samples the down converted output of a similar chip, but performs the demodulation in a DSP (Digital Signal Processor). AIS1 is developed by group 09gr650. The advantage of using a hardware based solution is lower power consumption. However, the hardware transceivers optimum package detection requires a training sequence of 48 bits or more[Analog Devices 07a, p. 36], but the AIS standard only include 24 training bits (section 2.1).

The goal of this project (09gr651) is to develop a working software defined AIS receiver (AIS2) that is expected to be used on AAUSAT3. The receiver must be developed such that it fits the requirements of the NAVIS experiment. Another reason for developing a software defined AIS receiver for AAUSAT3 is, that when the satellite is in space, there are no possibility of changing hardware. A software defined receiver allows for upload of new software to existing hardware, which is usable for optimization and reconfiguration of algorithms. At the same time, a software defined receiver is more feasible to detect the short AIS bursts, to compensate for Doppler shift and use interference cancellation and soft decision to improve the bit error rate.

1.2.1 Time Schedule

The following list gives an overview of the NAVIS project time line and involvement.

Dec 2008 BEXUS experiment application sent to ESA.

Feb 2009 Project accepted by ESA (4 students at selection workshop at ESTEC, The Netherlands).

Feb 2009 Two 6th semester groups (09gr650 and 09gr651) officially working on the BEXUS flight.

Mar 2009 Submission of status report to ESA.

Mar 2009 Preliminary Design Review (4 students at training week at DLR, Oberpfaffenhofen).

May 2009 Submission of status report to ESA.

Jun 2009 6th semester project report submission to Aalborg University.

Jun 2009 Critical Design review (4 students at ESTEC, The Netherlands).

Sep 2009 Experiment delivery to DLR.

Oct 2009 Flight Campaign (Esrange Space Center, Kiruna, Northern Sweden).

Jan 2010 Final experiment report submission to ESA.

1.3 Project Scope

Figure 1.2 illustrates the general flowchart of the software defined AIS receiver developed in this project. The signals received at the antenna are fed to the RF front-end which downconverts the signal to a frequency suitable for sampling with the Analog to Digital Converter (ADC). The sampled data is used as input for the Digital Signal Processor (DSP), which demodulates and decodes the AIS packets.

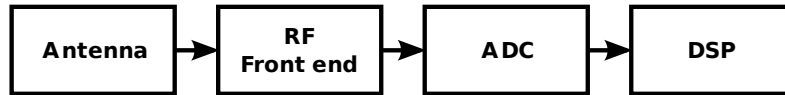


Figure 1.2: General flowchart diagram of the software defined AIS receiver.

The projects primary focus is the physical and link layers of the AIS standard. The physical layer defines the modulation and line coding schemes used, while the link layer provides framing and error detection mechanisms. The TDMA (Time Division Multiple Access) channel access method will only be mentioned briefly, as the access scheme is most important when developing a transmitter that will participate in the scheme. Detailed interpretation of the received packets is outside the scope of this project, and only executed to the extent needed to verify the received data.

Analysis

2.1 Automatic Identification System

This section describes the technical details of the Automatic Identification System (AIS). As described in the report introduction in Chapter 1.1, AIS is a ITU-R standardized system designed to improve safety at sea. Ships equipped with an AIS transponder continuously broadcasts navigational data to other ships and ground based stations. The current AIS standard is documented in ITU-R Recommendation M.1371-3[ITU 07] which is used as the primary source for this section. The description is concentrated on receiving AIS messages, and only briefly considers the various Time Division Multiple Access (TDMA) schemes used for self-organized Medium Access Control (MAC).

AIS messages either contain static or dynamic information. Static information includes the name of the transmitting ship, call sign, type of ship etc. Dynamic information contains data such as position and heading. Both packet types contains the ships MMSI (Maritime Mobile Service Identity) number, which uniquely identifies the ship. MMSI numbers are 9 digits long, of which the first 3 are the Maritime Identification Digits (MID) that indicates in which country the ship is registered. The danish MID are 219 and 220. Static information packets are transmitted every 6 min. The reporting interval of dynamic information depends on if the ships is at anchor or sailing. Table 2.1 lists the reporting interval for dynamic data. Ground stations transmits information every 10 s.

Ship's dynamic conditions	Nominal reporting interval
Ship at anchor or moored and not moving faster than 3 knots	3 min
Ship at anchor or moored and moving faster than 3 knots	10 s
Ship 0-14 knots	10 s
Ship 0-14 knots and changing course	3 1/3 s
Ship 14-23 knots	6 s
Ship 14-23 knots and changing course	2 s
Ship >23 knots	2 s
Ship >23 knots and changing course	2 s

Table 2.1: Reporting interval for dynamic information.

The AIS standard classifies transponders in two categories: Class A and class B. The main difference between the classes is the maximum transmission power. Class A transponders are required to transmit with 12.5 W while Class B only transmit with 1 W. Class B transponders are also not required to support Self-Organized TDMA MAC scheme and may use the simpler Carrier-

Sense TDMA (CSTDMA) instead. Class A AIS transponders should be able to both receive and transmit short safety related messages, containing important navigational or meteorological notifications. Class B receivers should only be capable of receiving these notifications. The rest of this section focuses on Class A transponders.

The AIS standard covers layer 1 to 4 in the OSI (Open Systems Interconnection) Reference Model. This description only covers the physical and link layers. The network layer is responsible for prioritization of messages, distribution of packets between transmission channels and data link congestion resolution, and is mainly relevant when developing AIS transmitters. The transport layers primary function is to implement sequencing and split messages that occupy multiple AIS slots. The AIS receiver developed in this project, does not support multi-slot binary messages.

2.1.1 Physical Layer

The physical layer is responsible for NRZI (Non-Return to Zero Inverted) encoding the data and applying the GMSK (Gaussian Minimum Shift Keying) modulation scheme for transmission on the VHF data link (VDL). NRZI and GMSK are explained in detail in Sections 2.5.4 and 2.3. Table 2.2 lists the main parameters of the physical layer. AIS signals are transmitted on maritime VHF channels 87B and 88B at 161.975 and 162.025 MHz. Each channel has a bandwidth of 25 kHz. AIS transponders should listen for AIS packets on both channels but alternate its transmissions between the channels. The channel distribution algorithm is implemented in the network layer. The AIS standard does not include any forward error correction, interleaving or bit scrambling mechanisms.

Parameter Name	Setting	Unit
AIS channel 1	161.975	MHz
AIS channel 2	162.025	MHz
Modulation scheme	GMSK	
Carrier frequency error	± 500	Hz
Transmit output power	12.5	W
Modulation index	~ 0.5	
Transmit BT product	~ 0.4	min
Receive BT product	~ 0.5	max
Bit rate	9600	bit/s
Maximum bit rate deviation	50	ppm
Training sequence	24	bits

Table 2.2: AIS specifications.

Figure 2.1 shows the theoretical spectrum of the two AIS channels. Figure 2.2 plots a measurement of the AIS channel spectrum. This figure is created using a VHF antenna connected directly to a Rohde & Schwarz FSEA spectrum analyzer, and clearly shows peaks around the two channel frequencies. Due to the short AIS packages, the spectrum analyzer is set to peak hold, and approximately one minute is analyzed to plot the spectrum. Naturally, the noise around the channels are also peak values.

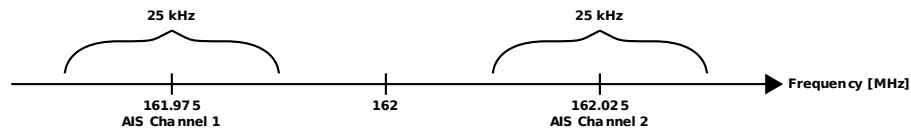


Figure 2.1: Expected AIS channel spectrum.

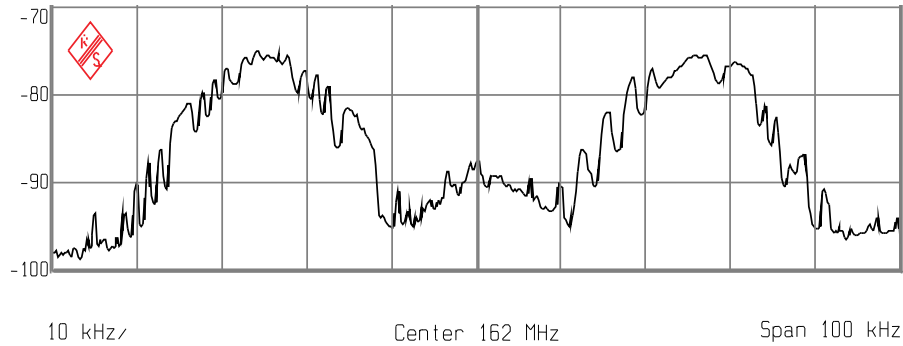


Figure 2.2: Measured AIS channel spectrum.

2.1.2 Link Layer

The link layer is divided into three sub layers: Medium Access Control (MAC), Data Link Service (DLS) and Link Manage Entity (LME). The MAC and LME layers main functionality is the channel access arbitration. The channels are shared among transmitting ships using TDMA schemes, by dividing time into transmission slots. AIS uses 1 minute frames that are subdivided into 2250 slots of 26.67 ms. At 9600 bit/s, one slot will thus allow 256 bits to be transmitted. The default transmission messages fits into one slot, but special binary messages may occupy up to 5 consecutive slots.

At open sea it is not desirable to have special "master ships" that determine when other ships are allowed to broadcast. The AIS uses a Self-Organized TDMA scheme using a common time reference. If possible, the AIS transponders are synchronized to UTC and decides on which slots it should use for transmissions depending on received transmissions from other AIS transponders. If UTC synchronization is not available, the standard defines alternate methods for time synchronization. This way, ships are arranged in synchronized TDMA zones in a self-organized manner. To ensure correct communication in the AIS, the system aims for TDMA zones not to be smaller than 20 nmi and not larger than 200 nmi. The typical size of a TDMA zone is approximately 30 nmi.

The DLS is responsible for data transfer and error detection and control. Figure 2.3 shows the AIS packet structure. AIS uses a packet format identical to the High-Level Data Link Control (HDLC), but with an additional 24 bit training sequence added to the start of the packet. The training sequence consists of alternating zeros (0) and ones (1) and aids receivers in attaining bit synchronization. The standard does not specify if the training sequence starts with a zero or a one bit, so the receiver should be able to handle both cases. Table 2.3 lists the packets fields in AIS. The total length of a data frame is 256 bits, includes including training sequence and 24 bits buffer. The packet thus fits into a single AIS transmission slot.

Training sequence	Start flag	Data	FCS	End flag	Buffer
-------------------	------------	------	-----	----------	--------

Figure 2.3: AIS data frame.

The start flag is 8 bits long and consists of a standard HDLC flag. It is used in order to detect the start of a transmission packet. The start flag consists of the bit pattern 01111110, corresponding to 0x7E in hexadecimal notation. The default data field is 168 bits long, although transponders may occupy up to a maximum of five consecutive slots for one continuous transmission. Only a single application of overhead is required for a long transmission packet. The length

Name	bits
Ramp up	8
Training sequence	24
Start flag	8
Data	168
FCS (frame check sequence)	16
End flag	8
Buffer	24
Total	256

Table 2.3: AIS data frame details [ITU 07, p. 17].

of a transmission packet should not be longer than necessary to transfer the data, i.e. the AIS transponder cannot be expected to add filler.

The Frame Check Sequence (FCS) uses a cyclic redundancy check (CRC) 16-bit polynomial to calculate a checksum as defined in ISO/IEC 3309. The CRC bits should be pre-set to one (1) at the beginning of a CRC calculation. Only the data portion should be included in the CRC calculation. The end flag is identical to the start flag.

The buffer is silence on the frequency which negates distance delay and synchronization jitter. The data and FCS fields should be subject to bit stuffing. If five consecutive 1 bits are found in the output data, the transmitter must insert an additional 0 bit. Similarly, the receiver should remove the first 0 bit following five 1 bits. Bit synchronization is used for two reasons: First of all to ensure that the HDLC start/end flags not occur in the middle of a transmission packet, but also to aid receivers to attain bit synchronization. The data stream is NRZI encoded in the physical layer. NRZI only has transitions when sending a 0 bit, so a long period of 1 bits could lead to the receiver losing synchronization. Bit stuffing ensures that a transition will occur after maximum five bits. According to statistical analysis performed by the ITU-R, most packages requires three bits or less for bit stuffing. The packet bytes are transmitted Least Significant Bit (LSB) first, in accordance with the HDLC specification.

2.1.3 AIS from High Altitude

In this section, the special circumstances when receiving AIS in space is considered. First, the worst case Doppler shift will be evaluated. The satellite is expected to be in a orbit of around 600 km above the earth, which equals a velocity of about 7500 m/sec.

The Doppler effect is given as [Serway 04, p. 525]:

$$f' = \frac{c + v_o}{c - v_s} f$$

where in worst case:

c is the speed of light, 299792458 m/s.

v_o is the speed of the observer, in this case the satellite, 7500 m/s.

v_s is the speed of the source, in this case the AIS transmitting ship, ignored (0 m/s).

f is the transmitter frequency, in this case approx. 162 MHz for both channels.

The resulting frequency, when the satellite is flying directly towards the ship:

$$f' = \frac{c - 7500 \text{ m/s}}{c} 162 \text{ MHz} = 162.004 \text{ MHz}$$

And in the case of the satellite flying directly away from the ship:

$$f' = \frac{c + 7500 \text{ m/s}}{c} 162 \text{ MHz} = 161.996 \text{ MHz}$$

This means that a frequency deviation of ± 4 kHz is expected for each AIS channel, which the receiver will have to compensate for. Note that this situation cannot be expected, as the satellite will never have the direction directly against or away from a ship.

As the satellite travels above an AIS transmitting ship, the Doppler changes. This is illustrated in Figure 2.4. Here the ship is placed in the center of the circle and the satellite moves from right to left. Note that during the transmission time of one AIS message (max 133 ms) the satellite moves 998 m (0.5 nmi). This movement will not make a significant change of the received frequency during the transmission.

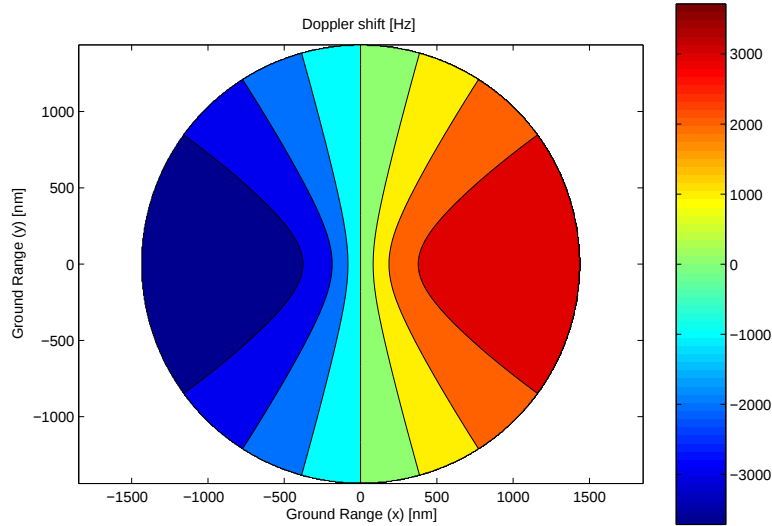


Figure 2.4: AIS Doppler shift for LEO satellite [Dahl 06, p. 17].

Another interesting part about AIS from high altitude is the collisions that is expected. The AIS system was originally intended to be used for ship-to-ship and ship-to-land transmission. This raises some challenges receiving messages from a much larger area than a ground station, due to the increased line-of-sight. This means that a satellite might receive collided AIS messages, due to the fact that a ship only synchronizes its transmissions with other ships within the line-of-sight.

2.1.4 Previous Analyses

The possibility of receiving AIS messages in space has been analyzed in a semester report by group 07gr506 from Aalborg University. The project developed simulations of the received signal strength at given height and the probability of AIS message collisions as a consequence of the enlarged footprint. This section will summarize their result with a short description associated as well as the results from group 09gr650 and ESA researchers.

Figure 2.5 illustrate the simulated link budget of a satellite based AIS receiver placed in a 1000 km altitude orbit. The ship antennas are placed in a height of 30 m and transmitting 12.5 W as defined in AIS standard for Class A transponder. Reflections from the water has been taken into consideration, which is why the signal strength has some sharp dives caused by signal cancellation. Note that the signal strength is calculated with the satellite receiving with an isotropic antenna.

Figure 2.6 illustrates the link budget expected for the BEXUS flight with the balloon in 35 km altitude. This simulation does not take reflections into consideration, but simulates the BEXUS experiment antenna with a half wavelength dipole on the receiver. According to the analysis, a -90 dBm received power corresponds to a SNR at 33 dB at the receiver. Figure 2.7 shows the probability that at least 1 AIS message will be received uncorrupted during a pass from each ship,

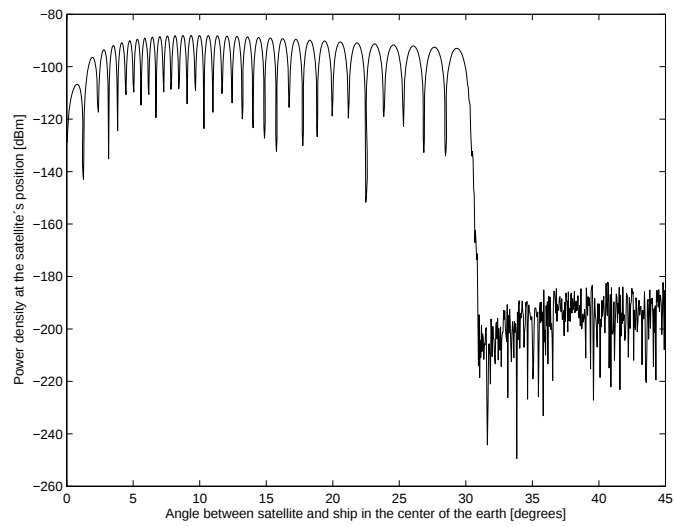


Figure 2.5: Link budget - AIS from AAUSAT3. 1 degree is approximately 111 km [07gr506 07].

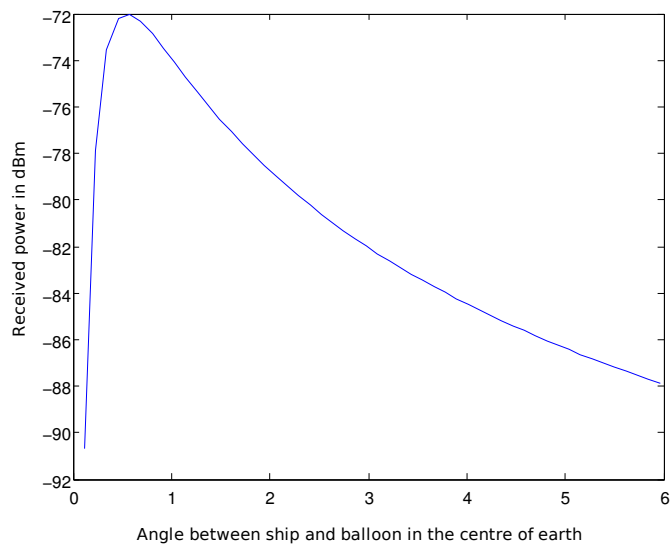


Figure 2.6: Link budget - AIS from balloon flight. 1 degree is approximately 111 km [09gr650 09].

when the ships is sending a message every 6 sec. The blue line plots the optimal pass for the satellite, which is when the satellite pass directly over the ship. The red line illustrates a shorter pass period. The simulation is done by ESA, with a footprint of 3000 nmi in diameter. Figure

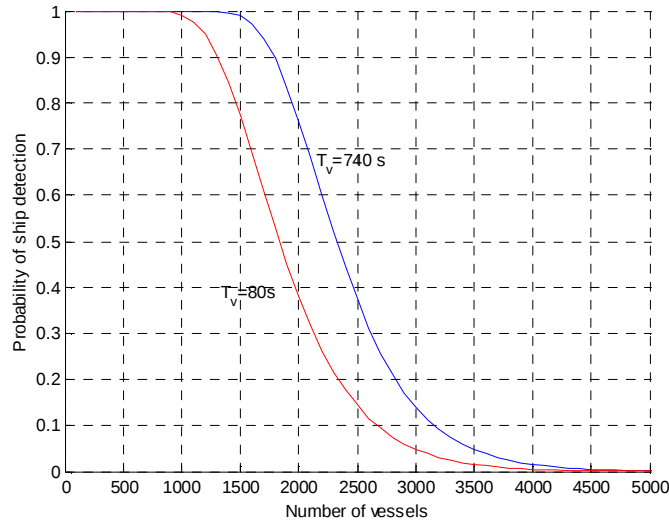


Figure 2.7: Probability of uncollided AIS packages [Cervera 08].

2.8 shows the expected antenna footprint on the BEXUS balloon flight. On the basis of this, it is expected that a certain number of ships will be in range in the Baltic sea. Undisclosed data from DaMSA showed 720 ships within this range of the BEXUS footprint at a given date in February 2009.

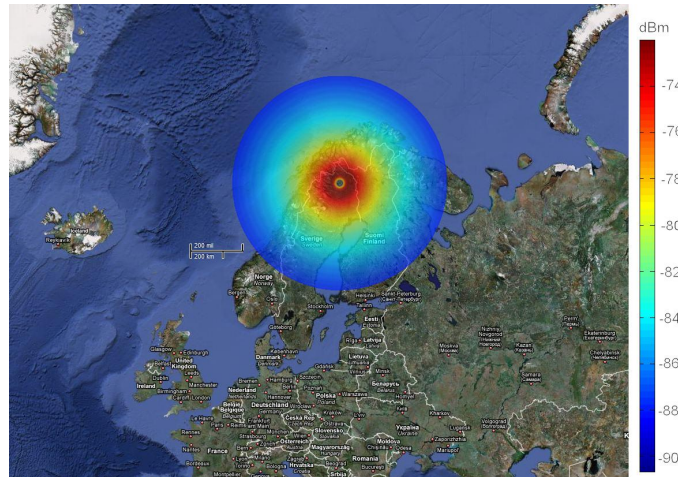


Figure 2.8: Antenna footprint from balloonflight [09gr650 09].

2.2 Wireless Communication

This section gives a short introduction to wireless communication systems, and introduces concepts such as orthogonal signals, signal constellation and bit error probability. Terminology and notation from [Fleury 07] is used.

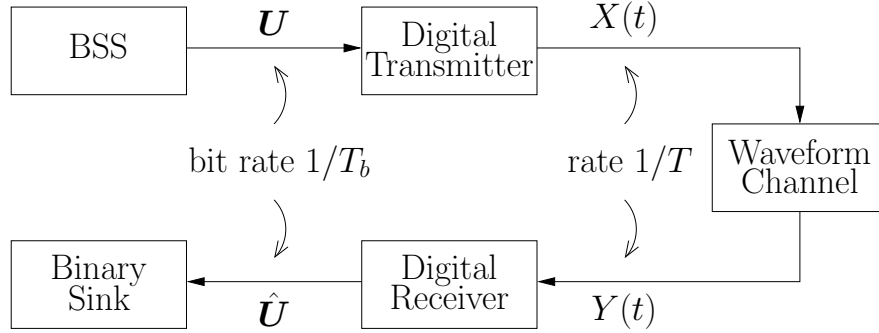


Figure 2.9: General wireless communication system [Fleury 07, p. 8].

Figure 2.9 shows a typical wireless digital communication system. The information source is considered to be a Binary Symmetric Source (BSS) that outputs a sequence of K independent and uniformly distributed bits, $\mathbf{U} = [U_1, \dots, U_K]$, $U_i \in \{0, 1\}$ at a bit rate of $1/T_b$. Each bit thus has a duration of T_b . The output of an efficient information encoder will look like the output of a BSS, so this introduces no loss in generality. The Digital Transmitter maps the K bits to a continuous function of time, $X(t)$, and outputs symbol waveforms at a rate of $1/T = 1/(KT_b)$, each with a duration of T . Note that $X(t) = 0$ for $t \notin [0, T[$. The waveforms are transmitted over a channel that may introduce noise, phase and frequency distortion, and interference from e.g. multiple transmission paths, before being received in the Digital Receiver. The receiver maps the received waveforms $Y(t)$ to the bit sequence $\hat{\mathbf{U}} = [\hat{U}_1, \dots, \hat{U}_K]$, $U_i \in \{0, 1\}$, before feeding the sequence to the Binary Sink destination. The receiver attempts to map the received symbols to $\hat{\mathbf{U}}$ such that $\hat{\mathbf{U}} = \mathbf{U}$. The probability of receiving erroneous bits, P_b , is called the Bit Error Probability (BEP).

$$P_b = \frac{1}{K} \sum_{k=1}^K \Pr(U_k \neq \hat{U}_k) \quad (2.1)$$

In AIS the BSS consists of two parts. An information source that outputs identification data, and a source encoder. The source encoder encodes the data from the information source in HDLC frames, using NRZI and bit stuffing, which is covered in Section 2.3. AIS uses a Gaussian Minimum Shift Keying (GMSK) modulation scheme in the Digital Transmitter. GMSK and a number of similar modulation schemes are described in Section 2.5 with consideration of BER in MSK and GMSK.

The Digital Transmitter maps the input bit sequence $\mathbf{U} = [U_1, \dots, U_K] \in \mathbb{U} = \{0, 1\}^K$ to an output waveform $x(t) \in \mathbb{S} = \{s_1(t), \dots, s_M(t)\}$ using a waveform lookup table (LUT). The LUT mapping $\mathbf{U} \mapsto x(t)$ is bijective, meaning that each waveform is the mapping of exactly one bit sequence.

The waveforms in $\mathbb{S} = \{s_1(t), \dots, s_M(t)\}$ spans a vector space. An orthonormal basis of vectors $\mathbb{S}_\psi = \{\psi_1(t), \psi_2(t), \dots, \psi_D(t)\}$ spanning the same space as spanned by $\mathbb{S}$ can be constructed by using the Gram-Schmidt orthogonalization process. A pair of vectors are orthogonal if their inner product is zero and a set of vectors form an orthonormal set if:

$$\langle \psi_k(t), \psi_l(t) \rangle = \int_{-\infty}^{\infty} \psi_k(t) \psi_l(t) dt = \begin{cases} 0 & \text{for } k \neq l \\ 1 & \text{for } k = l \end{cases} \quad (2.2)$$

Meaning that each of the vectors in $\mathbb{S}_\psi$ is of length 1 and orthogonal to all other vectors in the set. The waveforms in $\mathbb{S}$ can be constructed as linear combinations of the vectors in $\mathbb{S}_\psi$:

$$s_m(t) = \sum_{i=1}^D s_{m,i} \cdot \psi_i(t), \quad m = 1, 2, \dots, M \quad (2.3)$$

Each of the waveforms $s_1(t), \dots, s_M(t)$ can thus be formed by weighting the waveforms $\psi_1(t), \dots, \psi_D(t)$ with the vector $\mathbf{s}_m = [s_{m,1}, \dots, s_{m,D}]^T$. Figure 2.10 illustrates the mapping from the bit stream $\mathbf{U}$ to waveform. $X_1, \dots, X_D$ are used as the weights for $\psi_1, \dots, \psi_D$. The vector encoder maps the input bit sequence to the correct waveform and outputs the weights for the orthonormal basis functions to the waveform modulator. The modulator adds the signals and outputs the $X(t)$ waveform to the channel.

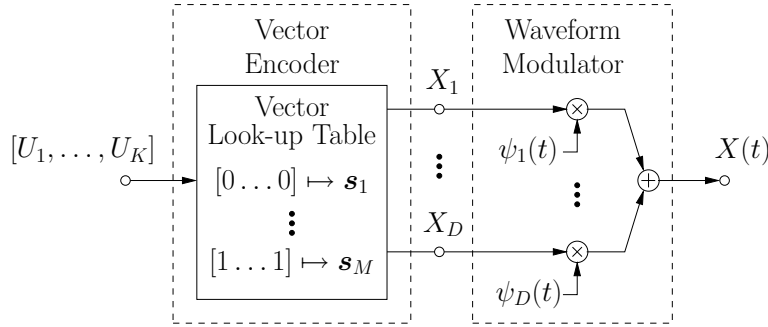


Figure 2.10: Decomposition of the digital transmitter [Fleury 07, p. 13].

2.3 Line Coding

The output bit stream from the AIS information source is NRZI (Non Return to Zero Inverted) encoded. NRZI is a differential line code, that encode data as a transition of the signalling level when a 0 bit is sent, and no transition when a 1 bit is sent. This means that a bit sequence can be NRZI encoded as two different sequences depending on the initial level of the line. NRZI encoding has the advantage, that decoded data is unaffected by the receiver swapping the 0 and 1 waveforms. A transition always indicates that a 0 was transmitted. The main disadvantage of NRZI is that a long series of 1 bits will have no transitions of the signalling level, making it difficult for the receiver to attain clock synchronization. AIS uses bit stuffing for sequences of more than five consecutive 1 bits, ensuring that the line will have at least one transition for every 6 bits transmitted. Figure 2.11 shows the original and NRZI encoded versions of 7 bits from an AIS training sequence. The AIS standard does not specify if training sequences should start with a 0 or 1 bit. Figure 2.11a is the original sequence starting with a 0 bit and Figure 2.11b and 2.11c are the NRZI encoded streams.

2.4 Channel Estimation

The wireless channel shown in Figure 2.9 can be estimated as a Additive White Gaussian Noise (AWGN) channel. Satellite communication is normally the textbook example on a AWGN channel, because multi path signals are minimal, due to the high gain directional antennas normally used in satellite communication. However, AIS vessels use omni directional antennas to improve horizontal communication, and this might to some extent introduce a signal patch where the ocean reflect the radio waves. The transmitter design and channel characterization impact the optimum receiver design. In this report, the focus is on a AWGN channel.

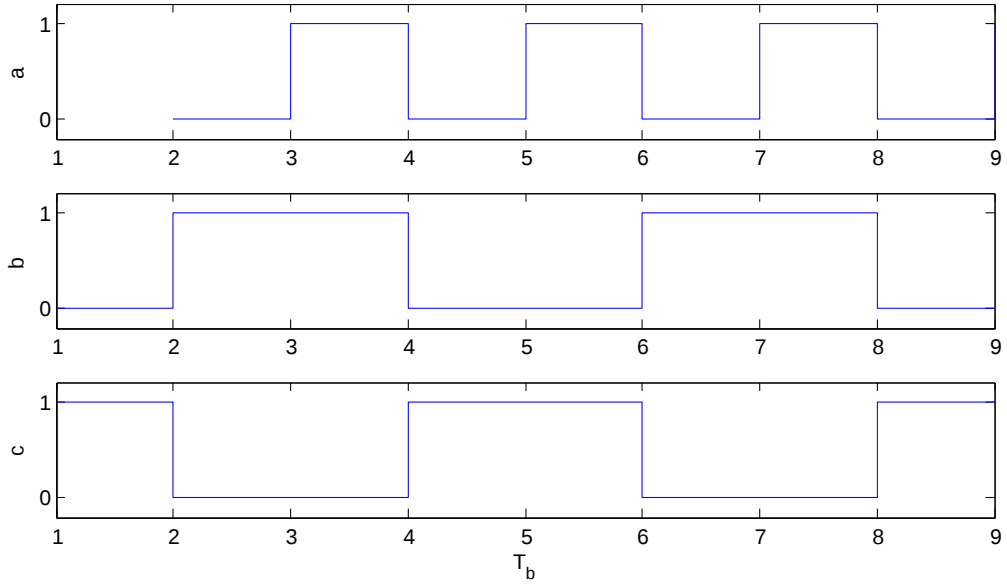


Figure 2.11: 7 training bits NRZI encoded.

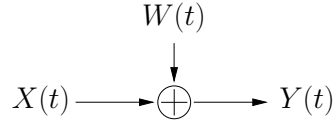


Figure 2.12: Additive White Gaussian Noise channel.

Figure 2.12 shows the AWGN channel. Here, the Gaussian distributed white noise are received together with the transmitted signal. The noise is normal distributed, have zero mean, and variance $N_0/2$. That is:

$$W \sim \mathcal{N}(\mu, \sigma^2) \quad (2.4)$$

$$\mu = E[W] = 0 \quad (2.5)$$

$$\sigma^2 = E[W - \mu^2] = \frac{N_0}{2} \quad (2.6)$$

Where W is the same dimension as X .

2.5 Digital Modulation

This section analyzes various Angle Modulation schemes, in order to understand the Gaussian Minimum Shift Keying (GMSK) modulation used in AIS. The section starts with a description of simple Binary Frequency-Shift Keying (BFSK), after which Minimum-Shift Keying (MSK), which is BFSK with minimum frequency separation, is examined. Finally MSK is extended to Gaussian Minimum Shift Keying (GMSK)

2.5.1 Binary Frequency-Shift Keying

In Binary Frequency-Shift Keying(BFSK), the information U_K is modulated as two frequencies, f_1 and f_2 . The first frequency f_1 in the duration of one bit, T_b , indicates a logic "0", and the

second frequency f_2 indicates a logic "1".

$$s(t) = \begin{cases} \sqrt{\frac{2E_b}{T_b}} \cos(2\pi f_1 t) & \text{for symbol 0} \\ \sqrt{\frac{2E_b}{T_b}} \cos(2\pi f_2 t) & \text{for symbol 1} \end{cases} \quad (2.7)$$

Where:

E_b is the transmitted energy per bit

T_b is the bit duration, measured in seconds

f_1 is the transmitted frequency when transmitting 0

f_2 is the transmitted frequency when transmitting 1

Figure 2.13 shows an example of BFSK, where the bit sequence 0b0111001 is transmitted with one bit pr. second.

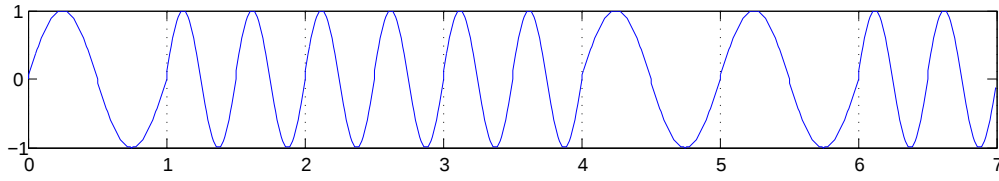


Figure 2.13: BFSK, 1 b/s, $f_1 = 1$ Hz, $f_2 = 2$ Hz.

Typically, the two frequencies f_1 and f_2 are chosen such as:

$$f_1 = \frac{n}{T_b} \quad (2.8)$$

$$f_2 = \frac{n+1}{T_b} \quad (2.9)$$

Where n is a integer. In Figure 2.13 $n = 1$, which is the minimum theoretical frequency choice. This is done to avoid discontinuities in the phase. Another possibility is to remember the phase at the end of the previous symbol, and compensate for that when transmitting the next symbol (Continuous Phase-FSK).

2.5.2 Minimum-Shift Keying

Minimum Shift Keying is a special case of BFSK, which ensures the minimum distance between the two frequencies, and thereby limiting the power spectrum - while still having orthonormal signal constellation and continuous phase. The frequency separation $f_\Delta = 1/2T$ used in MSK is the minimum separation such that $s_1(t)$ and $s_2(t)$ are still orthogonal. The proof is as follows: The two waveforms and the frequency separation are defined as:

$$s_1(t) = \sqrt{2P} \cdot \cos(2\pi f_1 t) \quad (2.10)$$

$$s_2(t) = \sqrt{2P} \cdot \cos(2\pi f_2 t) \quad (2.11)$$

$$f_\Delta = |f_1 - f_2| \quad (2.12)$$

Where P is the transmitted power, and f_1 and f_2 are the frequencies of the waveforms. A necessary requirement for $s_1(t)$ and $s_2(t)$ to be orthogonal, is that the scalar product of the waveforms is zero:

$$\langle s_1(t), s_2(t) \rangle = \int_{-\infty}^{\infty} s_1(t) s_2(t) dt = 0 \quad (2.13)$$

Applying Eulers formula:

$$\cos(2\pi f_0 t) = \frac{e^{j2\pi f_0 t} + e^{-j2\pi f_0 t}}{2} \quad (2.14)$$

And inserting in the expression of the scalar product:

$$\langle s_1(t), s_2(t) \rangle = \int_{-\infty}^{\infty} s_1(t) s_2(t) dt \quad (2.15)$$

$$= 2P \int_0^T \cos(2\pi f_1 t) \cdot \cos(2\pi f_2 t) dt \quad (2.16)$$

$$= 2P \int_0^T \frac{e^{j2\pi f_1 t} + e^{-j2\pi f_1 t}}{2} \cdot \frac{e^{j2\pi f_2 t} + e^{-j2\pi f_2 t}}{2} dt \quad (2.17)$$

$$= \frac{P}{2} \int_0^T e^{j2\pi(f_1+f_2)t} + e^{-j2\pi(f_1+f_2)t} + e^{j2\pi(f_1-f_2)t} + e^{-j2\pi(f_1-f_2)t} dt \quad (2.18)$$

In accordance with equation (2.12) $f_1 - f_2 = f_\Delta$:

$$\langle s_1(t), s_2(t) \rangle = P \left(\int_0^T \cos(2\pi(f_1 + f_2)t) dt + \int_0^T \cos(2\pi f_\Delta t) dt \right) \quad (2.19)$$

$$= \frac{P \cdot \sin(2\pi(f_1 + f_2)T)}{2\pi(f_1 + f_2)} + \frac{P \cdot \sin(2\pi f_\Delta T)}{2\pi f_\Delta} \quad (2.20)$$

As $\sin(x) \in [-1, 1]$ and $f_1 + f_2$ is usually $\gg 1$, the first term is approximately zero:

$$\langle s_1(t), s_2(t) \rangle = \frac{P \cdot \sin(2\pi f_\Delta T)}{2\pi f_\Delta} \quad (2.21)$$

Since $\sin(n\pi) = 0$ for $n \in \mathbb{Z}$, solving (2.21) for f_Δ yields:

$$\langle s_1(t), s_2(t) \rangle = \frac{P \cdot \sin(2\pi f_\Delta T)}{2\pi f_\Delta} = 0 \Rightarrow f_\Delta = \frac{1}{2T} \cdot k, \quad k \in \mathbb{Z} \setminus 0 \quad (2.22)$$

Hence, the minimum frequency separation is for $k = \pm 1$. Requiring f_Δ to be positive:

$$f_\Delta = \frac{1}{2T} \quad (2.23)$$

A simple example of the minimum frequency separation is shown in Figure 2.14a, where 0b0111001 is transmitted with 1 b/s, $f_1 = 1$ Hz and $f_2 = 1.5$ Hz.

Here the minimum frequency separation means that the phase must be remembered in the transmitter, and when the phase is π , the following symbol must be flipped vertically to avoid phase discontinuities, see Figure 2.14b. This gives the signal constellation shown in Figure 2.15.

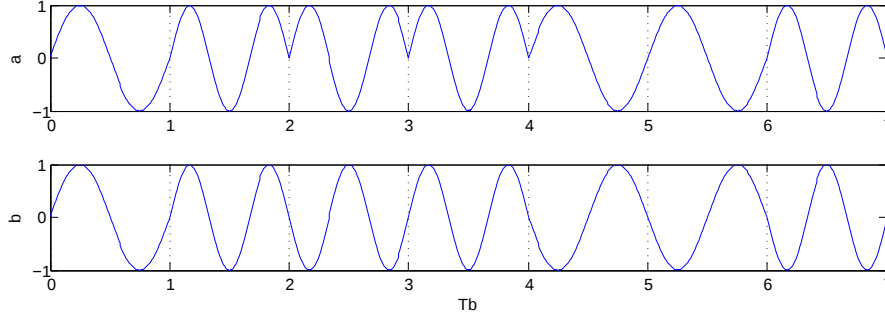


Figure 2.14: MSK waveform introduces memory to prevent phase discontinuities.

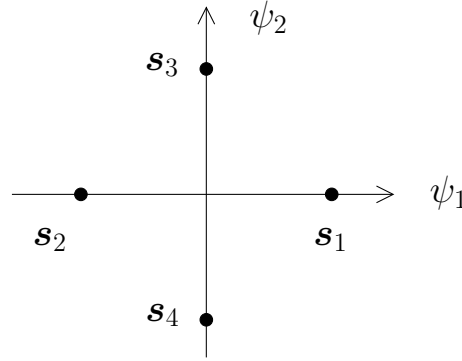


Figure 2.15: MSK Signal constellation [Fleury 07, p. 136].

The signal constellation corresponds to:

$$s_1 = +\sqrt{E_s} \cdot \psi_1(t) \quad (2.24)$$

$$s_2 = -\sqrt{E_s} \cdot \psi_1(t) \quad (2.25)$$

$$s_3 = +\sqrt{E_s} \cdot \psi_2(t) \quad (2.26)$$

$$s_4 = -\sqrt{E_s} \cdot \psi_2(t) \quad (2.27)$$

Where:

$$\psi_1(t) = \sqrt{\frac{2}{T}} \cos(2\pi f_1 t) \quad (2.28)$$

$$\psi_2(t) = \sqrt{\frac{2}{T}} \sin(2\pi f_2 t) \quad (2.29)$$

This looks like the signal constellation of Quadrature Phase Shift Keying (QPSK), however the modulation scheme is not the same. In QPSK, each symbol represents two bits, but in MSK only one bit is transmitted for each symbol. The previous accumulated phase thereby limits the next allowed transmitted symbol. This is shown in Figure 2.16. Hence, the bold pathway corresponds to 01100.

With the minimum frequency separation, f_Δ , and a desired center frequency, the modulation scheme may be represented by:

$$s(t) = \sqrt{\frac{2E_b}{T_b}} \cos(2\pi f_c t + \theta(t)) \quad (2.30)$$

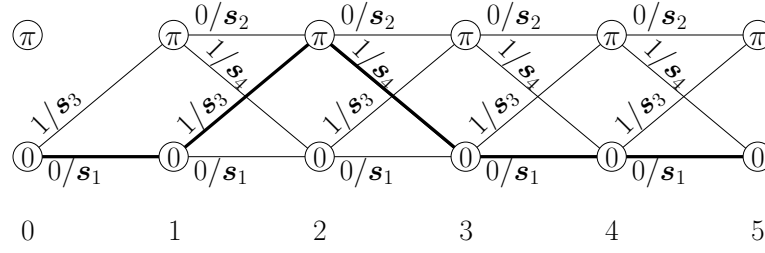


Figure 2.16: MSK trellis [Fleury 07, p. 141]. With f_1 as basis.

Where $\theta(t)$ is defined as follows:

$$\theta(t) = \frac{\pm h \cdot t}{2T_b} + \theta(0) \quad (2.31)$$

For $0 \leq t \leq T_b$.

Here, $\theta(0)$ is the phase memory in relation to the center frequency. When $h = 0.5$ as defined in the AIS standard, each bit transmitted will make a phase turn of 0.5π either positive or negative, depending on the transmitted bit. A trellis diagram showing the phase variation for the bit sequence 0b011001 is show in figure 2.17.

$$f_1 = f_c - \frac{1}{4T_b} \quad (2.32)$$

$$f_2 = f_c + \frac{1}{4T_b} \quad (2.33)$$

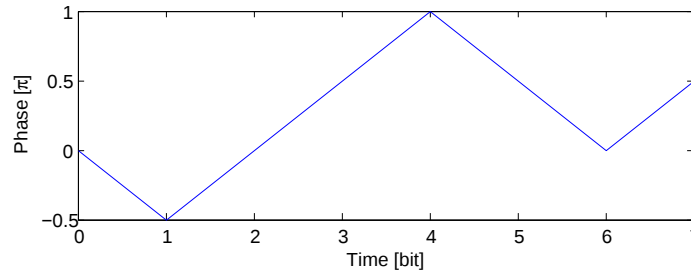


Figure 2.17: MSK Trellis diagram showing the bit sequence 0b011001. With f_c as basis.

2.5.3 MSK Demodulation and Bit Error Rate

The signal at the receiver is denoted $Y(t)$ as in Figure 2.9 on page 12.

Figure 2.18 shows an example of a non-coherent receiver using correlators, also known as product-integrators. Two correlators are required, one for f_1 and one for f_2 . The output of each correlator is denoted $Y_{j,1}$ and $Y_{j,2}$, and is therefore a measurement of the content of a transmitted f_1 or f_2 during the latest bit-time.

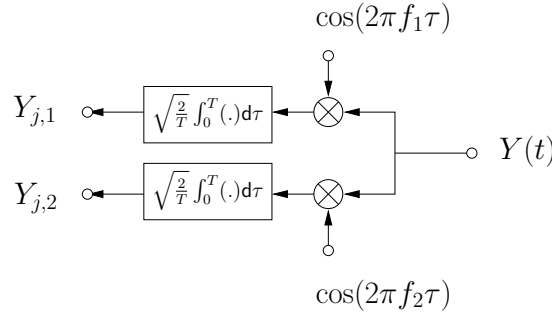


Figure 2.18: MSK demodulator [Fleury 07, p. 144].

The Decision Rule

By subtracting the two correlator output, y is created. A zero-crossing in y therefore indicates a bit transition, while a positive value indicates "1" and negative indicates "0". When y is sampled at T_b , the estimated bit sequence $\hat{U}_K$ is found. The decision rule is illustrated in Figure 2.19, where the non-coherent receiver only is able to estimate the transmitted frequency, and not the phase.

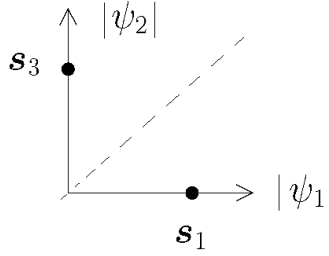


Figure 2.19: Non-coherent MSK decision rule.

This is the Most Likelihood decision rule, choosing the signal constellation closest to the received vector.

Bit Error Rate

Errors occurs only when the noise vector is large enough to move the transmitted vector to the other side of the decision rule boundary. That is, given s_3 is transmitted, error occurs when:

$$|y - s_1|^2 < |y - s_3|^2 \quad (2.34)$$

Where y is the received vector. Let $\tilde{d}$ denote the distance between s_1 and s_3 , and $N_0/2$ denote the noise variation. Then the error probability is given by [Vasudevan 07, p. 9]:

$$P_e = \frac{1}{2} \operatorname{erfc} \left(\sqrt{\frac{\tilde{d}^2}{8\sigma^2}} \right) \quad (2.35)$$

Where erfc is the complementary error function:

$$\operatorname{erfc}(x) = \frac{2}{\sqrt{\pi}} \int_{y=x}^{\infty} e^{-y^2} dy \quad (2.36)$$

And e^{-y^2} is shown in Figure 2.20.

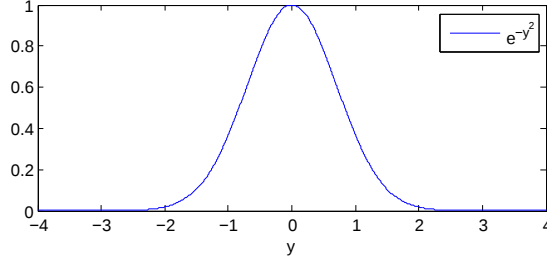


Figure 2.20: Normal distribution.

In this case, the distance $|\tilde{d}|^2$ is $\sqrt{E_s}^2 + \sqrt{E_s}^2$.

$$P_{e,nc} = \frac{1}{2} \operatorname{erfc} \left(\sqrt{\frac{E_s}{2N_0}} \right) \quad (2.37)$$

This BER is shown in Figure 2.26.

Another possibility is to demodulate coherently. That is, recover the phase of the signal in the receiver, and hence being able to detect whether s_1 or $-s_1$ was transmitted. Due to the phase memory in the transmitter, it is possible to use most likelihood sequence estimation, and achieve a BER 3 dB better [Haykin 01, p. 394]:

$$P_{e,c} = \frac{1}{2} \operatorname{erfc} \left(\sqrt{\frac{E_s}{N_0}} \right) \quad (2.38)$$

2.5.4 Gaussian Minimum-Shift Keying

The GMSK modulation technique is an enhancement of MSK, where a filter is convoluted with the NRZ encoded bits before generating the modulated signal. The Gaussian filter impulse response is defined in the time domain as follows:

$$h(t) = \sqrt{\frac{2\pi}{\ln(2)}} W \exp \left(-\frac{2\pi^2}{\ln(2)} W^2 t^2 \right) \quad (2.39)$$

$$W = \frac{BT}{T_b} \quad (2.40)$$

where BT is the time-bandwidth product which determines the softness of the filter. From section 2.1 it is known that $BT \approx 0.5$. BT is mentioned as WT_b in some literature.

Figure 2.21 shows how one square bit is shaped by the Gaussian filter. $BT = 0.3$ is shown to compare with the GSM standard.

In Figure 2.22 a Gaussian filtered bit sequence is compared to the unfiltered bit sequence. Figure 2.23 shows the phase that is formed by the same bit sequence.

Figure 2.23 shows the MSK phase that is formed by the same bit sequence, with and without Gaussian filter.

Figure 2.24 shows how the Gaussian filter improves the power spectral density.

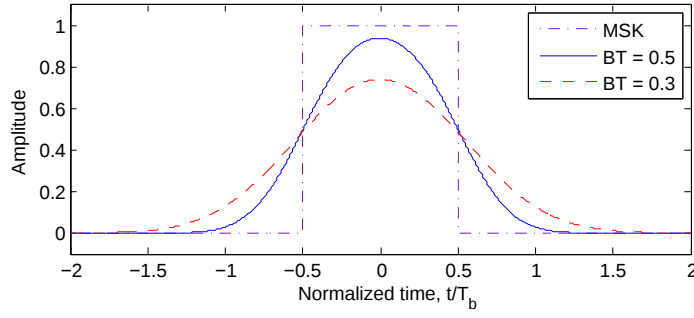


Figure 2.21: Gaussian filter applied to one bit.

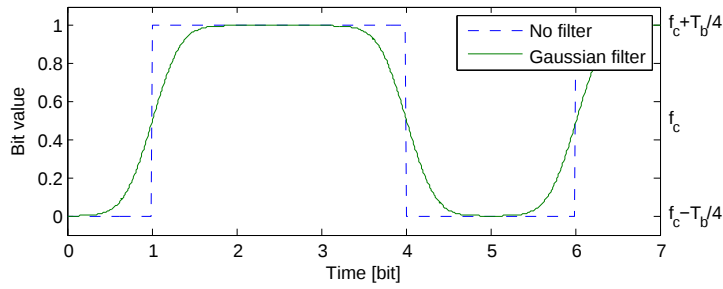


Figure 2.22: Bit sequence 0111001 with and without Gaussian applied filter.

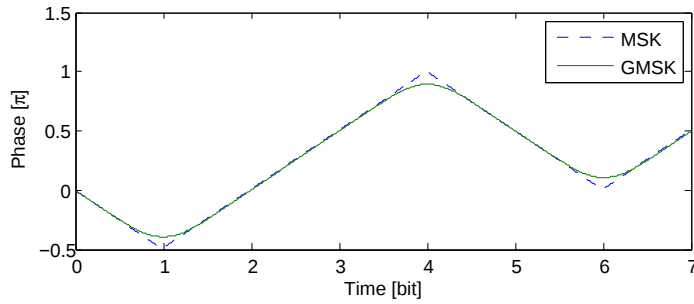


Figure 2.23: Trellis diagram showing the bit sequence 0111001 with and without Gaussian filter applied.

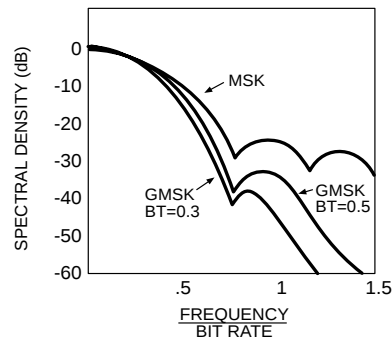


Figure 2.24: Power Spectral Density for MSK and GMSK with BT 0.3 and 0.5 [MX Com 95, p. 4].

2.5.5 Demodulating GMSK

Figure 2.25 shows the degradation in BER as a function of the BT [Haykin 01, p. 399]. This result is not found analytically, but empirical by means of simulations.

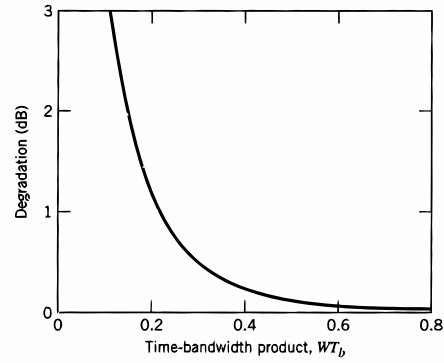


Figure 2.25: BER of GMSK compared to MSK [Haykin 01, p. 399].

In AIS the BT is 0.5, and therefore the degradation in BER is assumed to be about 0.1 dB. Figure 2.26 shows the Gaussian impact on the BER from section 2.5.3.

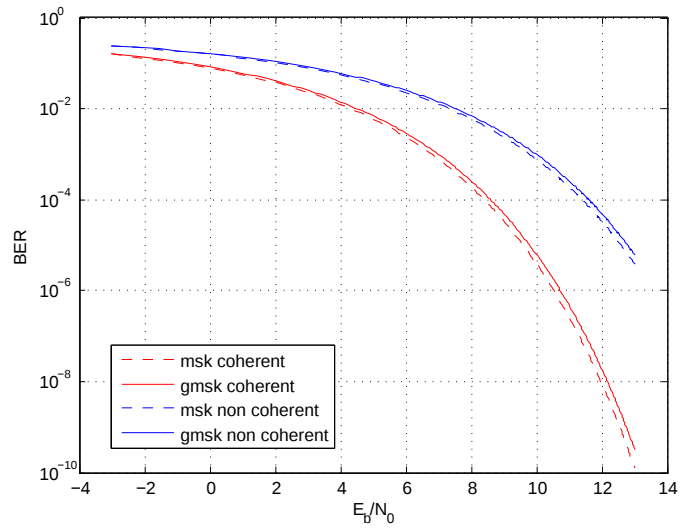


Figure 2.26: BER of MSK and GMSK.

Problem Description

The introduction and analysis yields the following problem description:

"How do you construct a software defined radio that can be used on AAUSAT3 to test the possibility to receive AIS signals in space?"

The goal of this project can be divided into the following two subgoals:

1. To develop and test prototype hardware and software for the software defined AIS receiver for AAUSAT3.
2. To sample raw data on the BEXUS flight, in order to investigate AIS from high altitude.

3.1 Requirement Specification

The receiver must:

1. Be able to sample raw data which contains both AIS channels with a sufficient sample rate to satisfy the Nyquist sample theorem.
2. Demodulate GMSK in software.
3. Be able to update all parts of the demodulating and decoding software remotely.
4. Be able to compensate for frequency drift, including Doppler shift when the satellite is moving with 7.5 km/s .
5. Be able to store raw sampled data for 3600 sec (1 hr).
6. Be functional from $-40 - +85^\circ \text{ C}$.
7. Have a weight of maximally 150 g.
8. Be operating for at least 6 months.
9. Comply with the standards defined by the AAUSAT3 system engineering group.
 - a) Be able to run at 3.3 V and/or 5 V only.
 - b) Use no more power than 1 W.
 - c) Comply with the PCB layout for AAUSAT3 including definition of stack connector and board outline.
 - d) Have at least one temperature sensor placed on a central part of the PCB.

- e) Be able to be controlled completely over CSP.
- f) Send telemetry to the LOG subsystem.

3.2 Test Specification

1. *Be able to sample raw data which contains both AIS channels with a sufficient sample rate to satisfy the Nyquist sample theorem.*
The front-end and ADC setup is investigated, and it is verified whether the sample rate is sufficient.
2. *Demodulate GMSK in software.*
A antenna is connected to the system, and the software demodulator is tested.
3. *Be able to update all parts of the demodulating and decoding software remotely.*
A changed version of the demodulating and decoding software is written and uploaded to the subsystem, after which it is tested whether the changes takes effect.
4. *Be able to compensate for frequency drift, including Doppler shift when the satellite is moving with 7.5 km/s .*
An artificial signal with a worst case frequency drift in both directions is generated using a GMSK modulator, and it is observed whether the signal is demodulated correctly.
5. *Be able to store raw sampled data for 3600 sec (1 hr).*
The receiver is admitted to run until the memory limit is reached, and the amount of raw data is observed.
6. *Be functional from $-40 - +85^\circ \text{ C}$.*
The receiver is tested for worst case temperatures, and in vacuum chamber.
7. *Have a weight of maximally 150 g.*
The weight of the experiment setup is measured.
8. *Be operating for at least 6 months.*
9. *Comply with the standards defined by the AAUSAT3 system engineering group.*
 - a) *Be able to run at 3.3 V and/or 5 V only.*
The required input voltages to the system is observed.
 - b) *Use no more power than 1 W.*
The power consumption is measured with the system running fully operational.
 - c) *Comply with the PCB layout for AAUSAT3 including definition of stack connector and board outline.*
The size of the PCB is measured and the electric interface is tested.
 - d) *Have at least one temperature sensor placed on a central part of the PCB.*
Tested by observing and measuring using the on board sensor.
 - e) *Be able to be controlled completely over CSP.*
Commands is sent to the subsystem, and it is checked whether the correct messages is returned.
 - f) *Send telemetry to the LOG subsystem.*
It is observed whether the relevant telemetry data is logged.

Design

The purpose of this chapter is to give an overview of the simulation setup used to design and test the GMSK demodulator. MATLAB code will be used to test the quality of these methods before implementing the decoder on a DSP. To ensure relevant testing data, a GMSK modulator which creates the same output as expected from the front-end is programmed. Note that only some parts of the code is shown in this report. The complete source code can be found on the attached CD. Figure 4.1 shows the included parts in the signal path. A front-end, ADF7020-

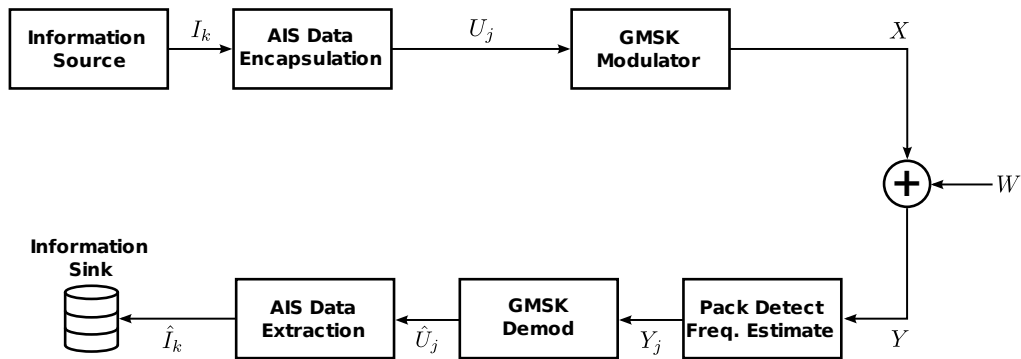


Figure 4.1: Simulation setup overview.

1[Analog Devices 05], has been selected as downconverter. This is due to the fact that it has been used previously in the AAUSAT3 project, and because it fits the requirements for the AIS receiver. The ADF7020-1 is a complete transceiver, but can be used as a downconverter only. When correctly configured, the receiver downconverts from 162 MHz to an intermediate frequency of 200 kHz, such that the two AIS channels will be centered around 175 kHz and 225 kHz. Both the in-phase and the quadrature output is available from the downconverter. The front end contains an image rejection filter rejecting frequencies outside the band from 150 kHz to 250 kHz. The configuration is described in section 6.5.

The simulation setup must therefore have the properties as seen in table 4.1 to match the specifications of respectively the AIS standard and the properties of the chosen front-end.

Parameter Name	Setting	Unit
AIS channel 1	175	kHz
AIS channel 2	225	kHz
Pandpass filter	150 – 250	kHz
Modulation scheme	GMSK	
Carrier frequency drift	± 4	kHz
Modulation index	~ 0.5	
BT product	~ 0.5	max
Bit rate	9.6	kbps
Training sequence	24	bits
Start and stop byte	0x7E	
Total packet length	224	

Table 4.1: Requirements to the modulator.

4.1 Information Source

The information source uniformly distributed random bits. A CRC16 calculator is not implemented, so the bits corresponding the FCS are also random bits. In AIS this is the real data, such as MMSI and position, here denoted I_k .

4.2 AIS Data Encapsulation

The output from the information source must be encapsulated prior to modulation, in order to resemble an AIS packet. This section describes the bit stuffing, HDLC framing and NRZI encoding.

4.2.1 Bit Stuffing

The encapsulated data must be bitstuffed due to the definition of HDLC and the AIS standard. In Code Example 4.1 MATLAB is first used to find sequences of 5 or more 1's. After this, it is checked whether these results are valid. This is because if for example seven 1's will appear in a row, MATLAB will return 3 values, although only the first one should be used. When the superfluous placements has been found and removed, an extra zero is added each time five 1's appear in a row in the data.

```

1 bitstufplacement = findstr(U,[1 1 1 1 1]);
2 % Remove results that less than 5 bits apart
3 ii = 2;
4 while ii <= length(bitstufplacement)
5     if bitstufplacement(ii) < bitstufplacement(ii-1) + 5, \
6         bitstufplacement(ii) = [];
7     else, ii = ii + 1; end
8 end
9 % Insert the actual bit stuffing
10 for ii = 1 : length(bitstufplacement)
11     U = [U(1:bitstufplacement(ii)+4) 0 \
12         U(bitstufplacement(ii)+5:length(U))];
13 end

```

Matlab Code 4.1: Bit stuffing.

4.2.2 HDLC Frame Format

After bit stuffing, the frame must be encoded due to the AIS standard, i.e. the start flag (0x7E) is added to the start and the end of the content. Code Example 4.2 shows this procedure. Also, the 24 alternating bits are added in this example, such that the packet contains all the information that shall be modulated after NRZI encoding. Note line 1 where a random bit is chosen to decide whether the training sequence shall start with either a 0 or a 1. This is due to the uncertainty of the initial NRZI state.

```

1 jj = rand(1) < 0.5;
2 for ii = 1:24
3     training(ii) = mod(ii+jj, 2);
4 end
5
6 STX = [0 1 1 1 1 1 1 0];
7 U = [training STX U STX];
8 U = NRZI(U, rand(1) < 0.5);

```

Matlab Code 4.2: HDLC encoding.

4.2.3 NRZI Encoding

The NRZI (Non Return to Zero Inverted) is the next part of the simulator. It is implemented in Code Example 4.3.

```

1 U = [round(rand(1)) U];
2 for ii = 2:length(U)
3     if U(ii) == 1, U(ii) = invert(U(ii-1));
4     elseif U(ii) == 0, U(ii) = U(ii-1); end
5 end
6 x = x(2:length(x));

```

Matlab Code 4.3: NRZI encoding.

In this example, a random bit is inserted at the start of the bit stream. This simulates the two possibilities for the start of the transmission that can be expected. An example of the two ways a bit stream can be encoded, is shown in example 4.2. The initial state of the NRZI encoder is listed in the second column.

Original Stream:	0	1	1	1	0	0	1
NRZI:	1	0	0	0	0	1	0
NRZI:	0	1	1	1	1	0	1

Table 4.2: The two ways a bit stream can be NRZI encoded.

4.3 GMSK Modulator

In this section, a GMSK modulator that accepts a bit streams as input will be created in MATLAB. The output of the modulator must be a 1 Msps signal. This choice is based on the expected output of the ADF7020-1 front-end chip.

4.3.1 Gaussian Filtering

The NRZI encoded data stream must be applied a Gaussian filter to fulfil the definition of GMSK. Figure 2.21 shows a square pulse convoluted with a Gaussian filter for varying time-bandwidth products. The variable F_s defines samples per second while W is the time-bandwidth product. XU_j is the bitstream, but with T_b/F_s samples per bit.

```
1 n = linspace(-150, 150, 301);
2 gauss = sqrt(2*pi/log(2))*W*exp(-2*pi^2/log(2)*W^2*(n/Fs).^2)/Fs;
3 XUj = conv(XUj, gauss);
```

Matlab Code 4.4: Gaussian filtering.

Code Example 4.4 shows the creation of the Gaussian pulse by formula 2.39 and the convolution of this and the array XU_j , which contains the data stream sampled at F_s . Note that the filter is normalized by the variable F_s (samples per second).

4.3.2 Phase Establishment

The filtered bit stream must be used to generate the phase used for modulating the carrier frequency, as described in section 2.5.

```
1 XUj = (XUj-0.5)*2;
2 P(1) = 0;
3 for ii=2:length(XUj)
4     P(ii) = P(ii-1) + XUj(ii)*pi*h/spb;
5 end
```

Matlab Code 4.5: Phase Generation.

Code Example 4.5 shows the construction of the phase in discrete time. The variable h is the deviation which in MSK is equal to $1/2$. The phase change is normalized by spb (samples per bit), so that a phase turn of $\pi \cdot h = \pi/2$ will occur at each bit, either positive or negative correspondent to the transmitted bit.

4.3.3 Frequency Modulation

The array XU_j that now contains the gaussian filtered phase of the desired signal, will be used to create the frequency deviation during the frequency modulation.

Code Example 4.6 shows the frequency modulation. First a random carrier frequency ± 4 kHz, according to section 2.1.3 is chosen to simulate a possible Doppler shift. A delay is then added to randomize the start of the bits, so a realistic test of the bit synchronizer is possible. The modulated signal gm is a combined version of I and Q, which is produced by formula 2.30.

4.3.4 Signal Addition

In the above sections, a GMSK signal has been generated. To fulfil the requirements of the AIS standard, two modulated signals must be added to simulate the use of the two channels.

```

1 fc = fc - 4000 + 8000*rand(1);
2 for ii = startdelay+1 : len
3     X(ii) = cos(2*pi*ii*fc/Fs + P(ii));
4 end

```

Matlab Code 4.6: Frequency modulation.

The signals is transferred at 161.975 MHz and 162.025 MHz, which corresponds to 175 kHz and 225 kHz when downconverted through the selected front-end. Therefore, the code from the above sections is executed twice to generate the two signals before they are combined as shown in Code Example 4.7.

```

1 X = X1 + X2;

```

Matlab Code 4.7: Signal addition.

4.4 AWGN Channel

The modulated signal, X , must have noise added to simulate the wireless channel. Therefore, the code in Code Example 4.8 is generated to add Gaussian noise to the signal using the MATLAB function `AWGN()`. The input signal-to-noise is pr. sample (SNR_b), which is the desired SNR divided by the number of samples pr. bit. After the signal is bandpass filtered to make the same bandwidth as the output from the ADF7020-1 radio front-end. The filter used is a 5th order Butterworth filter which is the same as used in the actual front-end. The variable SNR is the desired signal-to-noise ratio while f_{ci} is the center frequency of the downconverted signal and BW is the bandwidth of the filter. The bandwidth is planned to be 100 kHz, but is adjustable in the frontend. SNR is the signal-to-noise ratio per sample, and SNR_b .

```

1 SNRb = SNR - 10*log10(Fs*Tb);
2 Y = awgn(X, SNRb);

```

Matlab Code 4.8: AWGN Channel

```

1 low = 2*(fci-BW/2)/Fs;
2 high = 2*(fci+BW/2)/Fs;
3 [b,a] = butter(5,[low high]);
4 Y = filter(b, a, Y);

```

Matlab Code 4.9: IF filter

4.5 Packet Detection

The first part of the AIS decoding is the packet detection. It is important that the packet detection is fast, so that it is possible to execute the detector on the DSP in real-time. In the following section, a detector using FFTs is described.

When the detector is running, it continuously evaluates an FFT of a few bits durations, equalling a few hundred samples. The FFTs are calculated with a step so an AIS packet is guaranteed to be included in at least one FFT. The approximate area that is used for Fourier transforms is seen in Figure 4.2. Computational power will be saved by not calculating the FFT of the entire dataset. The areas of the FFTs just have to be large enough to reach sufficient frequency resolution. Zero padding to improve the resolution will have no purpose in this case, because calculating the FFT over more data will need almost the same computation power. The

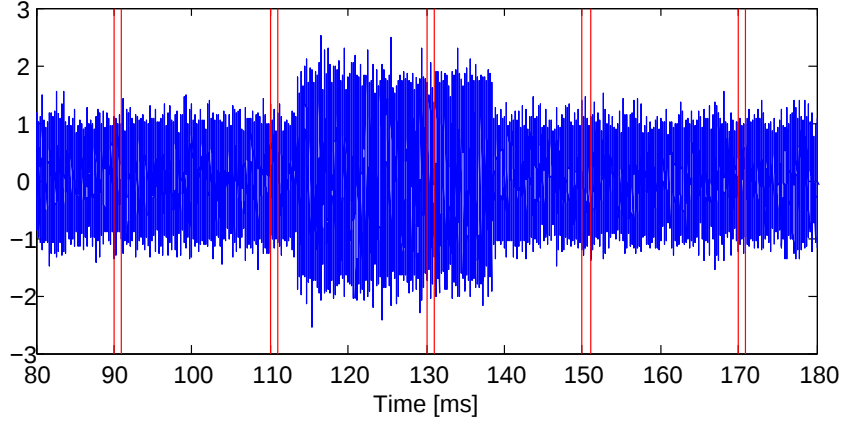


Figure 4.2: Continuously fourier transforms in selected areas.

frequency analysis is filtered by convolving with a Blackman window after which the value at 200 kHz and the maximum value between the two spectrums of the AIS channels, approximately 170 kHz – 180 kHz and 220 kHz – 230 kHz, is found. The difference of the 200 kHz value and the maximum value of the two AIS channels is used to determine whether the data that has been basis for the Fourier transform contains any AIS packets. Note that because the FFT used is very rough, no accurate frequency estimation is performed by this method. Figure 4.3 shows the frequency spectrum when a packet is detected. If the maximum value of the filtered FFT is

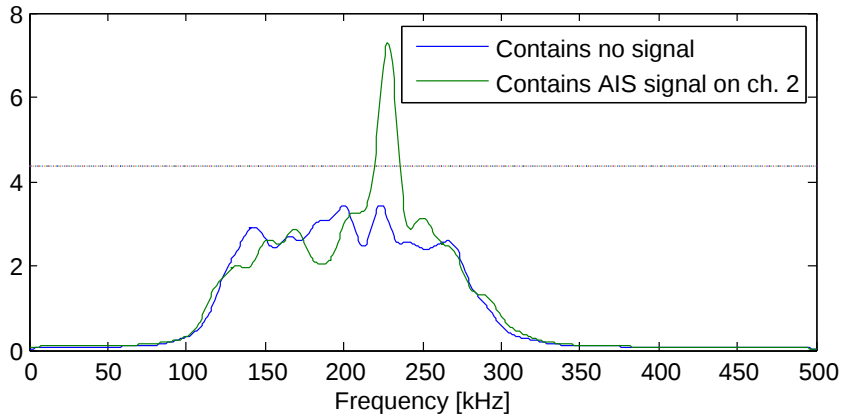


Figure 4.3: Rough FFT.

above the determined threshold, the data around the discovered packet is sent to a new function which seeks the data more accurate for finding the precise packet start and stop. The frequency estimation also uses FFTs, but with a smaller step size. The start and stop of the signal is now marked to be in the first couple of bits in the training sequence and a few bit times after the stop bit.

4.5.1 Verification of Packet Detection

The packet detector is verified by creating 100 seconds of simulated output from the RF frontend, each containing 1 AIS packet on each channel. Packets are subject to frequency drift and time randomization. The signal is generated with an SNR of 20 dB and 1 Msps. Figure 4.4 shows the difference from the actual packet start to the detected packet start. 1 bit equals approximately $104 \mu s$ which means that between 1 and 4 bits of the training sequence is missing. 187 of 200 packets was detected. This number might be possible to increase by fine tuning variables.

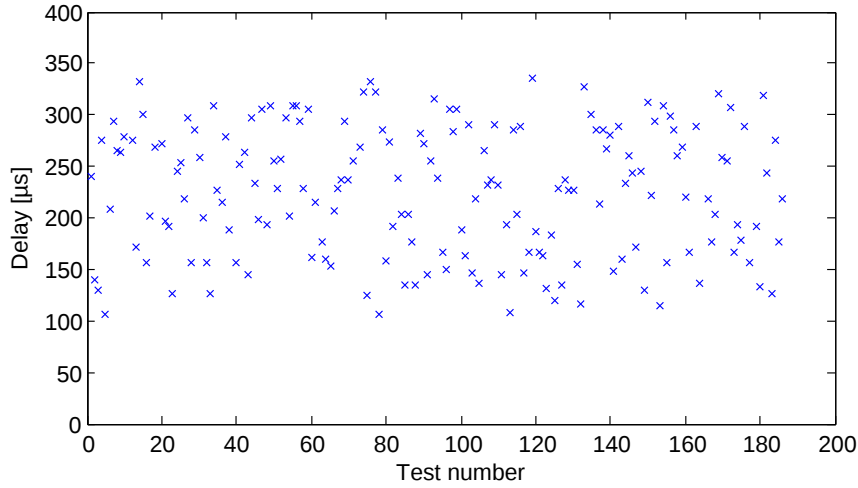


Figure 4.4: Test of packet detection.

4.6 Frequency Estimation

The frequency estimation exploits the 24-bit training sequence of the AIS packet. When the training sequence is found by the packet detector, an FFT over a multiple of 4 bits (maximum 16 to avoid that the FFT is made partly over the start sequence) is calculated. By calculating the FFT over a multiple of four bits, the training sequence will contain an equal number of 1 and 0 bits, so that the center frequency will appear as the maximum value when applying a suitable filter on the FFT. Note that zero padding is possible if considered necessary for making an accurate frequency estimation. Figure 4.5 shows an FFT of 4 training sequence bits, zero padded to a total length of 12. An improvement of the frequency estimation could be to correlate the signal

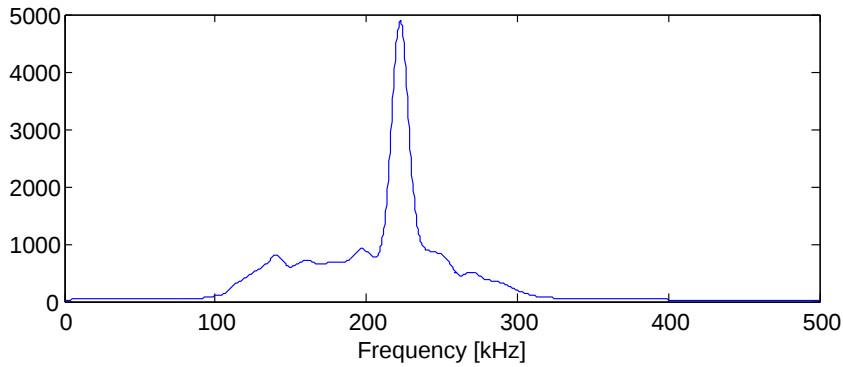


Figure 4.5: FFT of training sequence.

by signals of different frequency when the approximate frequency has been estimated by above method, and select the frequency with the highest correlation coefficient. This has not yet been implemented.

4.6.1 Verification of Frequency Estimation

The packet detector is verified by creating 100 seconds of simulated output from the RF front-end, each containing 1 AIS packet on each channel. Packets are subject to frequency drift and time randomization. The signal is generated with a SNR of 20 dB and 1 Msps per second. The packet detector is used to find the packets, which is why only 185 tests are performed. Figure 4.6 shows the difference from the actual modulation frequency to the frequency detected. The figure shows that most frequencies are found with a variation less than 200 Hz. This precision is possible to increase by using correlators when the approximate frequency is found.

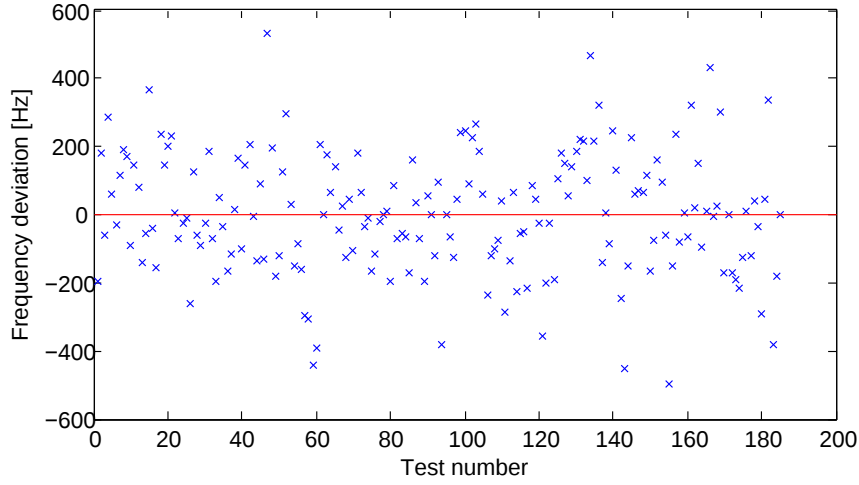


Figure 4.6: Test of frequency estimation.

4.7 GMSK Demodulator

The demodulator is designed as described in Section 2.5.3. First, two waveforms are generated, used to search for f_1 and f_2 , which is placed around the center frequency found by the frequency estimator.

$$f_1 = f_c - \frac{f\Delta}{2} \quad (4.1)$$

$$f_2 = f_c + \frac{f\Delta}{2} \quad (4.2)$$

The two waveforms are generated over one bit time, T_b , with the sample rate F_s :

$$t = n \frac{1}{F_s}, \quad n \in \{0, 1, 2, \dots, spb - 1, spb\} \quad (4.3)$$

$$\text{waveform1} = \cos(2\pi t f_1) \quad (4.4)$$

$$\text{waveform2} = \cos(2\pi t f_2) \quad (4.5)$$

Y_j is the received signal. Let $Y_{j,1}$ denote the cross correlation between waveform1 and Y_j . That is:

$$Y_{j,1}(m) = \sum_{n=0}^{N-m-1} \text{waveform1}_{n+m} Y_n \quad (4.6)$$

And similar for $Y_{j,2}$. A simple example of a correlator is shown in Figure 4.7. Here, the correlation between one period of a sine is correlated with two periods of a cosine. Figure 4.8 shows the actual

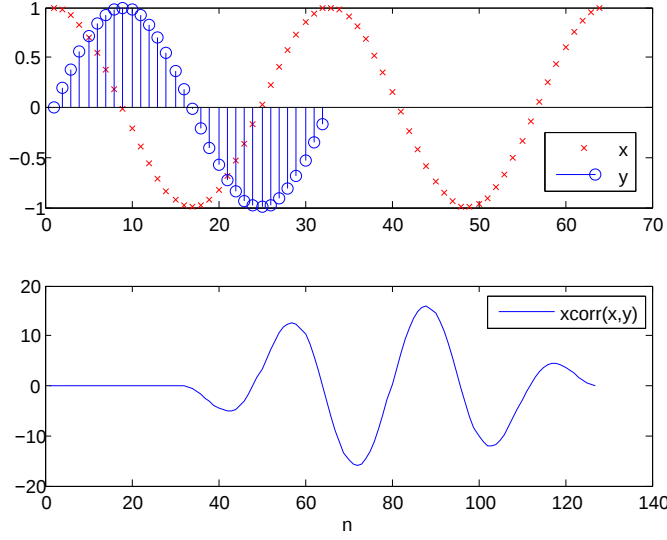


Figure 4.7: Example of a correlator.

demodulator. The correlator, or product integrator, is used and a soft output of the bit stream $U_{j,s}$ is generated. The bit synchronization is illustrated by a switch sampling at every T_b . Figure

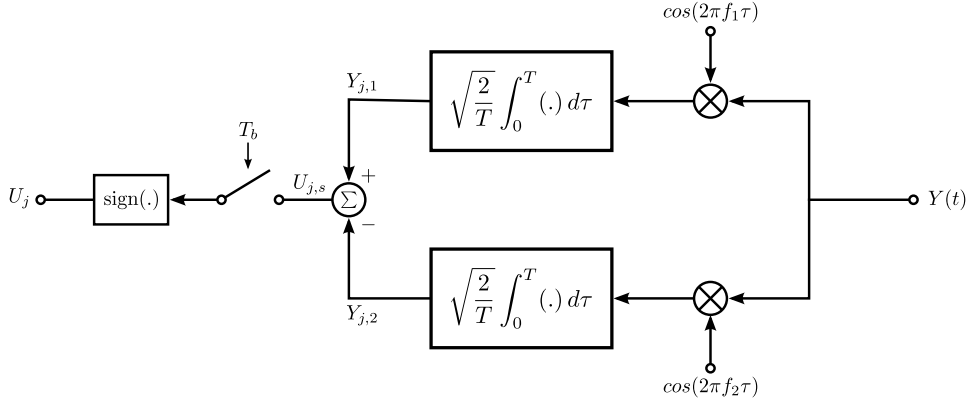


Figure 4.8: The demodulator.

4.9 shows the correlator output $Y_{j,1}$ and $Y_{j,2}$, of a training sequence and start flag at 25 dB SNR.

4.7.1 Bit Synchronization

The Bit synchronization is used to recover the clock of the received signal. The 24 bit NRZI encoded training sequence in the AIS standard ensures 11 bit transactions and thereby 11 zero crossings in the correlator demodulator output. The packet detector determines the package start and use this to ensure that the correlator demodulator starts just at the beginning of the training sequence. Therefore, the first 10 zero crossings are used to estimate the bit placement. The packet detector might miss the first zero crossing due to inaccuracy in the detection. First, the zero crosses

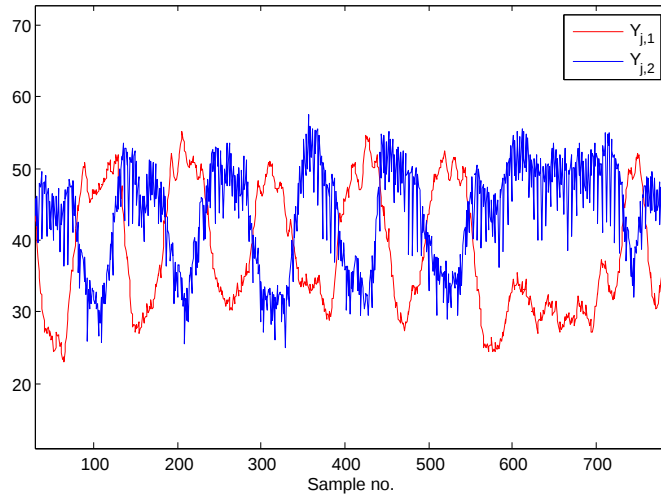


Figure 4.9: The waveform demodulator output $Y_{j,1}$ and $Y_{j,2}$.

are detected. This is done in Code Example 4.11. The first 10 zero crossings are then used to

```

1 j=1; % Zero-cross detect number
2 for ii=2:length(Ujs)
3     if (sign(Ujs(ii))~= sign(Ujs(ii-1))) % If sign has changed
4         crossdetected(j) = ii-1-0.5; % save zero cross placement
5         j = j+1; %
6     end
7 end

```

Matlab Code 4.10: Zero cross detection

determine the bit placement best fitted for the mean of this training sequence, knowing that the time $2T_b$ is expected between each zero crossing. This is shown in Code Example 4.11.

```

1 crossdetected = crossdetected(1:10);
2 error = 0;
3 for ii = 1:length(crossdetected)
4     currenterror = crossdetected(ii) - (ii-1)*(2*spb/decimation);
5     error = error + currenterror;
6 end
7 bitphase = error/length(crossdetected);

```

Matlab Code 4.11: Bit phase determination

Using the accumulated error per bit, the bit phase, the final bit placement in the pack is determined. This bit synchronization "lock" is used throughout the package, by adding the expected bit time T_b between each reading of the bit value. In Code Example 4.12 *nobits* is the number of bits expected throughout the soft output bit stream Ujs . Further work is needed to evaluate if the bit synchronization algorithm needs to be adaptive, to compensate for clock and frequency drifts, when receiving longer AIS packets.

```

1 nobits = floor((length(Ujs)-bitphase)/(spb/decimation));
2 for ii = 1:nobits
3     bitplacement(ii) = round( \
4         bitphase \
5         + (ii-1)*spb/decimation \
6         + spb/decimation/2);
7 end

```

Matlab Code 4.12: Final bit placement

4.7.2 Decision rule

The hard decision rule is as follows:

```

1 Uj_hat = (sign(Ujs(bitplacement))+1)/2;

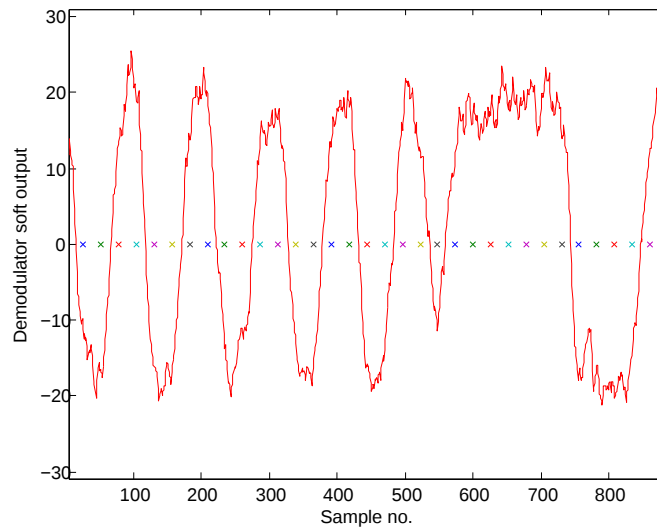
```

Matlab Code 4.13: Decision Rule

At each bit placement the sign of the soft output (U_{js}) from the demodulator is read. The sign is then converted from the range $[-1 : 1]$ to $[0 : 1]$. Figure 4.10 shows the implemented decision rule, where the blue dots indicates a bit placement.

4.7.3 Verification of GMSK Demodulator

Figure 4.10 shows the soft output, U_{js} , of the demodulator, simulated at 25 dB SNR. The corresponding U_j is 0b00110011... Figure 4.11 shows a simulation where the demodulator part

**Figure 4.10:** Soft output from the demodulator and bit placements.

itself is tested, estimating perfect bit synchronization.

The test is performed by modulating 20000 random bits for each SNR from 1 to 25 dB. AWGN is added to get the desired SNR, and the demodulator estimates the transmitted bits. The transmitted data is then compared to the demodulated and the number of errors are divided by

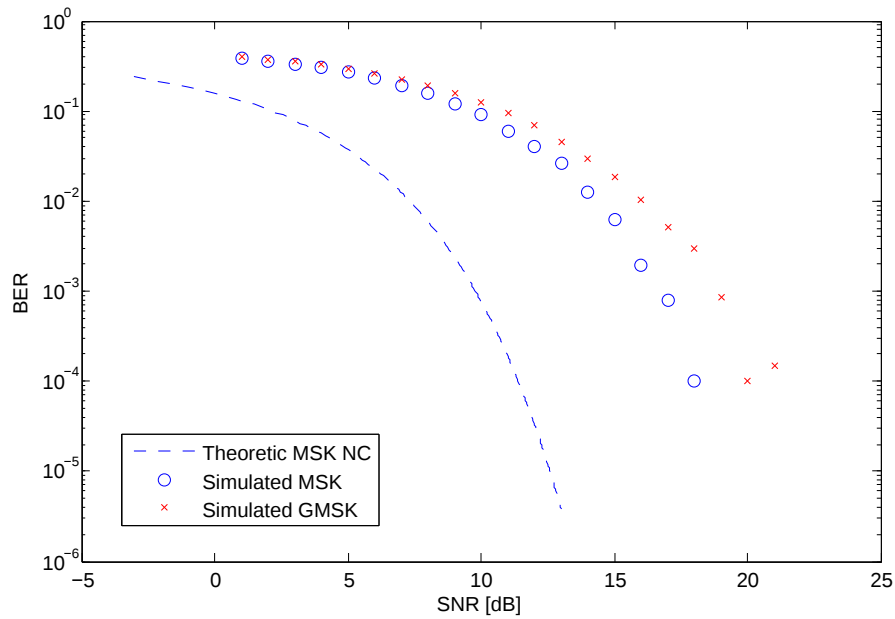


Figure 4.11: Test of GMSK demodulator

the total number of bits to calculate the BER. This is done both for MSK and GMSK with BT product 0.5.

4.8 AIS Data Extraction

The AIS data extraction performs the inverse operations of the data encapsulation. Each operation is explained in the following section.

4.8.1 NRZI Decoding

The first operation when receiving the raw demodulated bit stream, is the NRZI decoding. In Code Example 4.14 the procedure for this step is shown. The overall idea is to look at the bit shifts to decide whether a bit transition has occurred. In such case, a 0-bit has been transmitted, otherwise a 1-bit was sent.

```

1 for ii=1:length(Uj)-1
2     if Uj(ii) ~= Uj(ii+1), Uj(ii) = 0;
3     else, Uj(ii) = 1;, end
4 end

```

Matlab Code 4.14: NRZI decoding

4.8.2 HDLC Decoding

After the NRZI decoding, the start and stop flags must be found. Code Example 4.15 shows this procedure in line 1. The code example also shows the extraction of the packet from the sequence (line 6) after it has been checked whether both the start and stop flag has been found. After the extraction, it is checked whether the packet is long enough to be recognised as a valid AIS packet.

In such case, the bit destuffing is executed by searching after five consecutive 1-bits in a row, after which the following 0-bit bit is removed.

```

1 placement = findstr(Uj,[0 1 1 1 1 1 1 0]);
2
3 % Packet found?
4 if length(placement) > 1
5     % Extract pack from start and stop
6     rawpack = Uj(placement(1)+8:placement(2)-1);
7
8     if length(rawpack)>150
9         % Bit destuffing
10        bitstufplacement = findstr(Uj,[1 1 1 1 1]);
11        for ii = 1 : length(bitstufplacement)
12            Uj(bitstufplacement(ii)+5-(ii-1))=[];
13        end
14        ...
15    end
16 end
17 end

```

Matlab Code 4.15: HDLC decoding

4.8.3 Data Extraction

When the packet has been extracted, a CRC check should be executed to validate the packet content. A CRC check has been prepared to be implemented, although the implementation of the CRC routine itself is outside this project scope. Code example 4.16 shows the procedure to get the message ID, which is saved as Ij(1). The MMSI number of the transmitting ship (Ij(2)) and the FCS checksum are extracted in the following lines.

```

1 Ij(1) = bin2dec(strrep(int2str(Uj(1:6)), ' ', ''));
2 Ij(2) = bin2dec(strrep(int2str(Uj(9:38)), ' ', ''));
3 Ij(3) = CRCcheck(Uj);

```

Matlab Code 4.16: Data extraction.

Proof of Concept

To proof the receiver concept, a simple version of the receiver is implemented. For simplicity, a standard PC with MATLAB is used for signal processing and an analog sampling card is used to sample the output from a development board radio receiver.

The ADF7021 development board used by 09gr650 is used for the proof of concept. The receiver is configured over SPI, using a microcontroller platform developed for AAUSAT3 (AT90CAN128 [08gr414 08]). The receivers Local Oscillator (LO) is programmed to 161.925 MHz, thereby centering AIS ch. 2 (162.025 MHz) at the 100 kHz IF frequency. The Automatic Frequency Control (AFC) is disabled to ensure that the LO is fixed, rather than searching for a signal. The Automatic Gain Control (AGC) is used to ensure sufficient amplitude of the output. The ADF7021 has a differential output after the IF filter, thereby giving a 25 kHz bandwidth centered around the IF frequency, and providing image rejection.

A standard PC and an analog sampling card are used to sample the differential IF output using two analog 16-bit channels at 500 kSps. Furthermore, the Low Noise Amplifier and antenna described in section 6.1 are used for the receiver. Finally, the setup is places on the roof of the university, to ensure good reception and minimize obstacles in the line of sight.



Figure 5.1: The Proof of Concept setup on the roof.

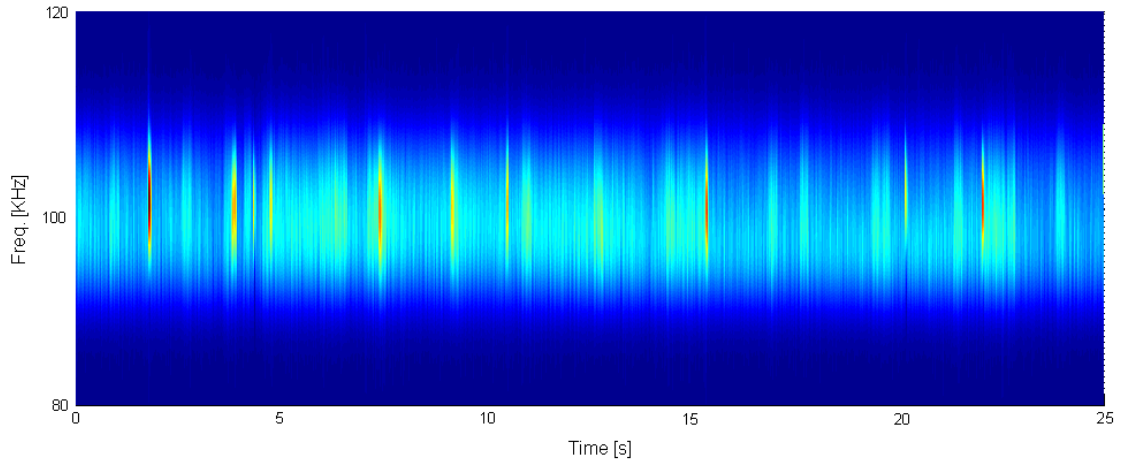


Figure 5.2: FFT waterfall plot of raw received data.

Figure 5.2 shows an FFT waterfall plot of 25 seconds of sampled data, in the frequency range from 80 to 120 kHz. AIS packets are clearly visible as high amplitude bursts centered at 100 kHz. The raw sampled data can be found in `2.mat` on the attached CD.

Figure 5.3 shows the correlator demodulator output from time 22.1 sec. and onwards. The green dots indicate a detected zero-crossing, and the blue crosses indicate where to apply the decision rule to read a bit value. The output shows the 20 bits of NRZI encoded training sequence, the 0x7E HDLC start flag and the first 10 bits on the AIS packet. Figure 5.4 is generated by

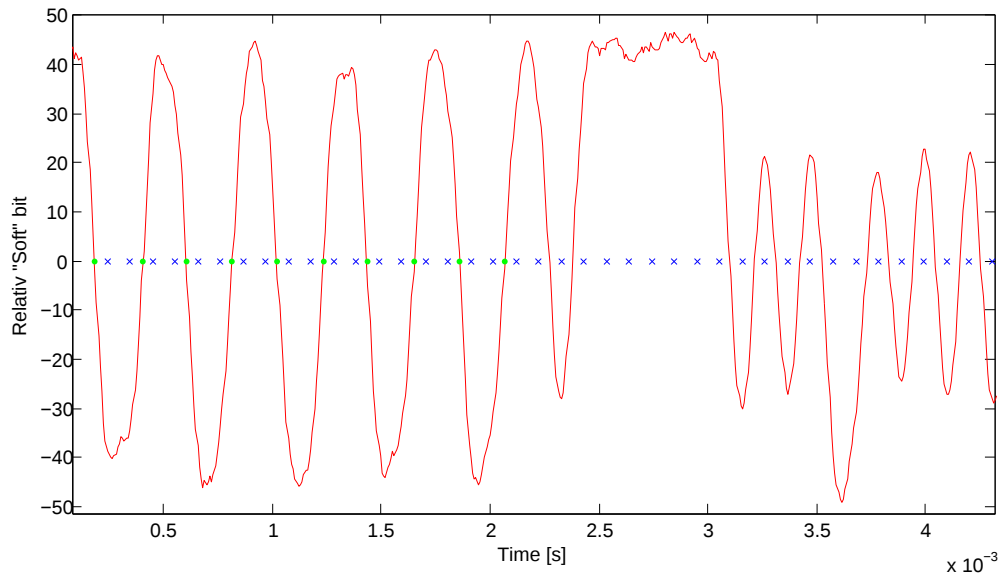


Figure 5.3: Correlator output from 22.1 sec. and forward.

reading the output of the two correlators on the estimated bit-placement positions. The content of each of the orthonormal basis functions ψ_1 and ψ_2 is plotted to show the signal constellation of the received package. The dashed line illustrates the decision rule boundary. The hard decision of the bit placements yields the following bit sequence:

01100110 01100110 11111110 10100101 01010101 10010100 001100...

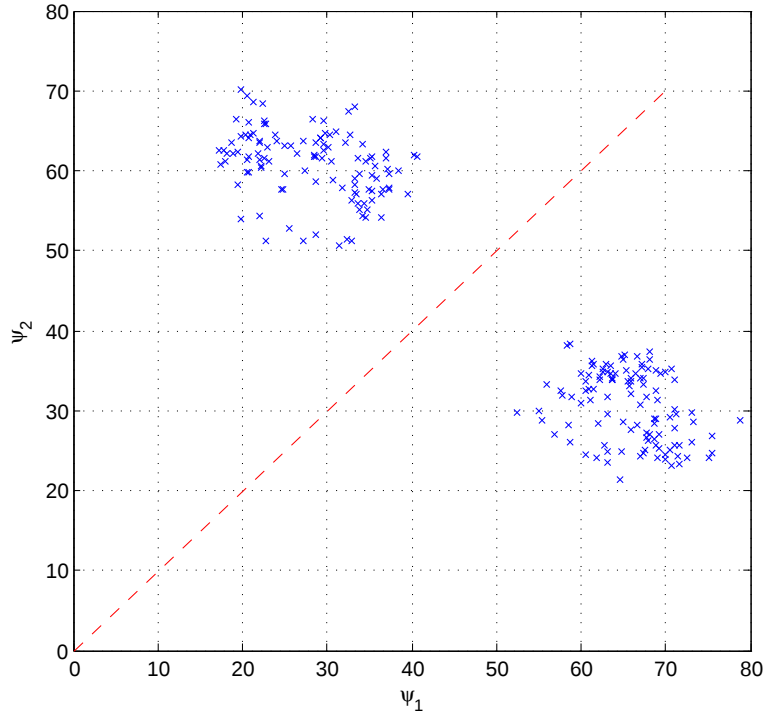


Figure 5.4: Signal constellation of the received package.

The initial state of the NRZI output stream is unknown, so the NRZI decoded data is:

?0101010 10101010 01111110 00001000 00000000 10100001 110101...

The raw data package, when omitting the training sequence and start flag (01111110), therefore is:

00001000 00000000 10100001 11010101 00101011 10111110 01010100 11010101...

After bit destuffing the packet is:

00001000 00000000 10100001 11010101 00101011 10111110 10101001 1010101...

HDLC frames are transmitted with LSB first, so the endianness is reversed to get MSB first:

00010000 00000000 10000101 10101011 11010100 01111101 10010101...

The 16-bit CRC16 checksum 01111000 10000011 is successfully validated with software written by the user Steve Underwood from the signal processing forum at dsprelated.com [dsprelated.com 07]. His code can be found in `checkcrc.c` on the attached CD. The first 6 bits in the packet indicate the Message ID. Packet type 000100 = 4 is Base station reports. The rest of the packet is decoded according to the AIS standard [ITU 07, p. 100]. The output parameter, bit placement, value and description are listed in table 5.1. The longitude and latitude is converted to degrees:

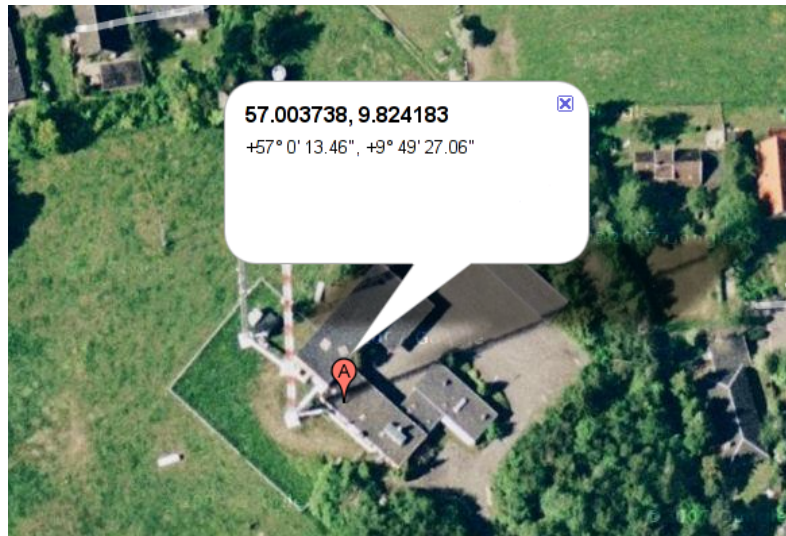
$$\text{Longitude} = \frac{5894510}{10000 \cdot 60} = 9.824183^\circ E \quad (5.1)$$

$$\text{Latitude} = \frac{34202243}{10000 \cdot 60} = 57.003738^\circ N \quad (5.2)$$

Parameter	Number of bits	Value	Description
Message Type	6	4	Identifier for this message
Repeat indicator	2	0	Number of message repetitions.
User ID	30	002190069	MMSI number
UTC year	14	2009	1-9999; 0 = UTC year not available
UTC month	4	5	1-12; 0 = UTC month not available
UTC day	5	21	1-31; 0 = UTC day not available
UTC hour	5	12	0-23; 24 = UTC hour not available
UTC minute	6	19	0-59; 60 = UTC minute not available
UTC second	6	51	0-59; 60 = UTC second not available
Position accuracy	1	0	1 = >10 m; 0 = <10 m; 0 = default
Longitude	28	5894510	Longitude in 1/10 000 min
Latitude	27	34202243	Latitude in 1/10 000 min

Table 5.1: Received AIS package decoded.

This position is verified by finding the position of MMSI 002190069 online. MMSI 002190069 is known as the *Frejlev Transmitter*, placed approximately 10 km from the experiment setup. MMSI numbers starting with 00 are base stations, while 219 is the Maritime Identification Digits used for danish ships and base stations. Figure 5.5 shows a screenshot from Google Maps of the transmitter position.

**Figure 5.5:** The Frejlev Transmitter in Google Maps.

Implementation

Figure 6.1 shows the signal path through the AIS receiver. The implementation of each block, from antenna to signal processor, is described in this chapter. The algorithms from Chapter 4 are not implemented on the Digital Signal Processor, so the section on the DSP is concentrated on selecting a suitable implementation platform for the BEXUS flight.

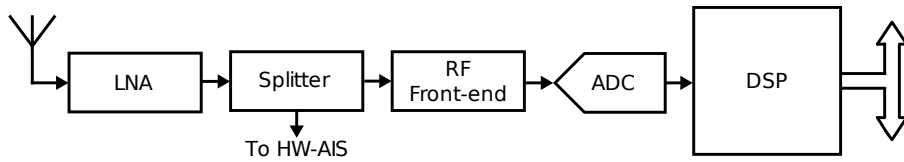


Figure 6.1: Hardware overview.

6.1 VHF Antenna

A maritime VHF dipole designed for signals with a frequency of 156–162 MHz is used for the receiving antenna. The length of the antenna itself is 90 cm which approximately makes it a half wave dipole for AIS signals:

$$\frac{\lambda}{2} = \frac{c}{2 \cdot 162 \text{ MHz}} = 92.5 \text{ cm} \approx 90 \text{ cm} \quad (6.1)$$

The impedance of a half wavelength is approximately $73 + j42.5 \Omega$, but the antenna is impedance matched to 50Ω , which is the same as the LNA and cables used in the signal path.

6.2 Low Noise Amplifier

The purpose of the Low Noise Amplifier (LNA) is to amplify the signal close to the VHF antenna, in order to introduce as little noise as possible. The antenna will be mounted underneath the BEXUS gondola, so the LNA and a filter are placed inside a small box suitable for mounting external to the gondola.

6.2.1 Mechanical Design

The LNA must have interfaces to the antenna and the experiment box. The antenna is fitted with a female UHF connector, so for simplicity this type of connector is used for both interfaces

to the LNA box. This also makes it possible to remove the LNA and test the connection without extra amplification. A male-male adapter will be used to connect the LNA case to the antenna. The box is 80x55x25 mm and weighs 79 g. The assembled LNA box is shown in Figure 6.2.



Figure 6.2: LNA box with UHF connectors.

6.2.2 Electronic Design

The Maxim MAX2371 EVKit development board is used for the LNA. A 3rd order Chebyshev low pass filter with 0.5 dB ripple, designed to attenuate UHF frequencies, is mounted directly on the development board. The normalized Chebyshev filter is calculated using the MATLAB function `cheby1`. Scaling the transfer function to a 180 MHz cutoff frequency yields the following transfer function:

$$H(s) = \frac{1.0352E27}{s^3 + 1.4168E9 \cdot s^2 + 1.9630E18 \cdot s + 1.0352E27} \quad (6.2)$$

Equation (6.2) is realized as a third order LC ladder filter of two capacitors and one inductor. An LC ladder filter is chosen to have as few components as possible through the signal path. The circuit realized with ideal components is illustrated in Figure 6.3.

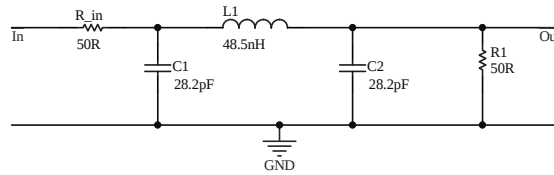


Figure 6.3: CLC filter with ideal components.

Two simulations of the filter has been performed. One with ideal components and one where the ideal components are changed to match the ones available in the department laboratory. The schematics for the simulation is shown in Figure 6.4, while the results of the simulations are shown in Figure 6.5.

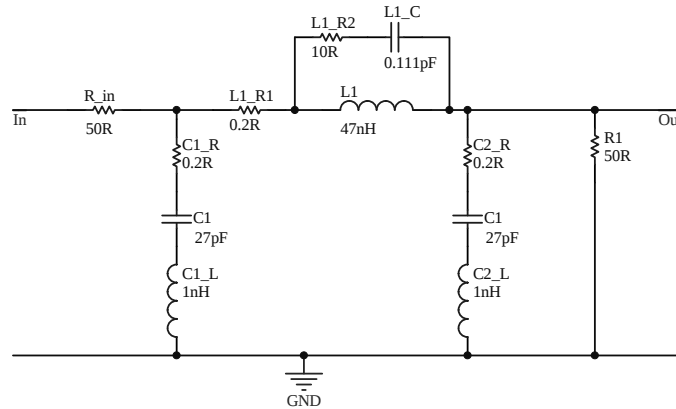


Figure 6.4: CLC filter with real components.

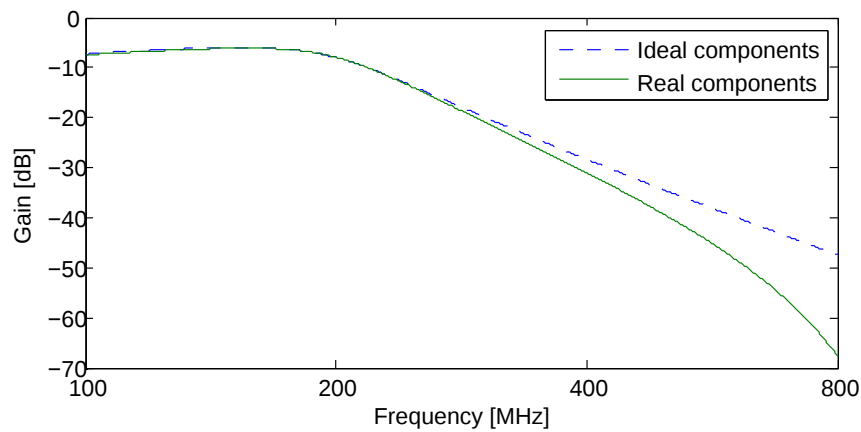


Figure 6.5: Simulation of Chebychev filter with real and ideal components.

6.2.3 Test

The LNA configuration and filter performance has been tested with a signal generator and a network analyzer. The main reason for using the filter is the transmitter from the service module on the BEXUS balloon, transmitting at 402.2 MHz [SSC 09]. Therefore special attention is given to the attenuation at this frequency. Figure 6.6 and 6.7 plots the LNA amplification with and without the filter applied. The attenuation of approximately 30 dB at 402 MHz is assessed to be sufficient, without disturbing the desired signal at 162 MHz. At 162 MHz, the LNA gives a gain of approximately 14 dB, which is considered enough for the BEXUS flight.

6.3 SAW Filter

To remove as much unwanted signal as possible, a SAW filter is inserted in the signal path. The main disturbance will be the BEXUS EBASS signal transmitted on 402.2 MHz [SSC 09]. SAW filters designed for a 162 MHz pass band and with a low insertion loss are rare, as the frequency is not used by much equipment. However, the LBT16201 from the Chinese company SIPAT matches the AIS bandwidth very well. The LBTs frequency response is shown in Figure 6.8. The filter has a relatively high insertion loss of 2 dB, and is therefore placed after the signal has been amplified by the LNA. Unfortunately, the filter did not arrive before the project submission and is therefore not tested.

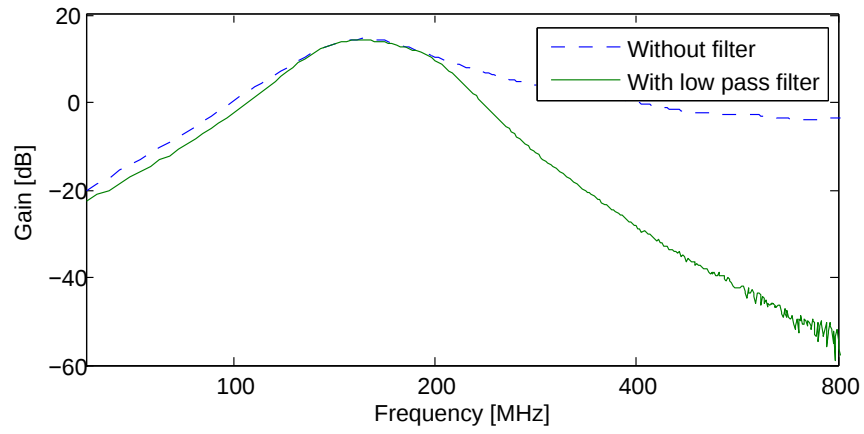


Figure 6.6: Test of LNA with and without filter applied.

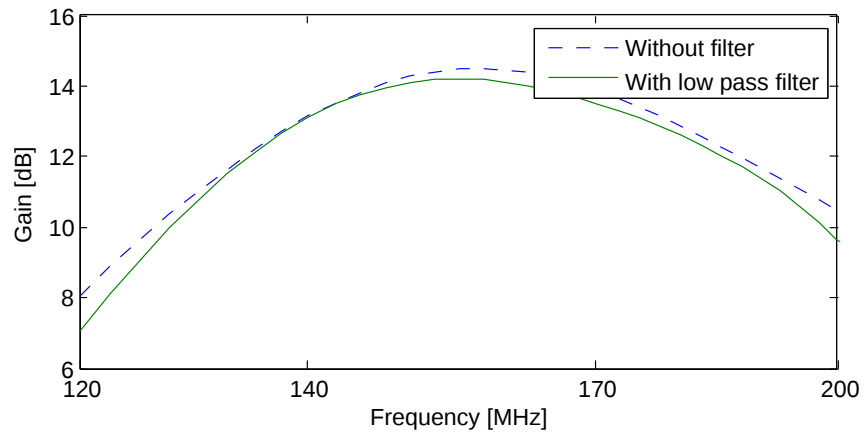


Figure 6.7: Test of LNA with and without filter applied, focus on AIS frequencies (162 MHz).

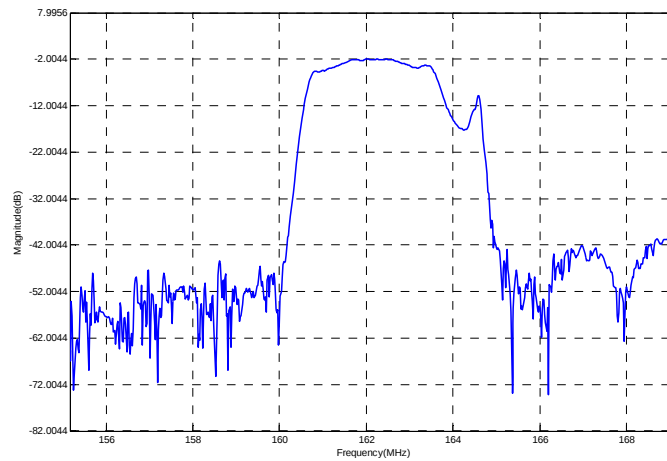


Figure 6.8: SAW filter frequency response [SIPAT Co. 09].

6.4 Power Splitter

A power splitter must be inserted to split the signal to the other AIS receiver on the BEXUS experiment, as only a single VHF antenna is mounted on the balloon. An ADP-2-1 passive splitter by Mini-Circui is used due to its low insertion loss (0.3 dB at 160 MHz) and small footprint. A frequency plot of the power splitter is shown in Figure 6.9.

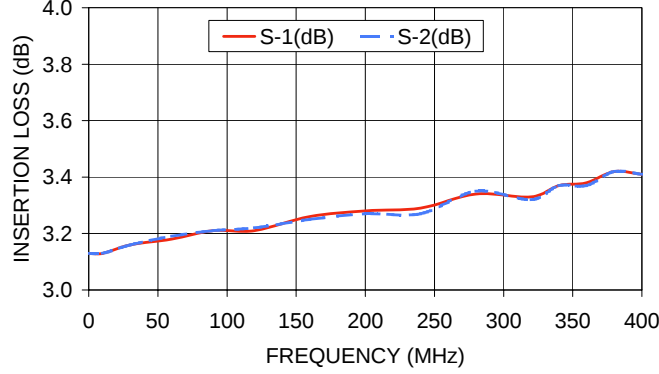


Figure 6.9: Power Splitter insertion loss.

6.5 RF Front-end

ADF7020-1 from Analog Devices is used as radio front-end for the software defined AIS receiver. The ADF7021 with similar configuration, is used by 09gr650 and for the UHF radio on AAUSAT3. The ADF7020-1 is able to output the in-phase, I and quadrature, Q signals downconverted to a center frequency of 200 kHz. The output uses pins I , $\bar{I}$ (I inverted), Q and $\bar{Q}$ (Q inverted) and measurements indicates a DC offset of approximately 1.1 V.

A configurable image rejection pass band filter is available in the front-end. Figure 6.10 shows the filter response by different passband bandwidths. The Intermediate Frequency (IF) I and Q output are not well documented in the ADF7020-1 datasheet. The pins are referred to as "Signal Chain Test Pins" and described as "High Impedance Output" [Analog Devices 05, p. 11]. Therefore a test is performed to evaluate the I/Q output possibilities and their relation to the AGC with the lower and upper threshold in the AGC configured to -70 dBm and -30 dBm. The results are listed in table 6.1. It is possible to adjust the AGC to change the output voltage. Schematics for the frontend can be found in Appendix B.

Input level [dBm]	AGC setting	I/Q output peak-peak [V]
-70	Automatic	60 mV
-80	Automatic	60 mV
-90	Automatic	60 mV

Table 6.1: Measured I/Q output vs. AGC settings and input.

6.5.1 Configuration

The ADF7020-1 is configured by setting 13 registers through SPI. As the IC can be used for both receiving (including hardware based demodulation) and transmitting, not all of these registers are actually configured.

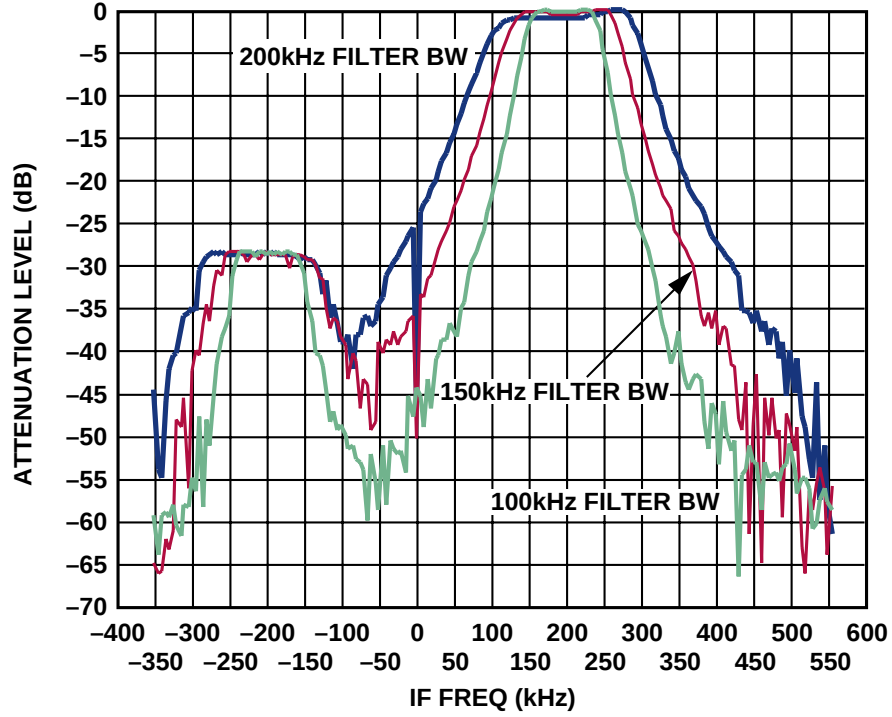


Figure 6.10: ADF7020-1 intermediate filter [Analog Devices 05].

The most important part is to configure the receiver frequency. A 19.2 MHz crystal oscillator is chosen for external oscillator, after which the appropriate properties is configured as follows in Equation (6.3). Note that the frequency of the Phase-Locked Loop (PLL) must be 200 kHz lower than 162 MHz to get the signal centered at 200 kHz. The frequency is calculated to be the twice the desired frequency ($= 323.6$ MHz), and then halved by using the VCO divide by two property.

$$\begin{aligned}
 F_{\text{PLL}} &= \frac{XTAL}{R} \cdot \left(N_{\text{Integer}} + \frac{N_{\text{Fractional}}}{2^{15}} \right) \quad [\text{MHz}] \quad (6.3) \\
 &= \frac{19.2 \cdot 10^6}{2} \cdot \left(33 + \frac{23211}{2^{15}} \right) = 323.6 \text{ MHz}
 \end{aligned}$$

The filter bandwidth of the front-end is configured based on the maximum bandwidth of the signals being 25 kHz with a deviation of 50 kHz. When a Doppler shift and oscillator frequency drift of 2 times 5 kHz is added, the maximum bandwidth needed is 85 kHz. Therefore, a filter bandwidth of 100 kHz with a center bandwidth of 200 kHz is used.

6.6 Analog to Digital Converter

The Analog to Digital Converter (ADC) is used to convert the downconverted signals I , $\bar{I}$, Q and $\bar{Q}$ to a serial digital signal that is readable from the DSP. Due to the Nyquist theorem, the sample rate should be at least twice the highest frequency component of the input signal. The RF front-end outputs signals centered at 200 kHz so an ADC with a maximum sample rate of 1 million samples per second is used.

The Analog Devices AD7262 [Analog Devices 08] is a dual differential ADC, with an adjustable sample rate of maximum 1 Msps. The ADC has a built-in amplifier, such that the signals with a peak-peak amplitude of 60 mV can be amplified to get an optimum digital resolution. The ADC expects that the input signals has a DC average of 2.5 V compared to 1.1 V that is the average output of the front-end IC. To compensate from this, a change in the DC offset is created by four

high pass filters and a quad operational amplifier. The operational amplifier is controlled by a reference voltage from the ADC. The design of the schematic can be found in Appendix B.

6.6.1 Interface and Configuration

As mentioned, the ADC is configured and used through the serial interface. The ADC contains one 12-bit control register, where the desired gain and an optional power-down mode is selected, and an internal calibration routine is started. It is possible to read back the result of the calibration.

Measurements are carried out from the ADC by sending a continuous clock to it. It is possible for the ADC to send measurements on either one or two channels. It is obvious that the requirements for the board layout is less stringent if the data is read over two channels.

6.7 Digital Signal Processor

This section describes the selection of implementation platform for the demodulator, for which two DSPs from Texas Instruments and Analog Devices have been considered. Both DSPs provide a high speed serial interface for data acquisition from the ADC but differ in a number of other features. The DSPs are described in the following two sections.

6.7.1 Texas Instruments OMAP3530

The OMAP3530 from Texas Instruments is a 600 MHz ARM Cortex-A8 and a 430 MHz TI TMS320C64x+ 16-bit fixed point DSP integrated in a single package. The OMAP supports up to 1 GB of DDR memory which can be soldered on using the Package-on-Package (PoP) technique, where the memory chip is installed on top of the OMAP. The processor includes a high speed Multi-channel Buffered Serial Port (McBSP) for data transfer from the ADC. The OMAP3530 does not support CAN, so an external controller is thus required for communication with the other subsystems on the satellite.

A low-cost community designed OMAP3530 development board is available through the Beagleboard project [Beagleboard 09]. The Beagleboard hardware design is open and the boards ARM processor is able to run the Ångström Linux distribution, which specifically targets embedded systems such as multimedia players and handhelds. Texas Instruments provide a `dspbridge` module for the Linux kernel to interface with the TMS320C64x+, but a license to TI Code Composer Studio is still required to compile code for the DSP. An open GNU toolchain, used for building the Linux distribution, exists for the ARM Cortex-A8. A growing community have emerged around the Beagleboard, but experience with development for the integrated DSP is limited, with only a few code examples available.

6.7.2 Analog Devices Blackfin BF537

The Analog Devices ADSP-BF537 Blackfin is a 500 MHz single core 16-bit fixed point DSP with a RISC-like instruction set. The DSP incorporates the Micro Signal Architecture designed as a cooperation between Analog Devices and Intel. The BF537 supports maximally 512 MB of SD-RAM memory and includes two high speed Serial Ports (SPORT) for data acquisition from serial peripherals as well as a CAN 2.0B compliant CAN interface.

An open development board using the BF537 is available in the ADZS-BF537-STAMP board from Analog Devices. The STAMP board is targeted to support the development of Linux on the Blackfin Series DSPs. The uClinux distribution with the U-Boot boot loader and a complete implementation of the GNU toolchain (`gcc`, `ld`, `as` etc.) is provided by the Blackfin Koop project which is supported by Analog Devices [Blackfin Koop 09]. The Blackfin Koop website hosts a well established community and comprehensive documentation with code examples for Linux on the Blackfin architecture. Linux device drivers for the Blackfin peripherals, including CAN and SPORT, are included in the uClinux distribution.

6.7.3 Score Chart

The score chart in Table 6.2 assess the features described in the previous sections. The OMAP-3530's dual processing unit architecture is superior when comparing computational power, but increases the complexity of software development. An OMAP-based demodulator would thus require code for both the ARM Cortex-A8 and TMS320C64x+ which could introduce parallelization issues and requires knowledge of two architectures. The need for the dspbridge kernel module is a minor drawback as the module is still in development. Open design development board exists for both processors which reduces the workload when designing hardware for the system. The Package-on-Package technology used in the OMAP3530 allows easy placement of memory, whilst the Blackfin requires memory devices placed on the PCB. The BF537 Blackfin's open toolchain is widely used and increases the reproducibility of the receiver, as an expensive IDE such as Code Composer Studio, is not required. The Blackfin includes a CAN interface, which is required for communication with the other subsystems in the satellite. An external CAN transceiver is required for the OMAP3530 to exchange data with the other subsystems. Finally, a large Blackfin community and the lack of available documentation for Linux with the DSP in the OMAP3530 is in favor for the BF537. The final selection is therefore to use Analog Devices ADSP-BF537 with embedded uClinux.

TI OMAP3530		ADI BF537	
Dual Core Processing Power	+	Single Core Processing Power	-
Dual Core Complexity	-	Single Core Complexity	+
Linux + DSPBridge	-	Native Linux	+
Code Composer Studio	-	Open Source Toolchain	+
Open Design Beagleboard	+	Open Design ADZS-BF537-STAMP	+
PoP Memory	+	Separate RAM ICs	-
No CAN Controller	-	Integrated CAN Controller	+
Total +'s	3	Total +'s	5

Table 6.2: DSP Score Chart. A + indicates advantages while - indicates disadvantages.

6.8 Analog Devices ADSP-BF537 Architecture

This section gives a more detailed description of the internal architecture of the ADSP-BF537 from Analog Devices. Further information is available in the ADSP-BF537 datasheet [Analog Devices 09a] and Hardware Reference [Analog Devices 09b]. Figure 6.11 shows a functional block diagram of the Blackfin core and the main peripherals and data buses. The core contains two 16-bit MACs (Multiply-Accumulate), two 40-bit ALUs (Arithmetic Logic Unit), two 40-bit accumulators and a 40-bit barrel shifter. The two MAC units are both able to execute one 16-bit by 16-bit multiplication and accumulate the result in one of the 40-bit accumulator in a single clock cycle. The 40-bit ALUs can execute regular 16 or 32-bit arithmetic instructions, and includes special DSP instructions such as modulo 2^{32} multiplication and saturation/rounding arithmetics useful for signal processing. The barrel shifter is used for bit shifts and rotations. The hardware supports zero-overhead loop structures, which are useful for speeding up signal processing algorithms such as correlations, FFTs and filters. The Blackfin processors use a modified Harvard architecture with dedicated L1 memory blocks for data and instructions. Stack and local variables are stored in a special L1 scratchpad memory. The Blackfins Memory Management Unit (MMU) does not support virtual memory, but provides supervisor and user mode memory protection. The uClinux kernel exploits this for separating user space applications from the kernel memory. Internal and external memory blocks, including peripheral I/O registers and Memory Mapped Registers (MMR) are located in a common 4 GB address space, of which up to 512 MB is available for external

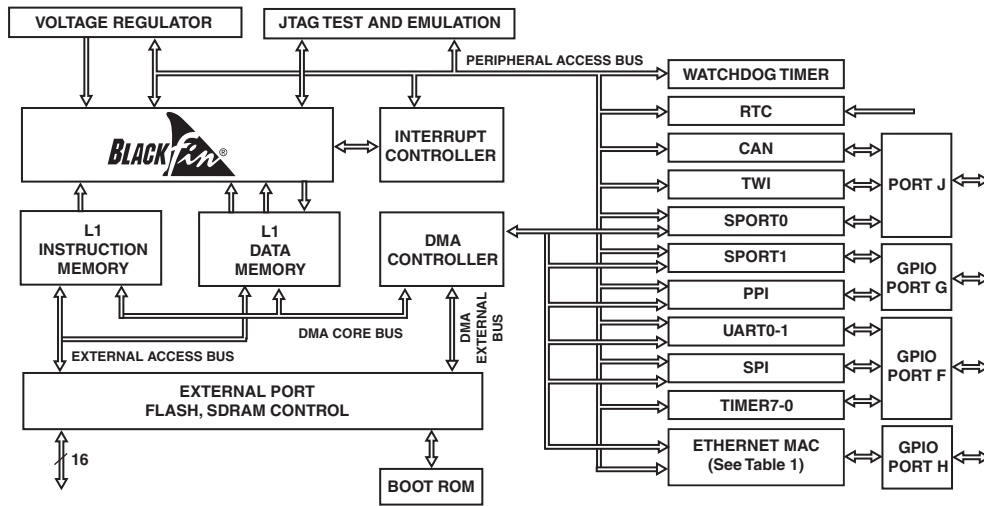


Figure 6.11: ADSP-BF537 Blackfin architecture [Analog Devices 09a].

SDRAM. Two full duplex synchronous serial ports (SPORT0 and SPORT1) are included in the

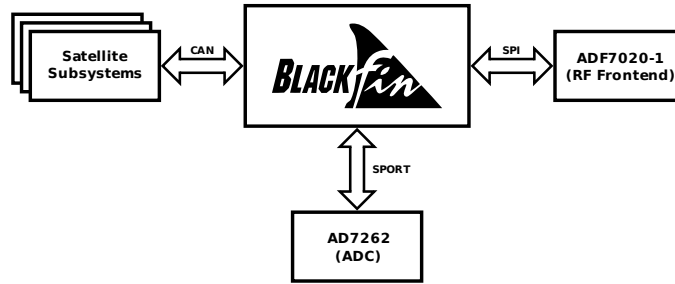


Figure 6.12: Peripheral connections for the ADSP-BF537 AIS receiver.

ADSP-BF537 architecture for high speed transfers. The SPORTs implements bidirectional transfers with support for Direct Memory Access (DMA), allowing data acquisition without blocking the processing unit for the transmission time. Figure 6.12 shows how the peripherals are used in the AIS receiver design. Sampled data from the ADF7020-1 in-phase and quadrature outputs is transferred from the AD7262 ADC using the SPORT. The ADF7020-1 is configured using Serial Peripheral Interface (SPI) bus. Communication with the satellite subsystems uses the built-in CAN controller.

6.9 PCB Schematics and Layout

To test the front-end of the software defined AIS receiver, a PCB is designed. This includes all of the hardware between the LNA and DSP. The desired features of the PCB is:

- LNA connector (RF input)
- Saw filter and power splitter
- Down converter and IF filter
- Analog to Digital converter

- Connector to DSP interface
- Data storage
- AAUSAT3 mech. outline and stack connector.

The circuit and layout is inspired by the application notes for the ADC[Analog Devices 08, p. 30] and the ADF7020-1 development board[Analog Devices 07b]. To follow some of the basic EMC rules[Armstrong 07], the PCB is designed using a standard 4 layered FR4 print type. The two inner layers are used for uncorrupted V_{cc} and GND plane, and the top and bottom layers are used for signalling. The PCB has been produced by Elprint A/S, via the normal AAU discount procedure. Figure 6.13 shows the layers of the PCB.

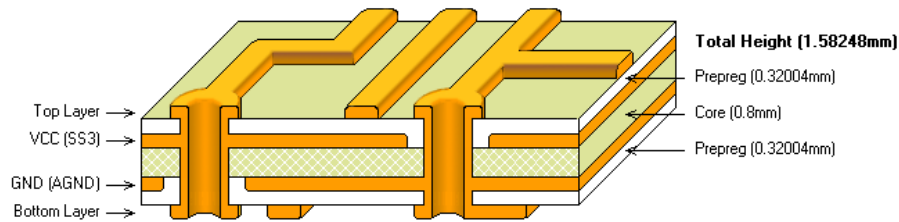


Figure 6.13: PCB Layer stackup.

Figure 6.14 shows a photo of the final PCB. The board outline is as defined in the AAUSAT3 standard with shape, holes and stack connector matched to the standard. On the top right is the interface for the DSP development board, connecting the microSD card and the ADC to the DSP (just beneath the DSP connector). The stack connector is placed in the bottom left corner, and supplies the board with power and CAN communication. The CAN transceiver is placed in the left side. The right side of the PCB is the HF part, with SMA connector for RF input. The HF filtering (SAW filter, power splitter and the possibility to add extra π - or T -filter) is placed in a square, prepared for EMC shielding. In the middle of the board is the downconverter and IF filter (ADF7020-1), with the debug and test connector just left of it. The differential I/Q signal from the AFD7020-1 is offset adjusted and connected to the ADC input.

The full schematics and PCB layout is shown in section B on page 65, followed by the component list.

6.10 Experiment Setup

The two AIS receivers and a prototype of AAUSAT3 will be included on the BEXUS flight in October 2009. The balloon will pass through the Earth's atmosphere to an altitude of approximately 35 km and experience temperatures from 20 °C to less than -80 °C. Figure 6.15 plots the expected ambient temperature as a function of the balloon altitude. Consequently, the experiment must be able to operate in this temperature range. This section explains the mechanical structure and thermal considerations for the experiment.

6.10.1 Mechanical Structure

The experiment subsystems will be mounted in a modified AAUSAT-II frame, with the frame height extended to 16 cm to accommodate six batteries. The experiment will not have solar panels, so extra batteries are needed to have sufficient power for 6 hours of operation and for heating the experiment. The entire frame will be fitted inside an insulated box for mounting on the balloon gondola to ensure a reliable temperature for the essential parts of the experiment. The outer box is constructed of SE20-3 aluminum corners from System Standex [System Standex A/S 09] and 20x20 mm aluminum profile with 2mm walls. The total weight of the outer box is 2986 g.

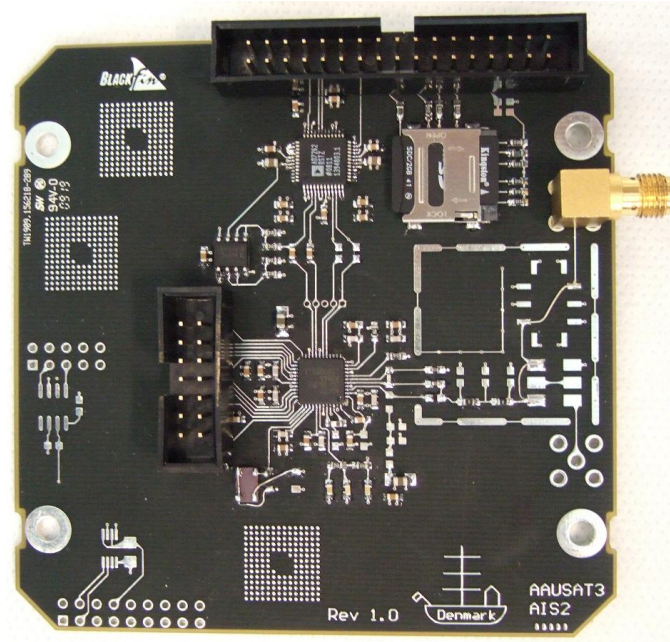


Figure 6.14: Photo of front end PCB Rev 1.0.

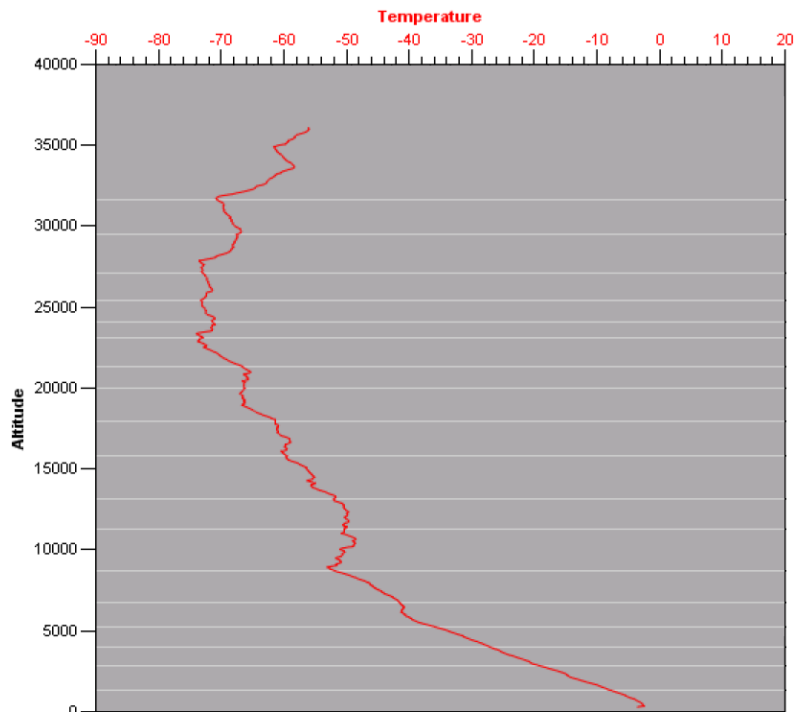


Figure 6.15: Expected temperature for the experiment[SSC 09].

Figure 6.16 shows the outer box with insulation and the extended frame. The final experiment box will be fitted with aluminum sides to keep the experiment and insulation foam in place. One side of the box will contain connectors for VHF and UHF antennas and the umbilical cable used for charging and external access to the CAN bus.

The PCBs are designed in accordance with the AAUSAT3 PCB layout standard illustrated in Figure 6.17.



Figure 6.16: Insulated experiment box and frame.

Number of	Description	Weight[g]	Total Weight [g]
8	Corner	74	592
4	220 mm tube	86	344
8	160 mm tube	63	504
4	260x200 mm Side shield	281	1124
2	200x200 mm End shield	216	432
Total			2986

Table 6.3: Mass budget for the experiment box.


$$A = \begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{pmatrix} \in GL_3(\mathbb{F}_2) \quad \text{with} \quad f = \begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{pmatrix} \in \mathbb{F}_2^3 \quad \text{and} \quad f: \mathbb{F}_2^3 \rightarrow \mathbb{F}_2^3$$

To ensure that the batteries will supply as expected, it is desired to calculate the power usage

Where the surface area of the Cubesat frame is give by:

Experiments have shown that the foam has an insulating effect such that 2.7 W dissipated inside

Usage	Component	Speed	Precision
RF downconverter	IQD FREQUENCY PRODUCTS - LF TVXO018792	19.2 MHz	± 3 ppm
DSP clock	EPSON MA-505 M-co	25.0 MHz	± 50 ppm

Table 6.4: Precision of the oscillators.

the box raises the temperature by 28 °C. By this, formula (6.4) can be rewritten, such that the temperature deviation depends only at the dissipated power times a constant. The constant is defined as:

$$K = \frac{P}{\Delta T} = \frac{2.7 \text{ W}}{28 \text{ K}} = 96.4 \frac{\text{mW}}{\text{K}}$$

The necessary power to secure an internal temperature of approximately 0 °C, when the lowest expected temperature is −80 °C will be:

$$K \cdot (0 \text{ °C} - -80 \text{ °C}) = 7.7 \text{ W} \quad (6.5)$$

This heat will be generated by the subsystems themselves and with power resistors placed around the batteries. The resistors will be actively controlled based on the inside temperature of the experiment. Experiences from AAUSAT-II has shown that the inner heat transport will be sufficient to keep all subsystems within their designed range.

6.10.3 Clock Drift

With the expected internal experiment temperature and the chosen components for the design, is it now possible to calculate the worst case clock drift. Due to table 6.4 the worst possible clock drift will be 53 ppm. On basis of this, frequency drift at 162 MHz will be as follows:

$$D_{\text{clk}} = 162 \cdot 10^6 \cdot \frac{53}{10^6} = 8.6 \text{ kHz} \quad (6.6)$$

Added to the calculated Doppler shift of up to ± 4 kHz (see section 2.1.3) the total worst case frequency drift will be 12 kHz. This can be minimized by choosing another oscillator for the DSP. It is possible to get temperature compensated oscillators which fits to the DSP with a frequency drift of less than 50 ppm. This will bring down the total worst case frequency deviation including Doppler shift to ± 5.3 kHz which will be sufficient for the algorithms.

Acceptance Test

1. *Be able to sample raw data which contains both AIS channels with a sufficient sample rate to satisfy the Nyquist sample theorem.*

The front end and ADC setup is investigated, and it is verified whether the sample rate is sufficient. As the DSP and SPORT interface for the ADC has not been implemented, this have not been tested although the setup is prepared for sampling with 1 Msamples per second. The I/Q signal from the front end (ADF7020-1) is sampled with a PC, and it is verified that the IF frequency is centered around 200 kHz and with its highest frequency component under 300 kHz. Therefore it is concluded that the selected setup is sufficient.

2. *Demodulate GMSK in software.*

An antenna is connected to the system, and the software demodulator is tested. It is shown in Chapter 5 that the software demodulator is working on real data in MATLAB, using ADF7021 for downconversion and thereby receiving only one AIS channel. At the same time it is shown through simulation that the algorithms will work, using ADF7020 and demodulating both AIS channels at the same time.

3. *Be able to update all parts of the demodulating and decoding software remotely.*

A changed version of the demodulating and decoding software is written and uploaded to the subsystem, after which it is tested whether the changes takes effect. As the software on the DSP has not been implemented yet, it is not possible to test this requirement.

4. *Be able to compensate for frequency drift, including Doppler shift when the satellite is moving with 7.5 km/s .*

An artificial signal with a worst case frequency drift in both directions is generated using the GMSK simulator, and it is observed whether the signal is demodulated correctly. Only the Doppler shift has been simulated and tested. However the algorithm has proven usable, and the up to 200 kHz IF bandwidth is be enough to compensate for clock drift also.

5. *Be able to store raw sampled data for 3600 sec (1 hr).*

The receiver is admitted to run until the memory limit is reached, and the amount of raw data is observed. The DSP and SPORT interface is not implemented, so the test is not executed. However, the SD storage should prove sufficient.

6. *Be functional from -40 - $+85^\circ \text{ C}$.*

The receiver is tested for worst case temperatures, and in the vacuum chamber. Only the DSP development board and SD card storage has been tested in climate chamber. No problem were encountered.

7. *Have a weight of maximally 150 g.*

The weight of the experiment setup is measured. The current running setup uses a PC for

signal processing. Therefore, the weight limit is exceeded. However, the front end and DSP should be able to fit one PCB and comply with the weight requirement. The weight of the PCB developed through this project is 43 g.

8. *Be operating for at least 6 months.*

As the final system is not implemented, the final burn-in test is not executed.

9. *Comply with the standards defined by the AAUSAT3 system engineering group.*

a) *Be able to run at 3.3 V and/or 5 V only.*

The required input voltages to the system is observed. 3.3 V and 5 V is needed.

b) *Use no more power than 1 W.*

The power consumption is measured with the system running fully operational. The estimated total power usage is around 1 W, depending on the needed DSP clock speed. Again, the missing final implementation means that this can not be fully tested.

c) *Comply with the PCB layout for AAUSAT3 including definition of stack connector and board outline.*

The size of the PCB is measured and the electric interface is tested. The PCB is 87 x 87 mm, and fits inside the produced frame. The final electrical interface has not been tested as the AIS2 PCB is the first PCB developed by AAUSAT3 standards.

d) *Have at least one temperature sensor placed on a central part of the PCB.*

Tested by observing and measuring using the on board sensor. The PCB layout is prepared for the temperature sensor, but the sensor has not been tested as the platform system that must interface to the temperature sensor has not yet been developed.

e) *Be able to be controlled completely over CSP.*

Commands is sent to the subsystem, and it is checked whether the correct messages is returned. CAN and CSP are not yet implemented.

f) *Send telemetry to the LOG subsystem.*

It is observed whether the relevant telemetry data is logged. As the DSP is not implemented, this is not yet tested.

Conclusion

The purpose of this project is to develop a software defined AIS receiver for the AAUSAT3 student satellite. The satellites mission objective is to evaluate the possibility to receive AIS signals in space. Previous scientific work by groups from Aalborg University and ESA indicates that uncollided messages with sufficient signal strength should be available in space. A satellite based AIS receiver will be able to cover a large area on ground, and enables the Danish Maritime Safety Administration to monitor vessels in the remote seas around Greenland.

A software defined receiver is desirable in space, as it allows for reconfigurations and optimizations of the algorithms after the satellite has been launched.

The AIS receiver and a prototype of AAUSAT3 will be tested on the BEXUS high-altitude balloon flight in October 2009. The BEXUS program allows European students to test scientific experiments in altitudes of up to 35 km and in temperatures below -80 °C. On the BEXUS balloon, the AIS receiver should be able to receive messages from most of the Baltic sea and parts of the Norwegian coasts.

The report begins with an analysis of the AIS standard with special focus on the physical and data link layers of the protocol. AIS uses a GMSK modulation scheme with NRZI encoded data and HDLC framing, so the receiver must be developed to support this standard. The signals are transmitted on two maritime VHF channels, with frequencies around 162 MHz. AIS transponders multiplex their transmission between the two channels, so the receiver must be able to cover signals at both frequencies. Furthermore, Doppler shift and oscillator drifts introduces frequency deviations that must also be taken into account. The analysis is used as a foundation for the requirement specification in Section 3.1.

The design chapter focus on the development of the receiver algorithms. A non-coherent correlator demodulator with support for packet detection and frequency estimation is designed and implemented in MATLAB. To verify the demodulator, algorithms are developed to generate realistic test data. A commercial radio front-end is chosen to downconvert the received signals to a frequency suitable for sampling with an ADC. The modulator is designed to simulate the sampled output of this front-end. A random bit stream is encapsulated in an HDLC frame and then NRZI encoded and GMSK modulated. Gaussian noise with an adjustable signal-to-noise ratio can be added to the signal, to simulate the transmission through a real channel.

The quality of the demodulator has been tested using this simulation setup. The required signal-to-noise ratio to achieve a Bit Error Rate of 10^{-3} is found to be approximately 8 dB above the theoretical value possible with a non-coherent GMSK demodulator. The demodulation algorithms could be optimized by using a quadrature demodulator and implementing soft decision. Adaptive bit synchronization could be implemented to minimize the effect of small drifts in the bit timing.

The developed receiver algorithms have also been tested with real-life AIS signals. A setup using a National Instruments sample card and a standard PC was installed on the roof of the university. Using a VHF antenna and a Low Noise Amplifier, data sets of 25 s was sampled and

processed with the MATLAB algorithms. Bit synchronization was achieved, and several packets were received and demodulated successfully.

For both the BEXUS flight and the final satellite, the receiver algorithms must be ported to a DSP. A DSP from Analog Devices is found to be suitable for the system. The current implementation includes a prototype PCB with a COTS radio front-end and an Analog to Digital Converter. The front-ends intermediate frequency output is sampled for processing in DSP. A connector is included for connection to the DSP development board.

Finally, the acceptance test shows that the implemented algorithms are suitable for receiving and decoding AIS signals. However, several requirement tests could not be passed, so additional work is needed. The deadline for the experiment delivery to the BEXUS flight is in early September, so the summer holiday will be used for further implementation and preparation for the BEXUS flight.

8.1 Further Work

This section lists the remaining tasks for preparing the software defined AIS receiver for the BEXUS flight.

The main task is to prepare the DSP for using the high-speed serial port for data acquisition from the ADC. A successfully sampled set of raw data from high altitude will be valuable for post-flight development of the demodulation algorithms. The BEXUS flight allows the sampling of data from varying altitudes, useful for evaluation of the SNR and collisions.

The current prototype PCB contains the radio front-end, the ADC and data storage for the DSP. The final flight model must furthermore include the DSP itself and necessary peripherals such as memory and oscillators.

Finally, the receiver algorithms must be ported and optimized for the DSP architecture, to allow real-time decoding. The AIS receiver must conform to the AAUSAT3 internal communication structure and implement the message protocols used for inter-subsystem communication in the satellite. It is desirable that the receiver is able to analyze the quality of the received data, to minimize down link bandwidth requirements.

Appendix A

Interface Control Documents

PRELIMINARY VERSION - TO BE CONFIRMED!

Be aware that this is a preliminary version. Arguments and return values are a work in progress. The ICD is primarily included to provide a functional overview of the AIS subsystem. In the NAVIS project, the software defined AIS receiver is referred to as AIS2.

The Interface Control Document (ICD) defines the interface and functionality provided by the subsystems. The AIS2 subsystem has two main states: Manual Mode or Automatic Mode. This is illustrated in Figure A.1.

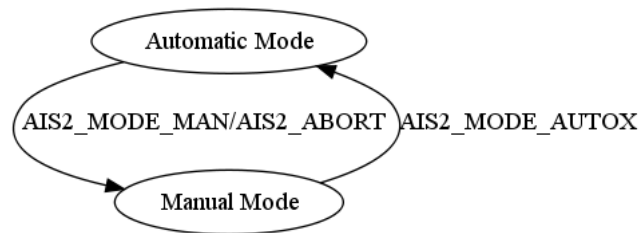


Figure A.1: AIS2 Main States

Abort Current Operation	
Purpose:	Abort the current operation and return to Manual Mode
Command:	AIS2_ABORT
Arguments:	None
Response:	AIS2.OK on success. If any issues arise, the system should still return to Manual Mode but return AIS2.WARNING
Comment:	Preliminary Version!

Get Current System State	
Purpose:	Get System State: Mode, Memory Usage, Temperature, etc.
Command:	AIS2_ABORT
Arguments:	None
Response:	System State Data on success, AIS2.FAIL on failure
Comment:	Preliminary Version!

Enable Automatic Mode	
Purpose:	Set the AIS2 subsystem in Automatic Mode
Command:	AIS2_MODE_AUTO(SUB_MODE)
Arguments:	SUB_MODE (AIS2_MODE_AUTO_SAMPLE AIS2_MODE_AUTO_PTRACK AIS2_MODE_AUTO_FREQUEST AIS2_MODE_AUTO_DEMOD AIS2_MODE_AUTO_BITSYNC AIS2_MODE_AUTO_DECODE)
Response:	Selected mode on succes, AIS2_FAIL on failure
Comment:	Preliminary Version!

Enable Manual Mode	
Purpose:	Set the AIS2 subsystem in Manual Mode
Command:	AIS2_MODE_MANUAL
Arguments:	None
Response:	AIS2_OK on success, AIS2_FAIL on failure
Comment:	Preliminary Version!

Sample Data	
Purpose:	Sample Data
Command:	AIS2_SAMPLE
Arguments:	None
Response:	AIS2_MODE_OK on success, AIS2_MODE_FAIL on failure
Comment:	Preliminary Version!

Track Packets	
Purpose:	Track packets in the sampled data
Command:	AIS2_PTRACK
Arguments:	None
Response:	Number of packets found in data
Comment:	Preliminary Version! Low resolution FFT

Center Frequency Estimation	
Purpose:	Estimate the center frequency of packets in the sampled data
Command:	AIS2_FREQUEST
Arguments:	None
Response:	AIS2_MODE_OK on success, AIS2_MODE_FAIL on failure
Comment:	Preliminary Version! High resolution FFT

Demodulate Data	
Purpose:	Demodulate packets in the sampled data
Command:	AIS2_DEMOD
Arguments:	None
Response:	AIS2_MODE_OK on success, AIS2_MODE_FAIL on failure
Comment:	Preliminary Version!

Bit Synchronization	
Purpose:	Perform bit synchronization on the demodulated data
Command:	AIS2_BITSYNC
Arguments:	None
Response:	AIS2_MODE_OK on success, AIS2_MODE_FAIL on failure
Comment:	Preliminary Version!

Decode AIS packets	
Purpose:	Decode AIS packets in demodulated data
Command:	AIS2.BITSYNC
Arguments:	None
Response:	AIS2.MODE_OK on success, AIS2.MODE_FAIL on failure
Comment:	Preliminary Version!

Received Packets Stats	
Purpose:	Return stats on received packets: Number of packets, Passed/Failed CRC, Package type distribution
Command:	AIS2.RECEIVER_STATS
Arguments:	None
Response:	AIS2.MODE_OK on success, AIS2.MODE_FAIL on failure
Comment:	Preliminary Version!

List Received Packets Headers	
Purpose:	List received packets headers
Command:	AIS2.LIST_PACKETS
Arguments:	None
Response:	AIS2.MODE_OK on success, AIS2.MODE_FAIL on failure
Comment:	Preliminary Version!

Get Specific Packet Content	
Purpose:	Get specific packet content
Command:	AIS2.GET_PACKET
Arguments:	Package number
Response:	AIS2.MODE_OK on success, AIS2.MODE_FAIL on failure
Comment:	Preliminary Version!

Set Algorithm Parameters	
Purpose:	Set Algorithm Parameter
Command:	AIS2.ALGORITHM_PARAM
Arguments:	Sample length, Store Failed CRC packets, Log settings, etc
Response:	AIS2.MODE_OK on success, AIS2.MODE_FAIL on failure
Comment:	Preliminary Version!

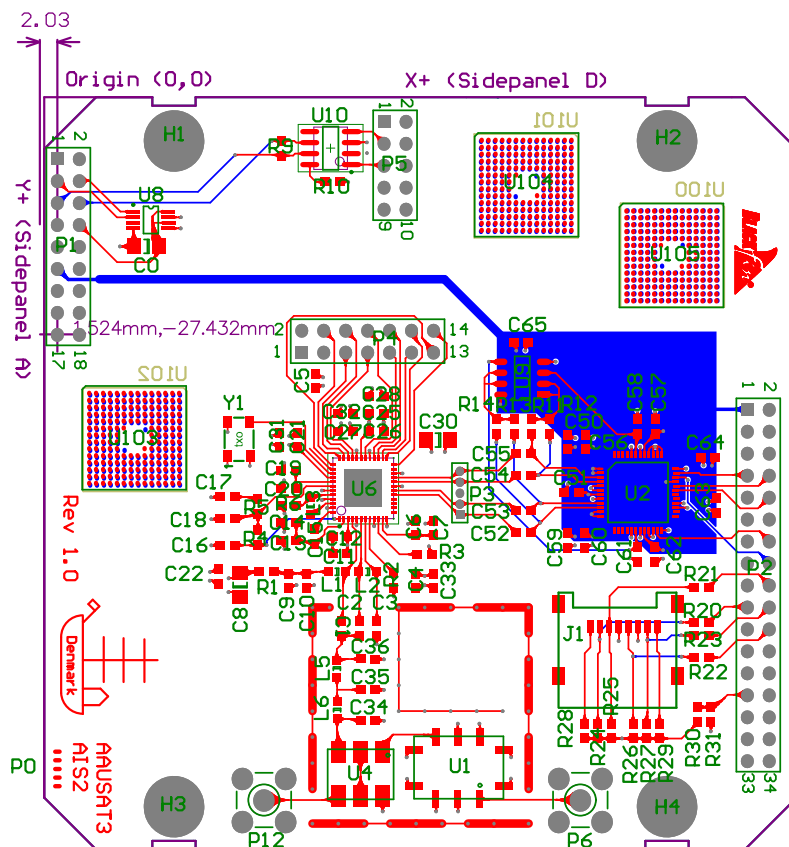
Run Test Case	
Purpose:	Run Test Case
Command:	AIS2.TEST
Arguments:	Test case number
Response:	AIS2.TEST_PASSED on success, AIS2.TEST_FAILED on failure
Comment:	Preliminary Version!

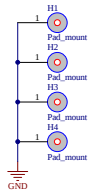
Power-On Self-Test (POST)	
Purpose:	Run Power-on Self Test
Command:	AIS2.POST
Arguments:	None
Response:	AIS2.TEST_PASSED on success, AIS2.TEST_FAILED on failure
Comment:	Preliminary Version!

Appendix B

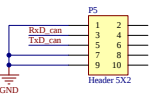
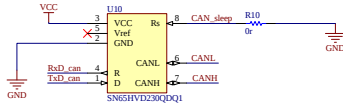
Hardware diagrams

The following two pages contain hardware diagrams and PCB layout for the software defined AIS receiver. The files are included on the attached CD, including the `hw_651.PrjPcb` Altium Designer project file.





Optional
Terminating for CAN



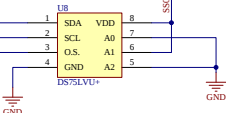
Stack connector:
CAN = communication has between subsystems.
DC = internal sensor bus.
SS = Power supply for each subsystem from EPS.

Sub-System power-pin assignment (BEXUS-08):

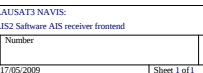
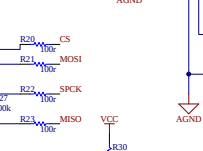
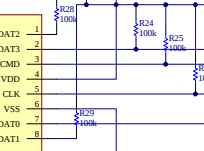
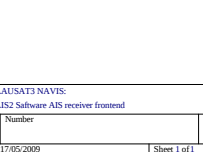
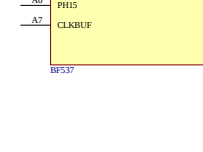
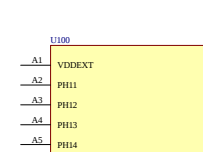
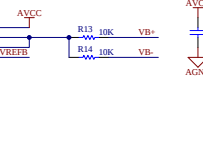
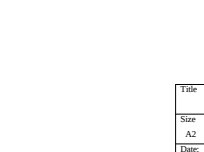
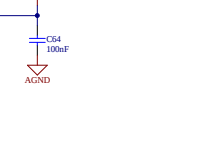
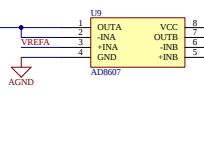
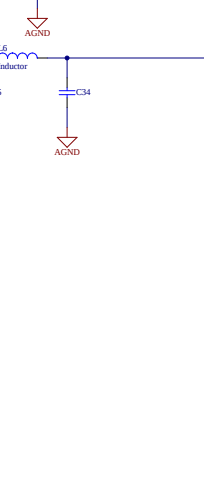
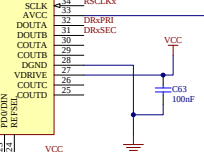
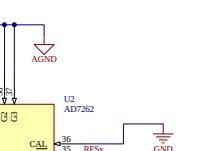
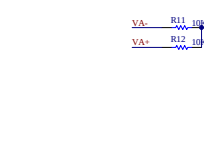
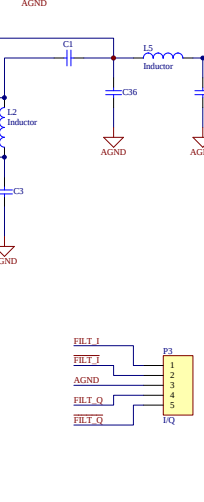
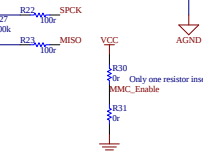
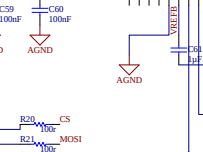
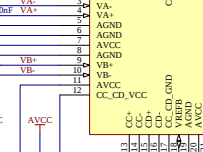
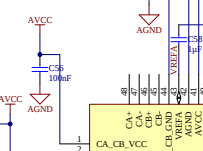
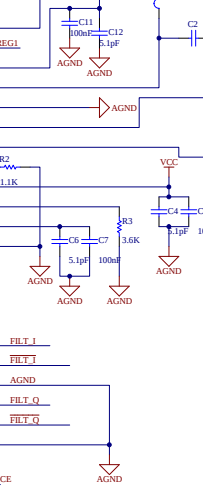
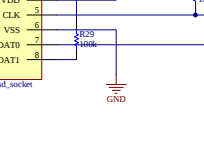
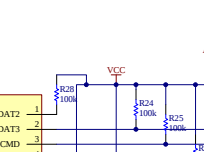
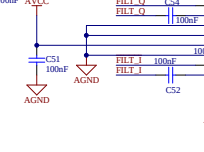
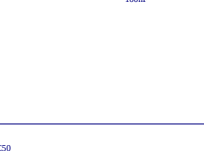
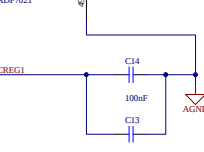
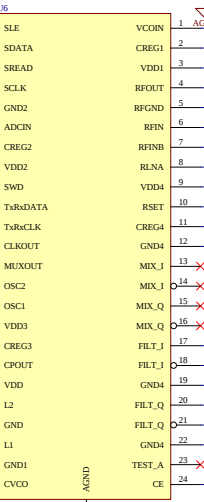
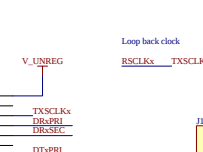
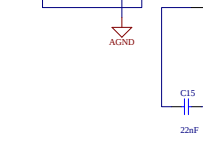
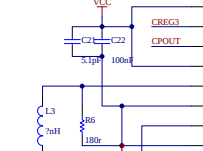
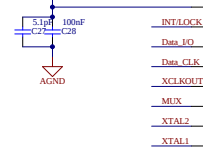
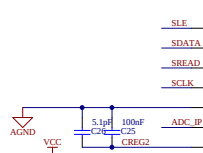
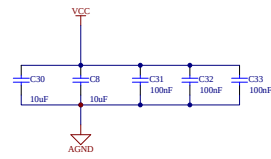
- SS0 = DC_3.3V (Vcc DC has for temp. sensor.)
- SS1 = AIS1_3.3V
- SS2 = ELA_3.3V
- SS3 = AIS2_3.3V
- SS4 =
- SS5 =
- SS6 = AIS2_5V
- SS7 =
- SS8 =
- SS9 =

Production no
P0

Temp. sensor. One for each of the following:
- Battery 1
- Battery 2
- CPU
- Radio
(TBD: different address)



North Atlantic Vessel Identification System



Title		
AAUSAT3 NAVIS		
AIS2 Software AIS receiver frontend		
Size	Number	Revision
A2		1.0
Date:	17/05/2009	Sheet 1 of 1
File:	C:\aarusat3_651_1w\SchDoc	Drawn By: 08p651

Appendix C

Component List

Here the component list is shown. To read the actual value of the component, please refer to the Altium Designer files on the CD.

Quantity	Designator	Description	Comment	Footprint
3	C0, C8, C30	Capacitor	Cap Semi	C1206
47	C1, C2, C3, C4, C5, C6, C7, C9, C10, C11, C12, C13, C14, C15, C16, C17, C18, C19, C20, C21, C22, C25, C26, C27, C28, C31, C32, C33, C34, C35, C36, C50, C51, C52, C53, C54, C55, C56, C57, C58, C59, C60, C61, C62, C63, C64, C65	Capacitor	Cap Semi	1608[0603]
4	H1, H2, H3, H4	AAUSAT3 mounting pad	Pad_mount	MountM3
1	J1	Micro_sd_socket	micro-SD-conn	
5	L1, L2, L3, L5, L6	Inductor	Inductor	2012_Chip
1	P0	Production_no	Production_no	
1	P1	Stack connector - AAUSAT3	STACK	STACK
1	P2	Header, 17-Pin, Dual row	SPORT_Blackfin	HDR2X17
1	P3	Header, 5-Pin	I/Q	MHDR1X5
1	P4	Header, 7-Pin, Dual row	Header 7X2	HDR2X7
1	P5	Header, 5-Pin, Dual row	Header 5X2	HDR2X5
2	P6, P12	SMB Straight Connector	SMB	SMB_V-RJ45
23	R1, R2, R3, R4, R5, R6, R10, R11, R12, R13, R14, R20, R21, R22, R23, R24, R25, R26, R27, R28, R29, R30, R31	Semiconductor Resistor	Res Semi	1608[0603]
1	R9	Semiconductor Resistor	Optional	1608[0603]
1	U1	SAW - 162 MHz - SIPATSAW	SAW	SAW
1	U2	ADC. 1 MSPS, 12-Bitg	AD7262	LQFP48
1	U4	2 way power splitter	ADP-2-1	ADP-2-1
1	U6	ADF7021 transceiver	ADF7021	CP-48-3
1	U8	DS75LVU+	uSOP8	
1	U9	AD8607	SOIC8	
1	U10	CAN Bus Transceiver	SN65HVD230	CAN
1	Y1	Fox CMOS cheramic oscillator	Oscillator	TXO83

Appendix D

Work agreement (Danish)

The following page shows the work agreement approval (*Godkendelse for projektsamarbejde*). In connection to this document is the 3 page description of the project and cooperation, that the study board granted the approval based on (*AIS modtager til ESA BEXUS ballon flyvning*).

Godkendelse for projektsamarbejde

– imellem 09gr650 og 09gr651

Studienævnet godkender hermed samarbejde imellem 09gr650 og 09gr651, som beskrevet i vedlagte dokument.

6/3-2009

Dato

Studienævn for Elektronik og IT
Aalborg Universitet
Fredrik Bayers Vej 7
9220 Aalborg Ø

Ove Andersen

AIS modtager til ESA BEXUS ballon flyvning

1. Baggrund og motivation

Farvandsvæsnets i Danmark har til opgave at administrere trafikken i de danske farvande og derved sørge for skibssikkerheden til vands. Dette gøres i dag med bøjer samt trafikseparation. De er godt undervejs i de danske farvande men i de grønlandske farvande, som også er Farvandsvæsnets ansvarsområde, er overvågningen af skibe mangelfuld. Derfor ønsker Farvandsvæsnets at overvåge disse skibe vha. det Automatiske IdentifikationsSystem (AIS), som det allerede foregår i de danske farvande.

AIS systemet

AIS er et civilt automatisk identifikationssystem, der gør det muligt at udveksle oplysninger mellem skibe indbyrdes og mellem skibe og landstationer. AIS er obligatorisk for alle skibe over 300 bruttotons samt alle kommercielle skibe. Oplysningerne omfatter identifikation, navigation (for eksempel position, kurs og fart) og rejserelaterede data (for eksempel destination og dybgang).

I Danmark modtages skibenes AIS signaler af 15 jordstationer, men denne metode er upraktisk ved Grønland da stationerne ikke kan modtage signaler udover line of sight. Målet med AAUSAT 3 projektet er derfor at evaluere muligheden for at modtage AIS signaler fra skibe i de Grønlandske farvande via satellit med henblik på senere udvikling af operative AIS satellitter.

Projektmål

Formålet med dette projekt er at udvikle udstyr til modtagelse af AIS signaler på ESA's BEXUS ballonopsendelse, med henblik på evaluering af muligheden for at modtage AIS signaler i rummet og dermed udvikling af en AIS modtager til AAUSAT3. Projektet vil bygge på allerede udførte studier af de teoretiske muligheder for AIS modtagelse i rummet.

ESA BEXUS – eksperimentel ballonflyvning

BEXUS står for "Balloon-borne Experiments for University Students", og er et tilbud fra en sammenslutning af følgende organisationer:

- ESA (The European Space Agency)
- SNSB (Swedish National Space Board)
- SSC (Swedish Space Corporation)
- DLR (Deutsches Zentrum für Luft- und Raumfahrt)

Projektets formål er at give universitetsstuderende mulighed for at udføre forsøg under en ballonflyvning. Flyvningen finder sted i oktober 2009 fra Kiruna i nordsverige, og vil typisk vare 5 timer og nå en højde på 35 km. Under flyvningen er formålet med AIS-eksperimentet, at afprøve to forskellige prototyper af AIS-modtagere i realistiske omgivelser, inden det endelige design til AAUSAT3 fastlægges.

AIS eksperimentet er pr. 10. februar blevet udvalgt af ESA til flyvningen i oktober 2009.

2. Projektbeskrivelser

Gruppe 650

Projektet vil tage udgangspunkt i udviklingen af et system til modtagelse af AIS data pakker fra skibe, eventuelt analysere disse, og sende disse videre til senere behandling. Systemet vil blive monteret på en ballon der opsendes til ca. 35km højde i samarbejde med ESA. Systemet vil bestå af en COTS radiomodtager, samt en mikrocontroller til dekodning af AIS signalerne. Dette signal vil blive analyseret og behandlet, således det kan bruges til sammenligning med referencesignaler fra de jordbaserede AIS modtagestationer.

Projektet omfatter således følgende:

Analyse af AIS standarden*

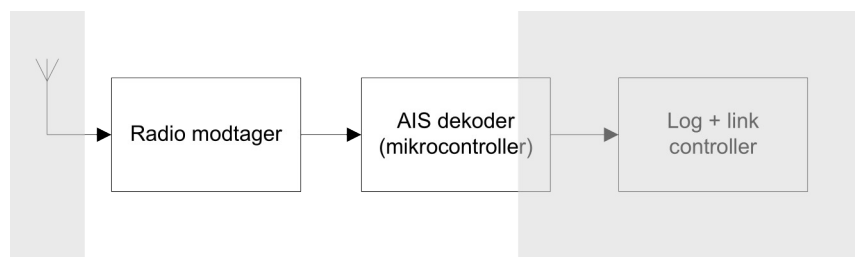
Analyse af GMSK modulationstypen*

Link budget for modtagelse af AIS signaler fra BEXUS ballonen*

Implementering af COTS** radiomodtager

Udvikling og implementering dekoder til AIS protokollen

Målet med projektet er således at have en fungerende AIS modtager prototype.



**COTS: Commercial Of The Shelf

Gruppe 651

Projektet vil tage udgangspunkt i udviklingen af et system til modtagelse af AIS datapakker fra skibe. Systemet skal implementeres i en digital radiomodtager. Modtageren vil blive monteret på en ballon der opsendes til ca. 35 km højde i samarbejde med ESA.

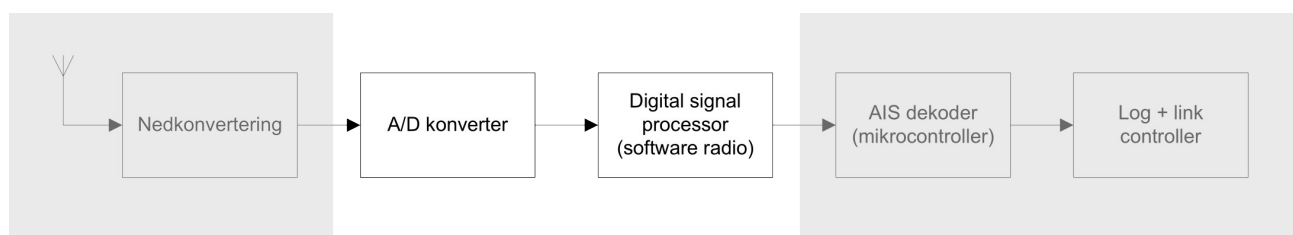
Projektet omfatter således følgende:

Analyse af AIS standarden*

Analyse af GMSK modulationstypen*

Link budget for modtagelse af AIS signaler fra BEXUS ballonen*

Implementering af softwarebaseret modtager/demodulator + RF frontend



3. Samarbejde mellem projektgrupper

To projektgrupper vil forestå udviklingen af hver deres AIS modtager, til montering på BEXUS ballonen. Således vil den ene gruppe fokusere på en COTS RF modtager og demodulator mens den anden gruppe vil fokusere på en softwarebaseret modtager. Dette indbefatter en ADC og en DSP, samt de tilhørende algoritmer og implementering.

Grundet projektets omfang vil det være nødvendigt for de to grupper at samarbejde om flere af emnerne i projekterne. Dette falder naturligt, da begge grupper arbejder på at løse samme problem, på to forskellige måder. Målet med samarbejdet er at nå længere, end hvis hver gruppe arbejdede isoleret.

De emner der samarbejdes om, er følgende:

- Indledning til AAUSAT3-projektet
- Analyse af AIS standarden
- Beskrivelse af BEXUS projektet
- Linkbudget for modtagelse af AIS signaler fra BEXUS ballonen

De to gruppers rapporter bliver individuelle, hvad angår design, implementering, test og konklusion (alt teknisk indhold), mens introduktioner og analyser vil være fælles beskrevet.

Det vil af rapporterne fremgå hvilke grupper der har samarbejdet, samt hvilke afsnit i rapporterne der er fælles udarbejdet. Gruppemedlemmerne vil således, som normalt, underskrive deres respektive gruppers rapport og dermed stå inde for såvel materiale udarbejdet af gruppen som fælles udarbejdet materiale.

Contents of the CD

The attached CD contains the report, all MATLAB code developed in the project and data sheets from all major components.

```
+report
- report.pdf
- CDR_SED.pdf
- CDR_Appendix.pdf
+matlab
- ADFoutput.m
- BitGen.m
- bitrev.m
- BitsToPhase.m
- Demodulate.m
- DetermineFrequency.m
- Extract.m
- FindPacket.m
- FindPackets.m
- GMSKmodulator.m
- Main.m
- NRZI.m
- PhaseToFrequency.m
- ReadFromDAQ.m
- ReadFromFile.m
- SaveToFile.m
- WaterfallPlot.m
+datasheets
- All datasheets used in project
+sources
- All digital sources
```

Abbreviations

ADC	Analog to Digital Converter
ADF	ADF7020 Radio transceiver
AGC	Automatic Gain Control
AIS	Automatic Identification System
AWGN	Additive White Gaussian Noise
BER	Bit Error Rate
BPF	Band Pass Filter
CAN	Controller Area Network
CDR	Critical Design Review
COTS	Commercial off-the-shelf
CSDR	Center for Software Defined Radio
CSP	CAN Space Protocol
DaMSA	Danish Maritime Safety Administration
DLR	Deutsches Zentrum für Luft- und Raumfahrt
DSP	Digital Signal Processor
ESA	European Space Agency
Esrangle	European Sounding Rocket Launching Range
ESTEC	European Space Research and Technology Centre, ESA
ESW	Experiment Selection Workshop
FSK	Frequency Shift Keying
GMSK	Gaussian Minimum Shift Keying
LNA	Low-Noise Amplifier
NMEA	National Marine Electronics Association
PCB	Printed Circuit Board
PDR	Preliminary Design Review
SNSB	Swedish National Space Board
SAW	Surface Acoustic Wave
SED	Student Experiment Documentation
SNR	Signal to Noise Ratio
SSC	Swedish Space Corporation (Eurolaunch)
STW	Student Training Week
TBD	To be determined
UHF	Ultra High Frequency
VHF	Very High Frequency

Bibliography

- [07gr506 07] Aalborg University 07gr506. *AIS Receiver for AAUSAT3*. CD, 2007.
- [08gr414 08] Aalborg University 08gr414. *Platformsystem til AAUSAT3*. CD, 2008.
- [09gr650 09] Aalborg University 09gr650. *AAUSAT3 AIS1 receiver*. CD, 2009.
- [Analog Devices 05] Analog Devices. *ADF7020-1: High Performance FSK/ASK Transceiver IC*. http://www.analog.com/static/imported-files/data_sheets/ADF7020-1.pdf, 2005.
- [Analog Devices 07a] Analog Devices. *ADF7021: High Performance Narrowband ISM Transceiver IC*. http://www.analog.com/static/imported-files/data_sheets/ADF7021.pdf, 2007.
- [Analog Devices 07b] Analog Devices. *Evaluation Boards for ADF7020 and ADF7020-1 ISM Band Transceiver*. CD:ADF7020-XDBXEvalNote.pdf, 2007.
- [Analog Devices 08] Analog Devices. *AD7262: 1 MSPS, 12-Bit, Simultaneous Sampling SAR ADC with PGA and Four Comparators*. http://www.analog.com/static/imported-files/data_sheets/AD7262.pdf, 2008.
- [Analog Devices 09a] Analog Devices. *ADSP-BF534/ADSP-BF536/ADSP-BF537 Datasheet*. http://www.analog.com/static/imported-files/data_sheets/ADSP-BF534_BF536_BF537.pdf, 2009.
- [Analog Devices 09b] Analog Devices. *ADSP-BF537 Blackfin Processor Hardware Reference*. http://www.analog.com/static/imported-files/processor_manuals/bf537_hwr_Rev3.2.pdf, 2009.
- [Armstrong 07] Keith Armstrong. Emc for printed circuit boards. ISBN 978-0-9555118-0-6, 2007.
- [Beagleboard 09] Beagleboard. *The Beagleboard Project*. <http://www.beagleboard.org>, 2009.
- [Blackfin Koop 09] Blackfin Koop. *The Blackfin Open Source Koop*. <http://blackfin.uclinux.org>, 2009.
- [Cervera 08] Miguel A. Cervera & Alberto Ginesi. *On the Performance Analysis of a Satellite-based AIS System*. 2008.
- [Dahl 06] Ole Fredrik Haakonsen Dahl. *Space-Based AIS Receiver for Maritime Traffic Monitoring Using Interference Cancellation*. <http://ntnu.diva-portal.org/smash/record.jsf?pid=diva2:121772>, 2006.

- [dsprelated.com 07] dsprelated.com. *Automatic Identification System CRC-CCITT*. <http://www.dsprelated.com/showmessage/79522/1.php>, 2007.
- [Fleury 07] Bernard H. Fleury & Ingmar Land. *Digital Modulation 1 - Lecture Notes*. <http://kom.aau.dk/project/navcom/CourseWebSites/DigitalModulation1/notes.pdf>, 2007.
- [Haykin 01] Simon Haykin. *Communication systems*. John Wiley and Sons, Inc., 2001.
- [IEEE 07] IEEE. *Information for IEEE Transaction Journals, and Letters Authors*. http://www.ieee.org/portal/cms_docs_iportals/iportals/publications/authors/transjnl/auinfo07.pdf, 2007.
- [ITU 07] International Telecommunication Union ITU. *ITU-R M.1371-3: Technical characteristics for an automatic identification system using time division multiple access in the VHF maritime mobile band*. <http://www.itu.int/rec/R-REC-M.1371/en>, 2007.
- [MX Com 95] Inc. MX Com. *Practical GMSK Data Transmission*. http://sss-mag.com/pdf/gmsk_tut.pdf, 1995.
- [Serway 04] Raymon A. Serway & Jr. John W. Jewett. *Physics for scientists and engineers*. Thomson Brooks/Cole, 2004.
- [SIPAT Co. 09] Ltd. SIPAT Co. *162.2 MHz SAW Part Number: LBT16201*. <http://www.sipatsaw.com/datasheet/RFFILTER/LBT16201.pdf>, 2009.
- [SSC 09] SSC & DLR. *Bexus user manual*. http://rexusbexus.net/index.php?option=com_content&view=article&id=51&Itemid=63, 2009.
- [System Standex A/S 09] System Standex A/S. *SE samleled, med 3 ben*. <http://www.system-standex.dk/aluminiumbyggesystemer/konstruktionssystem/samleled/se-samleled/se20/25/30-3/>, 2009.
- [Vasudevan 07] Kasturi Vasudevan. *Digital communications and signal processing*. ISBN 9781420068115, 2007.