UNIVERSITY OF TWENTE.

Faculty of Electrical Engineering,
Mathematics & Computer Science

Design of a Testbed for the Calibration and RFI Mitigation Algorithms used in OLFAR

M. F. Brethouwer

M.Sc. Thesis
February 2015

Supervisors:
P. K. A. van Vugt M.Sc.
Dr. ir. M. J. Bentum
Dr. ir. A. Meijerink
Prof. dr. ir. ing. F. B. J. Leferink
Dr. ir. A. J. Annema

Telecommunication Engineering Group
Faculty of Electrical Engineering,
Mathematics and Computer Science
University of Twente
P.O. Box 217
7500 AE Enschede
The Netherlands

Summary

The Universe has been observed for thousands of years. However, radio astronomy at low frequencies has only recently become a major research topic. Radio astronomy at frequencies below 30 MHz is expected to shed more light on the so called ‘Dark Ages’. This is a period in the history of the Universe when it was opaque for visible light. A new type of radio telescope will be built in space to perform these observations. This telescope will be called OLFAR (Orbiting Low Frequency Antennas for Radio Astronomy), and will consist of tens of small satellites which will be working together. This telescope cannot be built on Earth, because the ionosphere will distort or block these low-frequency signals. In addition, a lot of man-made interference is present on earth.

For OLFAR, new software algorithms for calibration and interference mitigation will be developed. The verification of these algorithms requires a testbed which imitates the hardware of OLFAR. The design of this testbed is presented in this thesis. The testbed is split into five components: an astronomical source simulator, the observational antennas, the receivers, the required software for operation of the testbed, and the physical construction. For the first four components, the specifications are presented and the design is developed and verified. For the physical construction of the testbed, some useful insights for future work are presented.

The designed testbed is flexible and can easily be reconfigured to the needs of the measurements. Only a slight modification to the firmware of the receivers still has to be incorporated, whereupon the testbed will satisfy all requirements to adequately imitate the hardware of OLFAR. Incorporating this modification and fabricating the testbed is left for future work.

Contents

List of abbreviations

COTS	commercial off-the-shelf
DAC	digital-to-analog converter
DCIS	Data Collection Colonies in Space
EMC	electromagnetic compatibility
FPGA	field-programmable gate array
LOFAR	Low-Frequency Array
MIMO	multiple-input and multiple-output
OLFAR	Orbiting Low Frequency Antennas for Radio Astronomy
PCB	printed circuit board
PLL	phase-locked loop
RFI	radio frequency interference
SDR	software defined radio
SMD	surface-mounted device
SPI	serial peripheral interface
VCTCXO	voltage controlled temperature compensated crystal oscillator
VHDL	VHSIC hardware description language
VSWR	voltage standing wave ratio

List of symbols

Symbol	Description
$\mathbf{0}$	Zero matrix
a_i	Complex amplitude of which the imaginary part denotes the phase of an incoming plane wave for antenna i
$\mathbf{a}_i$	Vector containing complex amplitudes of which the imaginary parts denote the phases of the incoming plane wave from source i for all antennas
$\mathbf{A}$	Matrix containing complex amplitudes of which the imaginary parts denote the phases of all incoming signals for all antennas
B	Observing bandwidth
D_{max}	Maximum baseline distance
D_{source}	Distance between the source and the receivers
g_i	The voltage gain of receiver i
g_{0i}	Direction and polarization dependent gain for an incoming plane wave for antenna i
$\mathbf{g}_{0i}$	Vector containing the direction and polarization dependent gains for the incoming plane wave from source i for all antennas
$\mathbf{G}$	Matrix containing all receiver voltage gains
$\mathbf{G}_0$	Matrix containing all direction and polarization dependent gains for all incoming signals for all antennas
$\mathbf{I}$	Identity matrix
k_{b}	Boltzmann constant
P	Number of antennas
Q	Number of sources
$\mathbf{Q}$	Matrix containing the mutual coupling coefficients between all antennas
$r_{\mathrm{n},ij}$	One element of the noise covariance matrix
$\mathbf{R}_{\mathrm{n}}$	Noise covariance matrix
$\mathbf{R}_{\mathrm{s}}$	Array covariance matrix containing the expected value of the visibilities
$\hat{\mathbf{R}}_{\mathrm{s}}$	Array covariance matrix containing the measured value of the visibilities
$T_{\mathrm{eq.input}}$	Equivalent input temperature
T_{sky}	Sky noise temperature
θ_{b}	Angular resolution in radians
λ	Wavelength
σ_i	Power of celestial source i
σ_{ref}	Power of the reference transmitter
$\mathbf{\Sigma}$	Matrix containing all celestial source powers
$\mathbf{\Psi}$	Element pattern overlap matrix

Chapter 1
Introduction

This thesis is the result of the work performed for a master's assignment in Electrical Engineering at the Telecommunication Engineering group of the University of Twente. This document provides the reader with all relevant information and the results of the work that has been performed. To familiarize the reader with the subject and the assignment, this thesis will start with some background information.

1.1 Context

The Universe has been observed by astronomers for thousands of years. This has been done with numerous different types of instruments. However, almost all of these instruments were designed to receive signals with frequencies starting from hundreds of megahertz. Only recently, radio astronomy at lower frequencies has become a major research topic. Radio astronomy at low frequencies is expected to provide new knowledge in many research areas in astronomy. Especially in the field of cosmology it is expected to provide invaluable insights. It will provide information about the so-called 'Dark Ages', which is the period between 380 thousand and 400 million years after the Big Bang, during which the Universe was opaque for visible light. Other usage for radio astronomy at low frequencies are complementing the current (extra)galactic surveys with a new frequency band, and observing space weather such as the solar wind.

The desired low frequency signals cannot be observed in high resolution by Earth-bound observatories. This due to three factors: At very low frequencies the ionosphere is non-translucent and will prevent the signals from space from reaching the Earth surface. Secondly, at somewhat higher frequencies, the ionosphere is unstable and will fluctuate rapidly. This will distort the signals from space just like the surface of water does with sunlight. And last, at these low frequencies many powerful sources of radio frequency interference (RFI) are present, such as radio broadcasting, aerospace communication and lightning.

A relatively new project named OLFAR (Orbiting Low Frequency Antennas for Radio Astronomy) will avoid the problems caused by the ionosphere and RFI by creating a radio telescope in space [1, 2]. The OLFAR telescope will consist of a swarm of tens of satellites in a formation up to 100 kilometer across. Each satellite will contain a low frequency receiver to measure the astronomical signals, these signals will be combined and correlated in space to reduce the amount of data that has to be transported to Earth. This process is called 'interferometry'. For this purpose, each satellite is fitted with an inter-satellite communication system for the distribution of the measurement data and a satellite-Earth communication system to transmit the correlated data back to Earth.

Just like Earth-bound radio telescopes, OLFAR needs to be calibrated to be able to perform its task. This calibration should at least include the antenna patterns of each antenna, the gains of all receivers, and the phase differences between all receivers. However, the design of OLFAR makes the calibration a much more difficult task than for traditional interferometers. The satellites of OLFAR will be constantly moving and rotating, and the antenna positioning will be imperfect due to the geometry of the satellites. Another issue is the omnidirectional antenna pattern of the satellites which results in many more possible signal sources in the field of view. Also, any necessary real-time calibration routines would have to be performed with limited computational power. This is caused by the limited amount of electrical power that will be available on board of the satellites.

Algorithms for the calibration and RFI mitigation are being developed in the Data Collection Colonies in Space (DCIS) research project. These algorithms are now tested in simulation, but it is very desirable to be able to verify the functionality of the algorithms with actual captured data. This data should be gathered by a testbed which imitates the functionality of OLFAR and an astronomical source, because no actual OLFAR-hardware is yet available.

1.2 Testbed outline

Creating a testbed is necessary to be able to verify the functionality of the calibration and RFI mitigation algorithms. Simulation alone is not adequate as it uses an idealized version of the system and the relevant parameters. The results of the simulation will thus be based only on foreseen parameters and will show an idealized version of the real-world behavior. To include all unforeseen parameters a testbed should be used which can provide a realistic representation of OLFAR.

The testbed is envisioned to consist of a number of miniature satellite mock-ups with functional antennas and receivers. These will be placed outdoors in a large empty area, like a pasture. An astronomical source simulator will be placed some distance away, so that it can be seen as a relatively small source. The testbed presumably cannot be placed indoors due to the large distance which is necessary between the satellite mock-ups and the source simulator. To mimic the movement of the satellites of OLFAR compared to the astronomical source the satellite mock-ups and the source simulator should be able to move relative to each other.

During operation, it is envisioned that the astronomical source simulator is transmitting a signal and all receivers will be receiving simultaneously for a specified period of time. The signals received by the receivers during this time frame will be stored as the measurement data for this single measurement. When the measurement is finished, the satellite mock-ups or the source simulator shall be relocated and a new measurement will be started. This process is repeated multiple times. When enough measurements have taken place, the data of all separate measurements shall be used as input data for the same algorithms as those that are used for the simulations.

1.3 Research objective

The goal of this master's assignment is to design, construct and demonstrate the envisioned testbed which will capture data for the calibration and RFI mitigation algorithms used in OLFAR. The testbed should mimic the behavior of the satellites in terms of properties, movements and inaccuracies. It should also be able to emulate all relevant parameters which are of influence on OLFAR and the emulation should be as realistic as possible. The testbed will consist of multiple miniature satellite mock-ups, an astronomical source simulator and all software necessary for operation. It shall also have a comprehensive manual. All these goals together result in the following research question:

Research question:
How to create a realistic testbed for the calibration and RFI mitigation algorithms used in the interferometer OLFAR with its arbitrary three-dimensional antenna distribution?

1.4 Thesis structure

After this introduction, the thesis continues with some essential theory in Chapter 2. An overview of the design of the testbed is provided in Chapter 3, after which the five main components of the testbed shall be elaborated in separate chapters. Chapter 4 provides the details for the astronomical source simulator, and Chapter 5 presents the design and implementation of the observational antenna system. The receivers of the testbed are described in Chapter 6. The details of the software used to control the testbed and the software used for data processing are presented in Chapter 7. Chapter 8 provides the research that has been performed for the physical construction of the testbed. The thesis ends with Chapter 9, where the results of the design and recommendations for future work are presented.

Chapter 2

Theoretical background

As an aid to the material described in the main part of this thesis, some theory will be provided in this chapter. For this work only limited knowledge of astronomy is required. This information is presumed known to the reader, but can otherwise be found in many well-known textbooks [3]. Additionally, a more advanced knowledge of the concept of interferometry and the associated calculations is essential. This knowledge will be provided in this chapter, together with a description of some methods of calibration and RFI mitigation.

2.1 Interferometry

The main reason to use interferometry is to create a radio telescope with a higher resolution than would otherwise be possible or practical to create. The resolution of a radio telescope is essentially determined by the wavelength and the size of the antenna. At a certain point it becomes difficult or impossible to increase the resolution further by building an even larger antenna. This problem can be solved by using interferometry. With interferometry, many smaller antennas are used to synthesize an antenna with a larger size. The largest distance between two of those smaller antennas is called the maximum baseline distance. This distance determines the resolution of the telescope instead of the size of the antennas. Thus, the resolution of an interferometer can be increased by placing the receivers further apart.

2.1.1 Principle of interferometry

Interferometry uses the correlation between signals received by multiple antennas. In this work the source shall always be placed far away from the receivers such that the incoming rays from the source to all antennas can be assumed parallel. Also, the bandwidth of the system shall be small compared to the center frequency and the system can therefore be assumed narrow-band. By using these two criteria, the transmitted signal can be represented by a quasi-monochromatic plane wave [4]. By using this plane wave representation the main part of the interferometer can be schematically drawn as in Figure 2.1.

The incoming plane wave from the source is received by all antennas of the interferometer, but with different time delays for each antenna. Because of the used narrow-band criterion, these delays can be described fully by using only phase differences between the output signals of the antennas. The output signals of the antennas are sent to the correlator via cables, amplifiers and other electronics. These impose a complex gain which is shown as g_i in the figure. The correlator calculates the array covariance matrix $\hat{\mathbf{R}}_\mathrm{s}$ by correlating all received signals with each other. This array covariance matrix contains the measured value of what is called the visibilities [5].

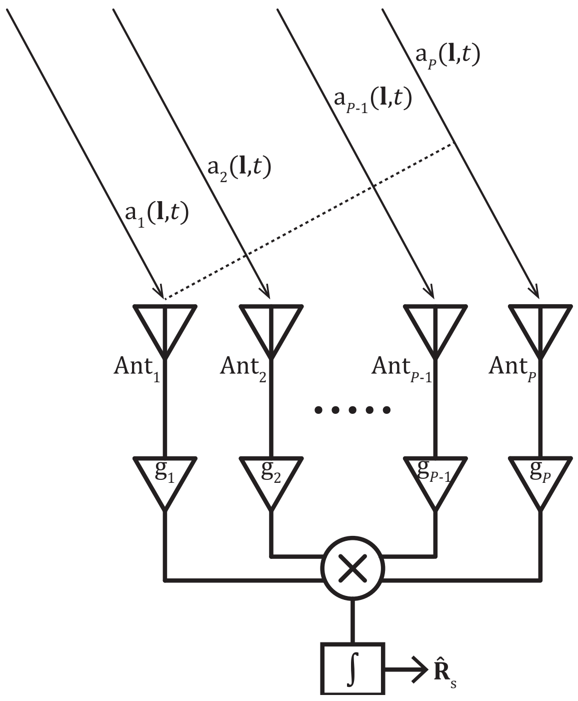


Figure 2.1: Diagram of an interferometer

2.1.2 Basic calculations

As stated before, the main purpose of using interferometry is to increase the resolution of a telescope. This parameter is also important for the testbed, as the resolution should be high enough to illustrate the performance of the algorithms. The resolution of a telescope is often expressed as an angular resolution. The angular resolution of a telescope is the smallest angular size of a feature that can still be discerned in the final output image. This terminology is slightly counter intuitive because to increase the resolution of a telescope its angular resolution must be decreased. A very simple equation can be used to approximately calculate this angle [5]

$$\theta_\mathrm{b} \approx \frac{\lambda}{D_\mathrm{max}}. \tag{2.1}$$

This formula indicates that the angular resolution θ_b is approximately equal to the wavelength λ divided by the maximum baseline distance D_max. For the design of the testbed another simple formula is of great importance. As stated before, the source of the signals should be placed far from the receivers. This ensures that the incoming rays from the source to all antennas can be assumed parallel. This is necessary for the used plane wave assumption. The minimum distance which is required between the source and the testbed D_source, can also be calculated based on the wavelength and the maximum baseline distance. The relation is provided by [5]

$$D_\mathrm{source} \gg \frac{D^2_\mathrm{max}}{\lambda}. \tag{2.2}$$

2.1.3 Data model

To be able to determine which parameters influence the output of the interferometer a data model has to be created. The model used in this work is a widely used model and is described in multiple publications [6, 7]. The output of an interferometer is the array covariance matrix containing the measured value of the visibilities ($\hat{\mathbf{R}}_\mathrm{s}$). In the model, this matrix is estimated by the array covariance matrix containing the expected value of the visibilities ($\mathbf{R}_\mathrm{s}$). For an interferometer consisting of P antennas which is receiving signals from Q sources, this matrix can be calculated by

$$\mathbf{R}_\mathrm{s} = \mathbf{G}\mathbf{Q}(\mathbf{A} \odot \mathbf{G}_0)\mathbf{\Sigma}(\mathbf{A} \odot \mathbf{G}_0)^H\mathbf{Q}^H\mathbf{G}^H + \mathbf{R}_\mathrm{n}. \tag{2.3}$$

This formula models the relation between the $(P{\times}P)$ array covariance matrix $\mathbf{R}_\mathrm{s}$, as the output of the interferometer, and the matrices $\mathbf{\Sigma}$ and $\mathbf{A}$ as the input. The $(Q{\times}Q)$ matrix $\mathbf{\Sigma} = \mathrm{diag}([\sigma_1, \sigma_2, \cdots, \sigma_Q]^\mathrm{T})$ contains all celestial source powers. These source powers are multiplied by the $(P{\times}Q)$ matrix $\mathbf{A} = [\mathbf{a}_1, \mathbf{a}_2, \cdots, \mathbf{a}_Q]$, $\mathbf{a} = [a_1, a_2, \cdots, a_P]^\mathrm{T}$, containing the phasors which represent the phases of all incoming signals for all antennas, and the $(P{\times}Q)$ matrix $\mathbf{G}_0 = [\mathbf{g}_{01}, \mathbf{g}_{02}, \cdots, \mathbf{g}_{0Q}], \mathbf{g}_0 = [g_{01}, g_{02}, \cdots, g_{0P}]^\mathrm{T}$, which describes the direction and polarization dependent gains for all incoming signals for all antennas. The gains of the receivers are contained in the $(P{\times}P)$ matrix $\mathbf{G} = \mathrm{diag}([g_1, g_2, \cdots, g_P]^\mathrm{T})$ and the mutual coupling between the antennas is described by the $(P{\times}P)$ matrix $\mathbf{Q}$.

The ever-present noise at the output of the interferometer can be modeled by the $(P{\times}P)$ noise covariance matrix $\mathbf{R}_\mathrm{n}$, this is given by

$$\mathbf{R}_\mathrm{n} = k_\mathrm{b}T_\mathrm{sky}B\mathbf{\Psi} + k_bT_\mathrm{eq.input}B\mathbf{I}. \tag{2.4}$$

The noise covariance matrix is determined by the Boltzmann constant k_b and the observing bandwidth B. The sky noise temperature T_sky together with the $(P{\times}P)$ element pattern overlap matrix $\mathbf{\Psi}$ describes the sky noise. The added noise by the receiver systems is described by the equivalent input temperature $T_\mathrm{eq.input}$ together with the $(P{\times}P)$ identity matrix $\mathbf{I}$.

2.2 Calibration

Formula 2.3 describes the full transfer function of an interferometer. The formula describes the conversion from the source powers $[\sigma_1, \sigma_2, \cdots, \sigma_Q]$ and the positions described by $[\mathbf{a}_1, \mathbf{a}_2, \cdots, \mathbf{a}_Q]$, to the output array covariance matrix $\mathbf{R}_\mathrm{s}$. The goal of radio astronomy is to measure these source powers and positions as accurately as possible. To accomplish this goal, all other parameters in Formula 2.3 should be known as accurately as possible.

There are many methods in which the parameters of Formula 2.3 can be determined. The decision to use a certain method for a specific parameter is mostly based on the dependance of that parameter to external factors like time and temperature. Some parameters are not influenced by external factors. These can be measured once and will remain constant during the measurement, or even the life-span of the instrument. Others are varying largely and need to be measured continuously. There are also parameters which can be simulated and others need to be calculated.

The calibration methods of OLFAR will differ from the calibration methods of traditional interferometers. These differences are partly due to the construction and arbitrary three-dimensional antenna distribution of the interferometer. Some of these differences also hold for the LOFAR (Low-Frequency Array) radio telescope [8]. The calibration algorithms of LOFAR implement different methods for calibration dependent on the specific parameter [9]. The receiver gains $[g_1, g_2, \cdots, g_P]$ and the observing bandwidth B are measured in advance and are assumed known during the astronomic measurements. The direction and polarization dependent gains $[\mathbf{g}_{01}, \mathbf{g}_{02}, \cdots, \mathbf{g}_{0Q}]$ and the mutual coupling matrix $\mathbf{Q}$ are simulated for certain directions and will be interpolated for the directions of all sources. An initial guess of the phasors which contain all phases of the incoming plane waves $[\mathbf{a}_1, \mathbf{a}_2, \cdots, \mathbf{a}_Q]$ is based on earlier measurements of the source locations. Still, none of these determined parameters will be determined with perfect precision. To increase the accuracy of the telescope, calibration is necessary.

An overview of possible calibration algorithms designed for LOFAR, which are of interest for OLFAR, has already been established in a paper [10]. Especially the station calibration section is of interest for this thesis. A possible method of calibrating a telescope stated in this paper is to assume that no noise is present ($\mathbf{R}_\mathrm{n} = \mathbf{0}$), and to use celestial sources with known locations and powers as signal sources for calibration. Another approach is an improvement on the first by using only off-diagonal elements of $\mathbf{R}_\mathrm{s}$ for calibration. These elements are less affected by $\mathbf{R}_\mathrm{n}$ because they do not include the system noise. The last method stated in the paper is to use a reference transmitter in a known direction with known and sufficient power such that the noise can be neglected. Thus, the power of the reference transmitter must be much larger than any of the elements in the noise covariance matrix ($\sigma_\mathrm{ref} \gg \max r_{\mathrm{n},ij}$).

Three algorithms are devised specifically to overcome the problems with the calibration of LOFAR [8]. Of these algorithms, the third one is very interesting for OLFAR. This algorithm is dubbed 'peeling' and it vastly reduces the amount of processing needed by calibrating for just one source at a time. The assumption is made that the most powerful source is the only visible source. This source will be used for the calibration in that specific direction. The contribution of the source is then subtracted from the measured data and the algorithm repeats for the next most powerful source. The algorithm will continue for the first couple of thousand most powerful sources.

2.3 RFI Mitigation

RFI is one of the main reasons for building an interferometer in space. At the low frequencies where OLFAR will be measuring, a large number of powerful interference sources are present. RFI mitigation is therefore a important topic for OLFAR.

2.3.1 Methods for RFI mitigation

Five methods for RFI mitigation will be summarized below [11]. These methods are used in general and where not specifically designed for OLFAR.

Thresholding
Thresholding is performed by measuring the output power of each antenna using a high resolution in time and frequency. If the output power of the antenna is much larger than the mean output power, the data is deemed contaminated by RFI. The measurement data for that specific antenna, time and frequency slot will be discarded.

Filtering
In this method, the RFI waveform received by an antenna is estimated in real-time using a filtering technique. This estimated signal will be subtracted from the original contam-

inated output signal to generate a clean output signal. This clean output signal of the antenna will be used as input for the correlator.

Reference channel
For this method, a separate antenna is used which does not receive the source signal but does receive the RFI signal. This reference channel will however not contain exactly the same RFI signal as the measurement channels. An adaptive filtering technique will be used to estimate the RFI waveform present in the measurement channels from the RFI waveform received by the reference antenna. The result of the estimations will be subtracted from the contaminated signals to produce clean output signals.

Spatial filtering
For spatial filtering, the beam pattern of the interferometer will be adapted by shifting the phases of all input signals of the correlator. By adapting the beam pattern a null can be directed to the source of the RFI which will attenuate the signals from that specific direction.

Probability distribution analysis
By measuring the output of an antenna in real-time, the difference in the probability distributions of the source signals and RFI signals can be used to detect RFI signals. The astronomical signals will generally have a Gaussian probability distribution with zero mean and their power spectrum will have an exponential distribution [11]. The presence of an RFI signal will change these statistics and therefore the presence of an RFI signal can be detected and the measurement data can be discarded.

2.3.2 Applicability to OLFAR

The stated RFI mitigation methods impose some specifications on the design of OLFAR. For all methods the signals from each antenna should be sampled with a high resolution in time and frequency. The filtering and reference channel methods require adequate real-time processing power to be able to calculate the filters. The probability distribution analysis method requires even more processing power to be able to calculate the statistics of the signals in real-time. Lastly, the reference channel method requires an additional antenna which should receive the RFI signal without receiving the source signal.

The OLFAR design does not contain the additional reference antenna required by the reference channel method [1, 2, 12]. The design also has limited real-time processing power, presumably too little power for the probability distribution analysis method. The only viable methods of RFI mitigation are therefore: thresholding, filtering, and spatial filtering. The requirements imposed on the design of OLFAR by these methods are; sampling the signals with high resolution in time and frequency, and the availability of real-time processing power. These requirements are viable options to be incorporated in the design of OLFAR and the testbed should be designed to incorporate these.

Chapter 3

Testbed overview

In the previous chapters, the reader has attained the some background information and has acquired the necessary theory to comprehend the design of the testbed. In this chapter, the specifications of the system will be listed as the basis for the design of the testbed. The used design approach will be looked into and a functional design shall be presented. The components of the functional design will be described in more detail in later chapters.

3.1 Specifications and boundary conditions

The design of a system starts with determining the specifications and boundary conditions. For this assignment, the specifications are based on those of OLFAR and the boundary conditions have been devised to make the testbed usable. Three simple boundary conditions have been conceived at the start of the assignment. These will assure that the testbed will be practical for its intended use. The first states that the testbed should be portable; it should be possible to transport the testbed by car or with a trailer. Secondly, the testbed should be affordable; as the assignment includes the construction of the proposed testbed, the components costs should remain within reasonable budget. And thirdly, using the testbed should be legal; for example, the signals should be transmitted in allowed frequency bands.

The specifications of the system are based on the specifications of OLFAR. However, it is very important to note that the testbed is not required to have exactly the same specifications as OLFAR. The testbed only has to represent these specifications. For example, the frequency range of the testbed can be scaled with a certain factor, compared to that of OLFAR, as long as all other frequency dependent specifications (like the baseline distances and accuracies) are scaled with the same factor. The specifications of the receiver are another example; the testbed does not need to contain the same receivers as those that will be used for OLFAR, but the behavior and output of the receivers should be able to represent them. A summary of the specifications of OLFAR, which are relevant for the testbed, is provided in Appendix A.

3.2 Design approach

The testbed envisioned as outlined in Section 1.2 can be designed using the above described specifications and boundary conditions. This design process will start by researching and evaluating the available commercial off-the-shelf (COTS) receivers that can be used for the receivers of the testbed. These receivers will mimic the low frequency receivers of the satellites and are the most important part of the testbed. The specifications of the selected receiver will influence the design criteria of other components like the antennas, the astronomical source simulator, the data processing and storage facilities. When all components have been designed, each part will be constructed separately and its functionality shall be verified.

3.3 Functional design

The outline of the testbed described in Section 1.2 has been expanded to a functional design, as can be seen in Figure 3.1. This design shows the components and the workings of the testbed and is based on the specifications of OLFAR as are stated in Appendix A. The design ensures flexibility of the testbed by using software defined radios (SDRs) as receivers. This results in the use of software, which is inherently more flexible than hardware, for all processing in the testbed.

The simulated astronomical signals will be generated by a signal source. This source is visible on the left side of Figure 3.1. The signal will be received by multiple receivers, which will transmit the data they collect to a computer. The receivers are bundled in five groups of three, as each satellite of OLFAR also contains three separate receivers. These three receivers will share a single clock source (indicated by the red lines), as is the case in OLFAR, such that they will be perfectly synchronized. All groups of receivers will also be synchronized (indicated by the green lines), this will make sure that the measurement will start at exactly the same time for all receivers. This synchronization is only used to synchronize the start of the measurement to allow for realistic clock drift between the groups.

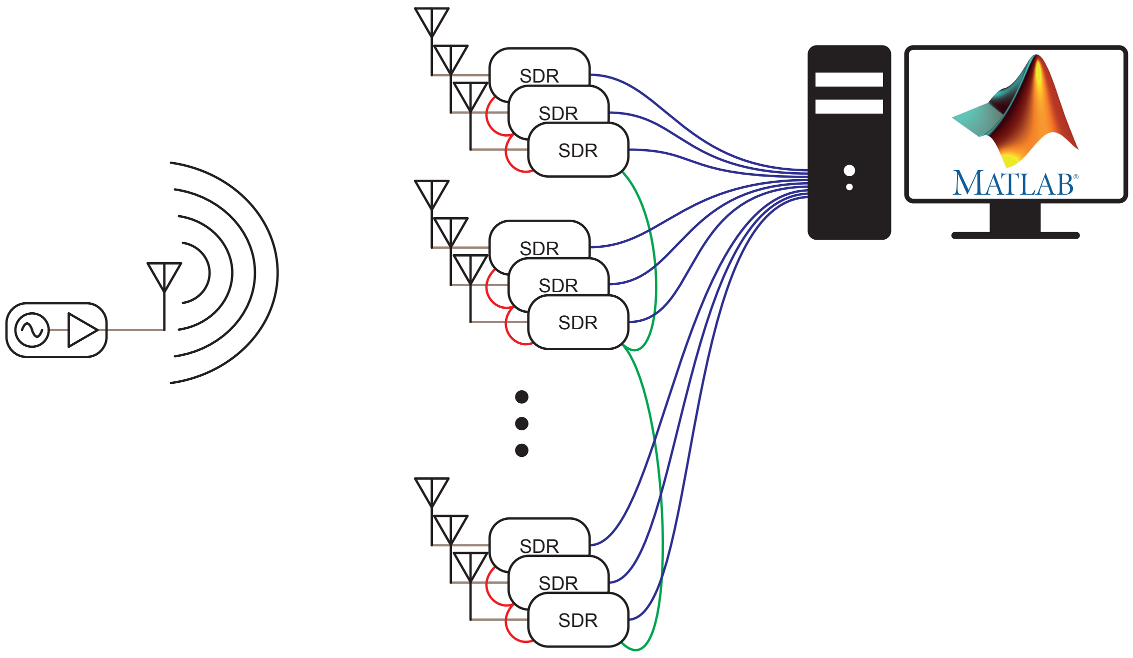


Figure 3.1: Functional diagram of the testbed

3.3.1 Global design choices

A large difference between the specifications of OLFAR and the design of the testbed is the frequency band which will be used. OLFAR will measure at low frequencies, but the testbed will use a much higher frequency. The choice for using a higher frequency is made in order to reduce the size of the testbed. The influence of the frequency on the size of the testbed is described by Equations 2.1 and 2.2. Two 1 MHz wide frequency bands have been chosen to be used for the testbed: the 1271–1272 MHz and the 1294–1295 MHz bands. These two frequency bands will be used because they are reserved for wideband experiments. People with an amateur radio license are allowed to transmit in these bands with up to 120 W peak envelope power [13, 14].

For the design of the testbed, the frequency centered between the two frequency bands will be used for all calculations. This frequency is 1283 MHz and the used wavelength for the testbed is thus 0.2337 m. The maximum baseline length is selected to be 5λ, which corresponds to 1.17 m. This length will ensure the portability of the testbed whilst the angular resolution remains adequate. According to Equation 2.1, the angular resolution becomes 0.20 rad or 11.5°. The chosen value for this maximum baseline distance has been reviewed in multiple simulations by my supervisor, P. van Vugt. These simulations demonstrated that the distance was adequately long. By using this baseline length, Equation 2.2 states that the distance between the receivers and the source must be much larger than 5.84 m.

3.3.2 Components of the testbed

The testbed can roughly be split into five components: The astronomical source simulator, the receiving antennas, the receivers, the software, and the physical construction. The purpose of each component is introduced below and each component will be elaborated in more detail in subsequent chapters.

Astronomical source simulator

Signals transmitted by astronomical sources are mostly wideband noise-like signals with very low or no polarization [3, 5]. In the testbed, comparable signals will be created and transmitted by the astronomical source simulator. A signal generator will create wideband noise in one of the chosen frequency bands and these signals will be transmitted with a suitable antenna.

Observational antenna system

The three antennas mounted on each OLFAR satellite are designed to have an omnidirectional field of view. Each antenna consists of two pieces of almost 5 meters, connected to a receiver. They are used as an active short antenna system as is stated in Table A.1. In the testbed, this antenna system will be replaced by three orthogonally mounted antennas. The antennas will have a generic 50 Ω output impedance which matches the input impedance of the receivers.

Receivers

The low frequency receivers of OLFAR are represented in the testbed by a different type of receiver. These receivers will not have a high input impedance and will not be used for active antennas. Instead they will consist of COTS hardware with a standard 50 Ω input impedance. The receivers must be able to receive the signals from both chosen frequency bands, and they should support clock sharing and clock synchronization. Many typical receiver specifications like linearity, noise figure, and gain are mostly irrelevant as the signal power from the source can be chosen almost arbitrarily. However, the stability of the oscillator is of importance as it determines the maximum integration time of the receivers.

Software

Part of the functionality of the testbed will be implemented in software. Two different types of software will be used: One part of the software will be used to arrange the communication between the computer and the receivers. The other part, implemented in MATLAB, will be used to control the receivers and perform all data processing.

Physical construction

The physical construction of the testbed will hold all previously mentioned components in place. Especially the mounting of the observational antenna system is of importance, as it highly influences the accuracy and repeatability of the measurements performed by the testbed.

Chapter 4
Astronomical source simulator

The purpose of the astronomical source simulator is to resemble a general astronomical source. As indicated in the previous chapter the source should therefore transmit a broadband noise-like unpolarized signal. But, as the antennas from the observational antenna system will be linearly polarized, they are unable to make a distinction between unpolarized and circular polarized signals. This results in the possibility to let the source simulator transmit circularly polarized signals instead of unpolarized signals. This choice will reduce the complexity of the source. Only one signal generator with one antenna is necessary to generate a circularly polarized output signal, but to generate an unpolarized signal two generators and two antennas are needed. However, the algorithms can measure the polarization by combining the signals from multiple antennas. Thus, if the algorithms incorporate these measurements the astronomical source simulator should be adapted to transmit real unpolarized signals.

4.1 Specifications

The spectral specifications of the source simulator where already provided in the functional design in Section 3.3. Together with the choice to transmit circularly polarized signals the following list of specifications for the astronomical source simulator can be created:

- Transmit with a center frequency of 1271.5 Mhz
- Transmit with a center frequency of 1294.5 Mhz
- Generate noise with a bandwidth of 1 MHz
- Generate a circularly polarized output signal

4.2 Implementation

The source simulator will consist of two parts, a signal generator and an antenna. The implementation of these two parts will be described separately.

4.2.1 Signal generator

The signal that should be generated is a straightforward signal to generate, the signal can be generated with ease by many COTS signal generators. During my assignment an Agilent E4438C vector signal generator was available, an illustration of the signal generator can be seen in Figure 4.1. This vector signal generator is able to generate noise signals with bandwidths up to 2 MHz within a frequency range from 250 KHz to 6 GHz. These specifications make this signal generator an easy and capable signal source for the testbed.

Figure 4.1: The used vector signal generator

4.2.2 Source antenna

For the design of the source antenna a helical type of antenna was chosen. This type is stated to have a very low axial ratio and the design has large construction tolerances [15, 16]. The axial ratio of the antenna is of importance as it indicates the ratio between the magnitude of the transmitted signal in both linear polarizations. When this ratio is 0 dB, both magnitudes are equal and the transmitted signal will contain no linear polarization and is thus fully circularly polarized.

The antenna design was based on well-known formulas as stated in many textbooks [17]. To lower the axial ratio of the antenna even more, the design was adapted to include a tapered end [18, 15]. The antenna was designed to consist of 16 normal windings and 2 tapered windings. This design proved to be capable of achieving a very low axial ratio of 0.2 dB with an antenna designed for a frequency of 900 MHz [18].

As a construction material, perspex (PMMA) was chosen for the antenna. This type of acrylic is commonly used for the construction of antennas due to the ease of use, the strength, and the reasonably low dielectric constant. For rigidity two pieces of perspex where used to fixate the helically wound conductor in both the X and Y axis. The constructed antenna can be seen in Figure 4.2.

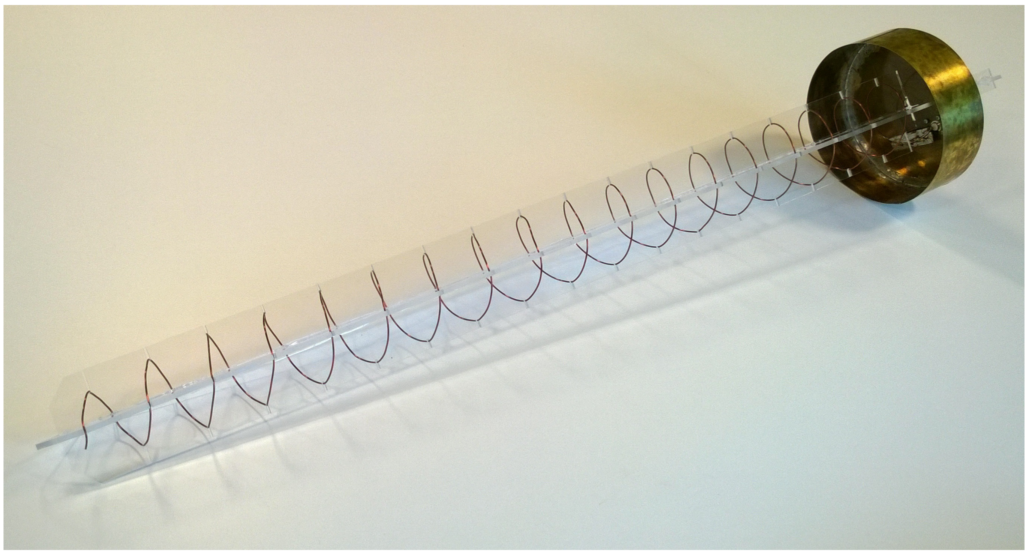


Figure 4.2: The constructed source antenna

4.3 Measurements

To verify the performance of the constructed astronomical source simulator only the antenna has to be validated. The used signal source is COTS equipment with clear specifications. To verify the functionality of the antenna the voltage reflection coefficient, better known as the S11 scattering parameter, was be measured. This measurement has been performed in a frequency range from 100 MHz to 10 GHz with an Agilent N5230A network analyzer, and the result can be seen in Figure 4.3.

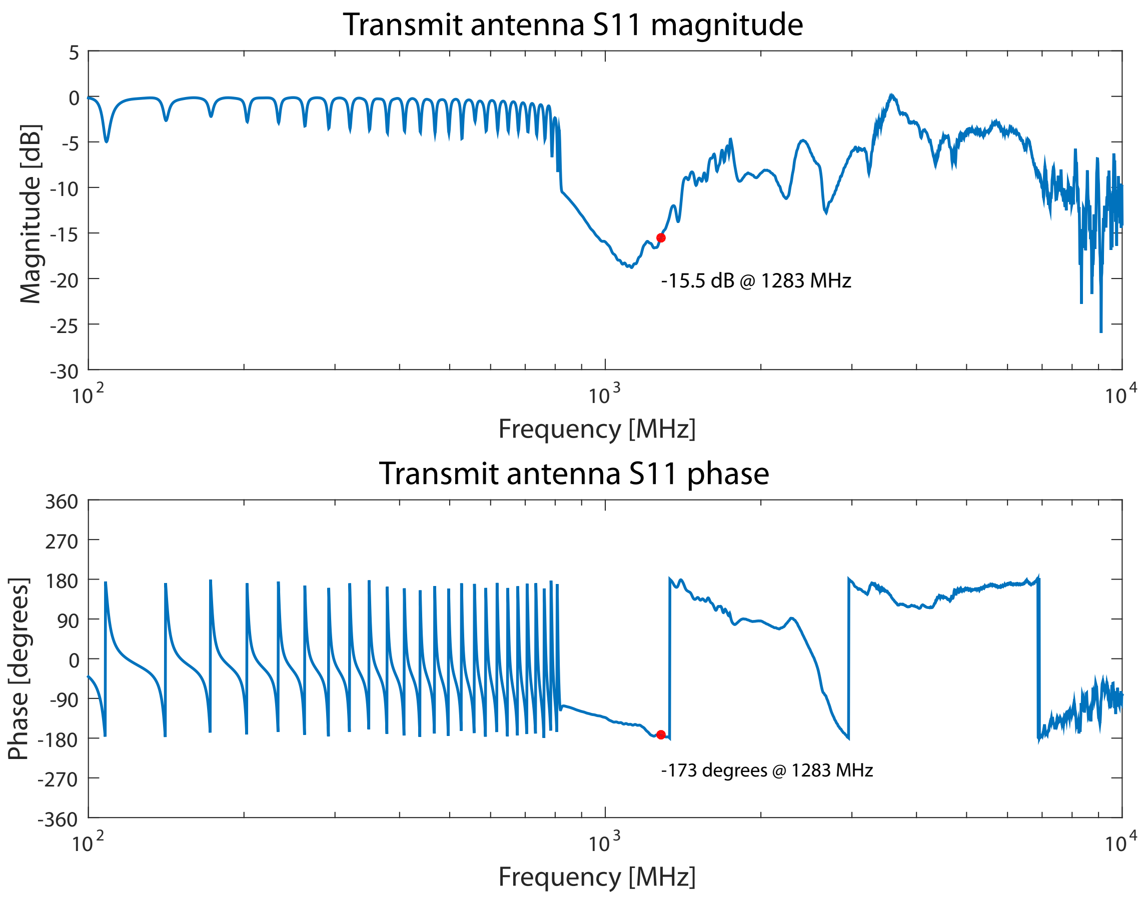


Figure 4.3: The S11 of the source antenna

As can be seen in Figure 4.3, the S11 is at its lowest at 1124 MHz where the value is -18.8 dB. At the frequency of interest, 1283 MHz, the S11 has a value of -15.5 dB. Also visible in the graph are the many dips at lower frequencies, these repeat roughly every 30 MHz.

From the S11 measurement of the antenna its voltage standing wave ratio (VSWR) has been calculated. This property is used in many cases to define the antenna's performance and for comparisons between antennas. The VSWR of the source antenna is plotted in Figure 4.4, together with a zoomed-in version around the frequency of interest.

The antenna logically has the lowest VSWR at 1124 MHz as the S11 is lowest at that frequency, the VSWR at this frequency is 1:1.3. The VSWR at the frequency of interest is slightly higher with a value of 1:1.4.

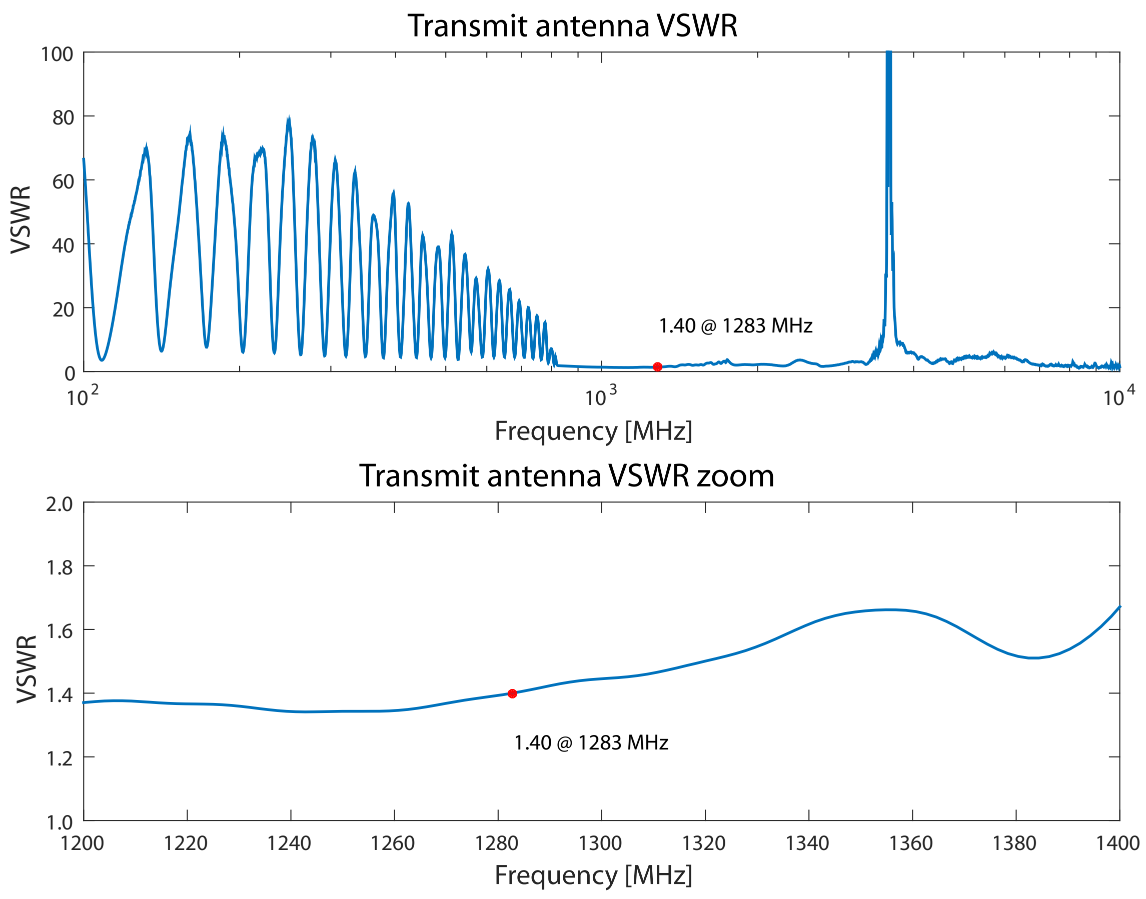


Figure 4.4: The VSWR of the source antenna

A second measurement that has to be performed to verify the functionality of the antenna is to measure the axial ratio. This is one of the important specifications of the antenna. However, during my assignment unfortunately no equipment was available to measure this parameter.

4.4 Analysis

Both the measurement of the S11 in Figure 4.3, and the calculation of the VSWR in Figure 4.4 show resonances in frequencies up to 800 MHz. These are in all likelihood caused by the helix antenna transmitting in normal mode.

The measurements indicate that the resonance frequency of the source antenna is not centered on the desired frequency of 1283 MHz. However, in the desired frequency ranges stated by the specifications, the antenna has a VSWR of 1:1.4. A VSWR of less than two is generally considered as very good, and is used as an indicator that no further matching is necessary. The specifications regarding the frequency range of the antenna are therefore met.

The axial ratio of the antenna should have been determined. No measurement of the axial ratio has been executed and thus performing this measurement is highly recommended as future work.

Chapter 5
Observational antenna system

The observational antenna system imitates the low frequency antennas present on each OLFAR satellite. The antenna system will consist of three orthogonal antennas to receive signals from all directions. The antenna system will receive the signals from the astronomical source simulator, just as the antennas of OLFAR will from a true astronomical source. But, for the observational antenna system a different antenna type will be used. The antennas used in OLFAR will be active short antennas with a dipole-like antenna pattern [12]. This antenna design is chosen because the length of non-active antennas which are usable at low frequencies would be too large. For example, a standard half-wave dipole created for the lowest operating frequency of OLFAR, 0.3 MHz, would be 500 m long. This clearly is a very unwieldy or even unfeasible length to be used for a nano satellite.

The testbed will be operating at a much higher frequency. At the center frequency of 1283 MHz, a standard half-wave dipole will only have a length of 12 cm. This is a very manageable length and the challenges imposed by active antenna design are therefore unnecessary for the testbed. The antenna patterns of the antennas of the observational antenna system should be dipole-like, just as those from OLFAR. Because antennas are reciprocal, the antenna pattern of an antenna while transmitting will be identical to the antenna pattern while receiving. This characteristic will be used to verify if the pattern is dipole-like in simulation.

One factor that produces a large problem in the observational antenna system design of the testbed are the antenna feed lines. These feed lines will connect the antennas to the receivers which will be placed elsewhere. If a feed line, or another conductor, is placed in the electromagnetic field of an antenna, common-mode currents will be generated in that conductor. These currents will generate an additional electromagnetic field which will interfere with the original field of the antenna. This effect will result in a distorted antenna pattern of the antenna. In many cases this problem is solved by placing the antenna far from any conductors. When this is not possible, like for example with the feed lines, the conductors can be placed orthogonal to the antenna which will negate their influence.

Most antenna systems consist of only one or two antennas at one location. This makes it possible to place the feed line orthogonal to the antennas and negate its influence on the antenna pattern. In OLFAR the problem of nearby conductors is not present. The only conductor is the satellite body which is placed in the center of the antennas where its influence on the antenna pattern is very small. However, in the observational antenna system for the testbed these solutions are not possible, because placing three antennas orthogonally at one location is necessary. Since a 3D space consists of three orthogonal directions, no orthogonal direction remains for placing the feed lines to negate their influence. Also, the receivers are too large to be placed in the center of the antennas. In the design of the observational antenna system this problem must thus be taken into account.

The antenna system has been designed and extensive simulations have been performed to verify its performance. All result are presented in this chapter and they should provide the reader with enough information to be able to produce and test a prototype of the system.

5.1 Specifications

The design process of the observational antenna system starts with determining the specifications. These specifications will ensure that the antenna system resembles the low frequency antennas of OLFAR [12]:

- Three orthogonal antennas
- Antenna phase centers close together
- Linear polarized antennas
- Omnidirectional dipole-like antenna patterns
- Center frequency of 1283 MHz
- Similar antenna patterns for each antenna
- 50 Ω output impedances

5.2 Antennas

There are not many systems which require three orthogonal antennas. Those systems are therefore not often described in literature and are also not COTS available. However, two methods to create a three orthogonal antenna system have been found and examined. The first method uses three monopole sleeve antennas which are placed such that their phase centers are close together [19]. A second method uses dipole antennas under a specific angle such that the influence of the feed lines on all antennas are equal [20, 21]. The required angle for this purpose is 54.7° with respect to the feed line. For the testbed, the second method will be used to construct the observational antenna system. This choice is made because the antenna pattern of the antennas should resemble the antenna pattern of an dipole. This is not the case with the first method as it uses monopole antennas.

5.2.1 Single dipole in free space

The starting point of the antenna system will be the antenna pattern of a single dipole in free space. This antenna produces the reference antenna pattern which should be approximated by the antennas in the completed observational antenna system.

A model of a single dipole in free space has been created in the NEC language, the code of the model can be found in Figure B.1. The dipole has been rotated about the Y-axis such that it is tilted 35.3° with respect to the X-axis. Using this angle will make sure that the angle between the antennas and feed line are equal for all three antennas when they will be added to the model [20, 21]. The dipole is modeled to be constructed from copper and to be connected to an 100 Ω balun. The length of the elements has been optimized to 55 mm, which results in the lowest VSWR in simulation. The results of this optimization can be seen in Table 5.1.

Table 5.1: Simulated VSWR with different dipole element lengths

Element length	VSWR (100 Ω)
0.056	1.37
0.055	1.35
0.054	1.45
0.053	1.58

This single dipole antenna has been simulated with the program 4nec2 [22]. The simulated antenna pattern can be seen in Figure 5.1. This antenna pattern will be used as the reference pattern for the antennas of the observational antenna system.

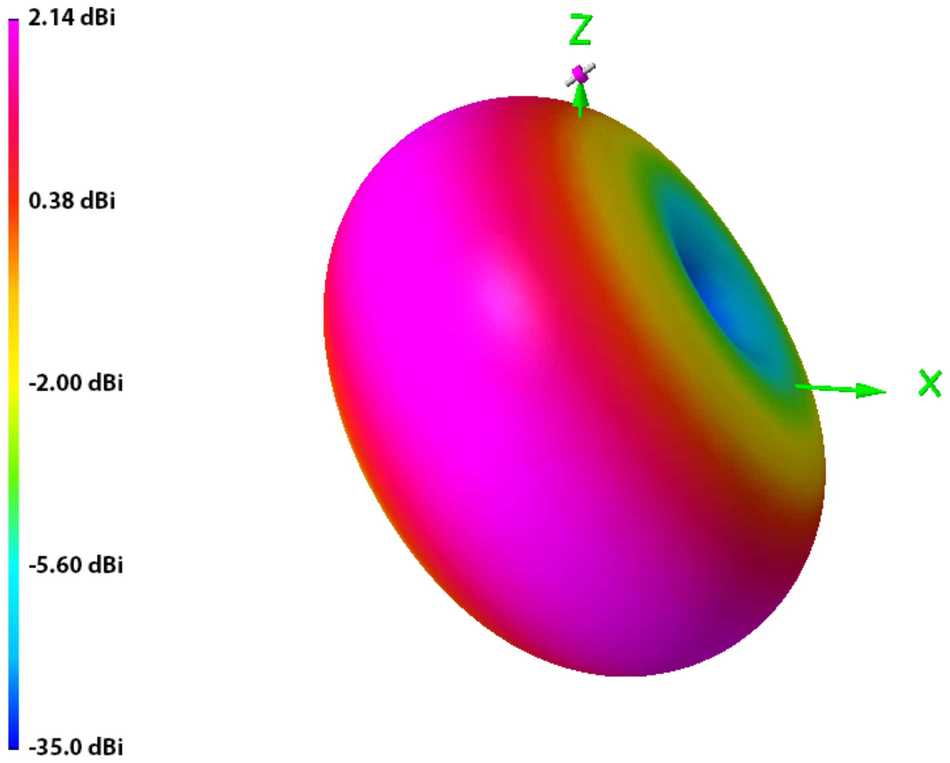


Figure 5.1: One dipole in free space

5.2.2 Single dipole above ground

The first step to make the model more realistic is to incorporate the influence of the surface of the Earth. The Earth surface is integrated in the simulation by adding a mesh structure 1 m below the antenna. The conductivity of this mesh has been set to 0.2 S/m to represent the most conductive type of soil, wet ground, at 1283 MHz [23]. To incorporate the permittivity of the soil, this mesh has been coated with a material with a dielectric constant of 30 [23].

The results of the simulation proved to be indistinguishable from the simulation of a dipole in free space. The influence of ground at 1 m distance is thus negligible to the antenna pattern of the antenna. This result was expected because the distance between the antenna system and the ground is relatively large. The ground is located in the far-field of the antenna and therefore it has little influence on the antenna pattern.

5.2.3 Three dipoles above ground

The model has been extended to incorporate the two additional antennas. These are placed orthogonal to the first antenna and each other. The additional antennas are not driven in the simulation, but are terminated with an 100 Ω load. The phase center of the antennas is placed 1 cm from the center of the antenna system. This offset is necessary for construction purposes. An illustration of the antenna system model can be seen in Figure 5.2, and the simulated antenna pattern is provided in Figure 5.3.

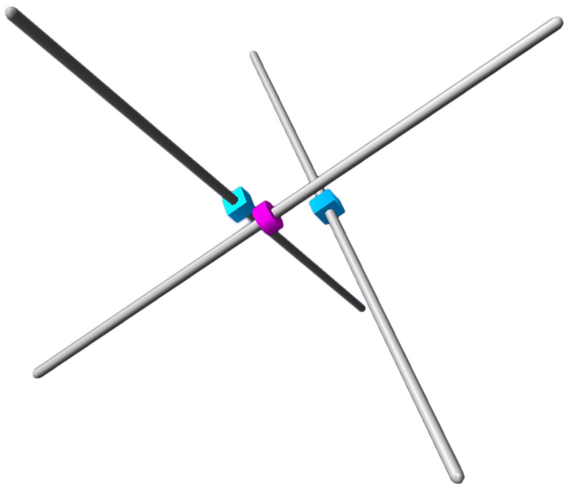


Figure 5.2: Three dipole antenna system

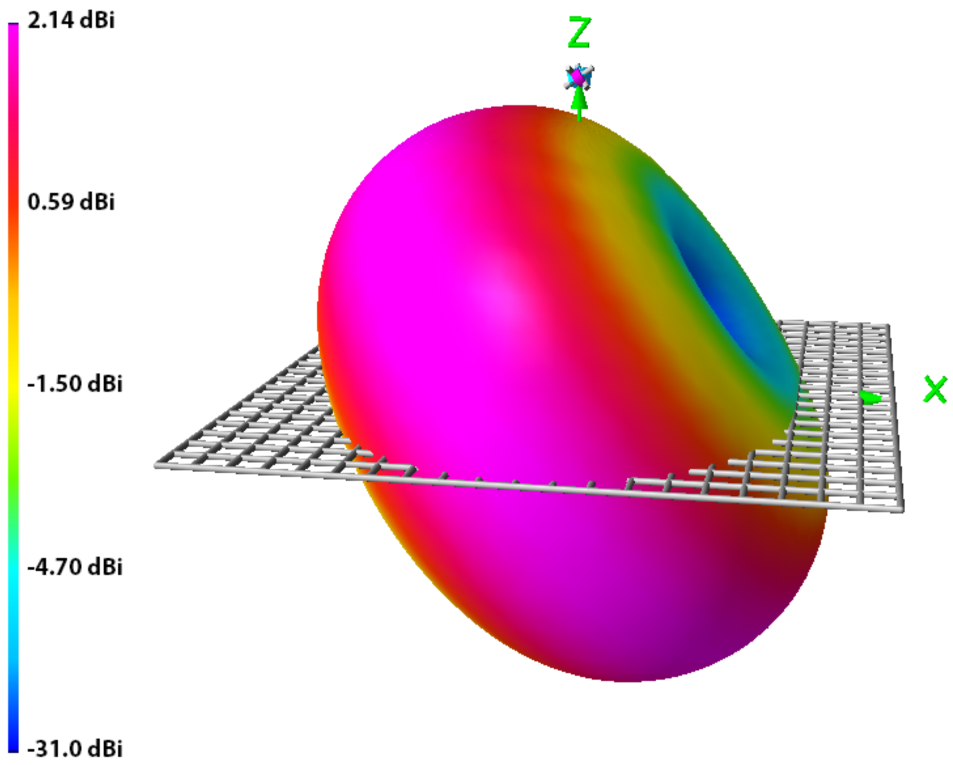
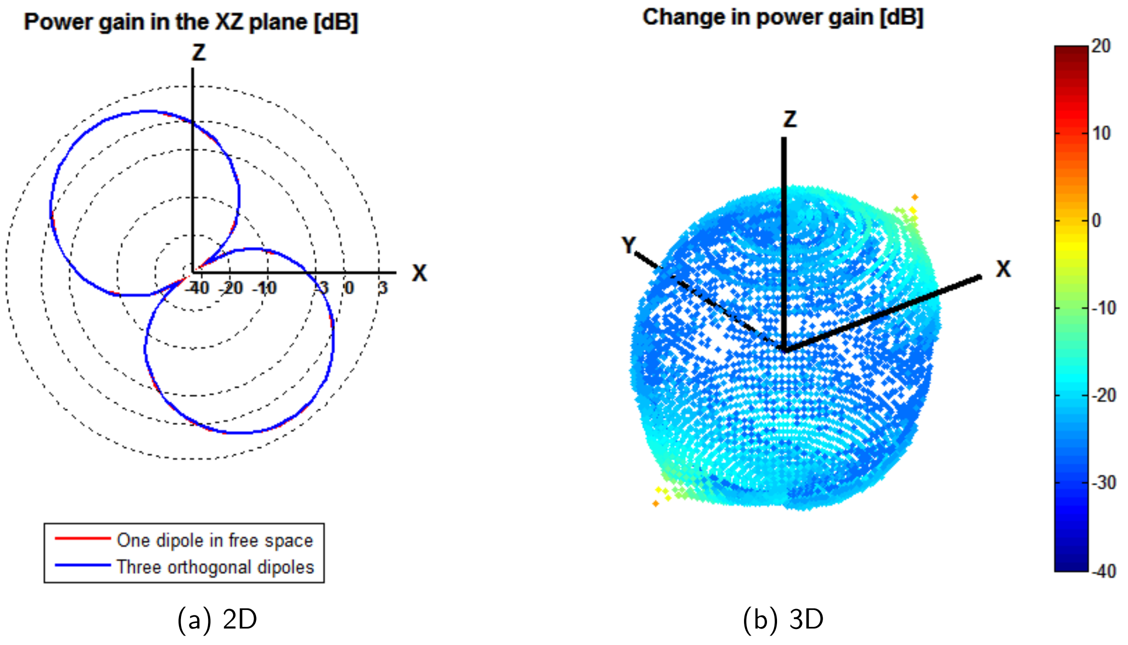


Figure 5.3: Three dipoles above ground

As can be seen in Figure 5.3, the antenna pattern of this antenna system is very similar to the single dipole in free space. The antenna pattern has been calculated with a resolution of 3° for all directions. This data has been used to compare the antenna pattern of the system to the reference antenna pattern. The difference between the antenna patterns in the XZ-plane is shown in Figure 5.4a.

The difference between the antenna patterns has also been calculated for all measurement points to provide a 3D visualization of the differences. The differences are expressed in dB and have been plotted in Figure 5.4b. As this figure presents the difference in dB, a value of -40 dB indicates that the antenna pattern of the system differs 0.01% from the reference pattern at that point. Likewise, a value of -20 dB indicates an 1% difference and a value of 0 dB indicates that the gain in that direction is twice as large or twice as small as the reference.



(a) 2D (b) 3D

Figure 5.4: The difference in antenna pattern between one dipole in free space and three dipoles above ground

Figure 5.4b displays one downside of this type of relative comparison. The antenna pattern of a dipole has two directions where the gain is very small, one in each direction along the axis of the antenna. Due to the small gain in these directions, a small absolute difference between the patterns will generate a large relative difference. In the figure this is visible as the two yellow and orange peaks on the sphere. These differences between the two antenna patterns in these two directions will not be taken in to account during the analysis in this chapter. These differences will only make a very small absolute difference and the gain in those directions will always be very small compared to the gain in other directions.

When Figure 5.4b is analyzed for the directions of interest, it is clear that the difference between the two antenna patterns is very small. The error is between -30 and -15 dB and the influence of the addition of the other elements is almost negligible. This small error is inevitable in this design. The error could be reduced by placing the phase centers of the antennas closer together. This will result in a more orthogonal and therefore ideal setup. Unfortunately, this distance is inevitable to keep the construction of the antenna feasible.

5.2.4 Three dipoles above ground with feed

Adding the feed lines is the last necessary step to complete the model of the observational antenna system. The feed lines are modeled as a single copper conductor connected to the ground mesh and rising up to the antennas. The conductor ends in the middle of the antennas. This conductor models a bundle of three separate coaxial cables, each connected to one of the antennas. The model has been simulated and the result can be seen in Figure 5.5.

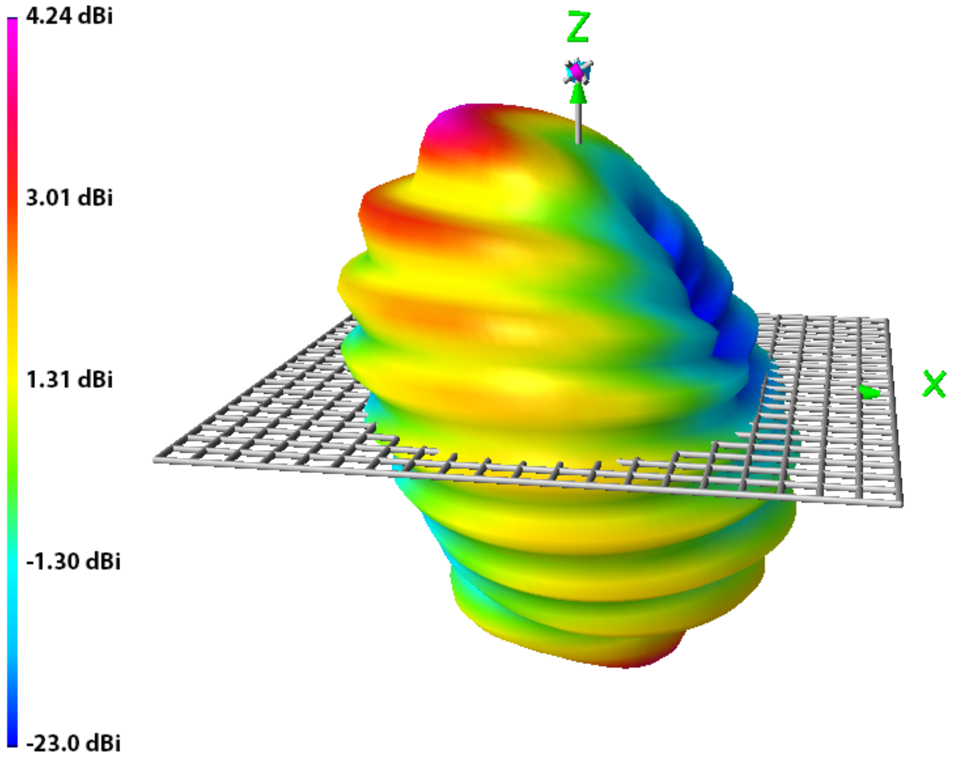
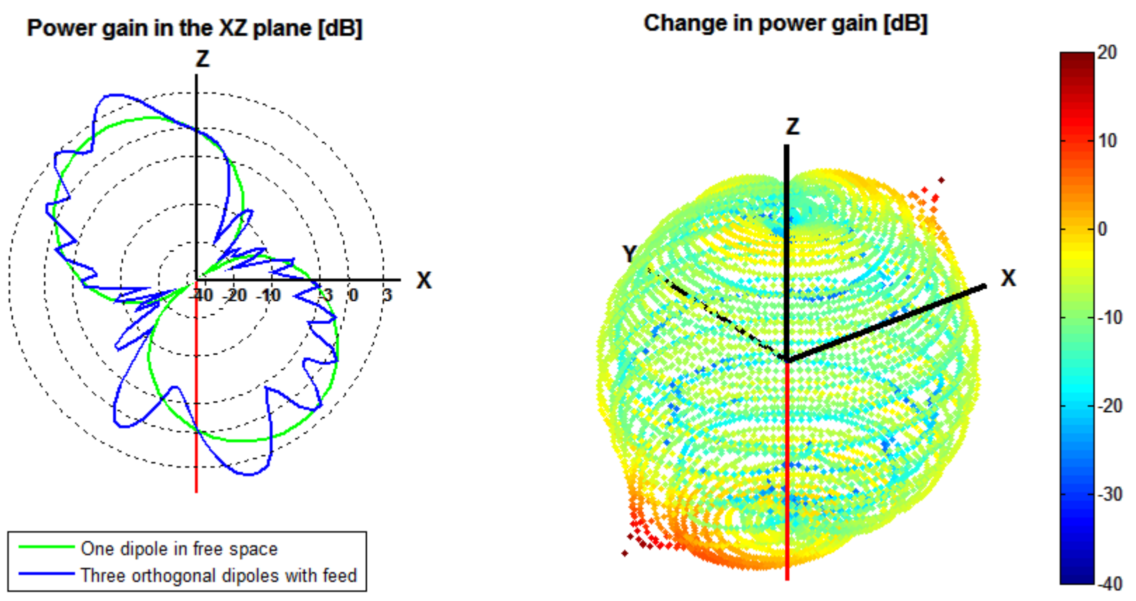


Figure 5.5: Three dipoles above ground with feed

This simulation reveals the devastating influence of the feed lines on the antenna pattern. The antenna field induces common mode currents on these feed lines which result in a second electromagnetic field. This second field interacts with the original field of the antenna. This is visible in the antenna pattern by the generated lobes. The difference between this antenna pattern and the reference has been calculated. The result is given in Figure 5.6.

Figure 5.6a clearly shows the lobes which are present in the antenna pattern due to the presence of the feed lines, which is represented by the red line. In the 3D representation of Figure 5.6b the magnitude of the differences can be found. The difference between the antenna patterns of the system and the reference is in general around -5 dB. But in some directions, especially in the general direction of the feed lines, the error can be as large as 5 dB. Thus, the antenna pattern of the antenna has an insurmountable deviation from the reference and the influence of the feed line has to be reduced greatly.



(a) 2D (b) 3D

Figure 5.6: The difference in antenna pattern between one dipole in free space and three dipoles above ground with feed

5.3 Feed lines

As mentioned before, the distortions in the antenna pattern are due to the common mode currents on the feed lines induced by the field of the antenna itself. These currents should be visible as a rectified sinusoidal shaped current distribution on the feed lines. This distribution should have a repetition distance which is half that of the wavelength of the transmitted signal. The current distribution on the feed lines from the antenna system has been simulated to verify multiple solutions. The results from these simulations are provided in Figure 5.7. The original case, the above described situation without any applied solution, is presented by the black line.

The black line displays the current distribution on the feed lines from the antenna at 1 m height to the ground. The rectified sinusoidal shape is clearly visible and the repetition length is around 12 cm. This length is half the wavelength of the amplitude of the signal and thus equal to the repetition distance of the magnitude of the signal, as was expected. The magnitude of the current distribution ranges from 0.8 mA to 3 mA.

5.3.1 Cover with ferrites

Undesired common mode currents are a regularly occurring problem in the field of electromagnetic compatibility (EMC). This frequently occurring problem has an often used solution which is suitable in most cases; adding ferrites. When placed around a conductor, ferrites will increase the impedance of the conductor for common mode currents. The impedance for differential mode currents will remain unaffected.

To verify this solution, a suitable ferrite should be selected which will be modeled in the simulation. This ferrite should be useable for high frequencies and it should have a high impedance. For the antenna system, the HFB170070-000 from Laird Technologies has been selected as a fitting candidate for the ferrite. This ferrite has been designed specif-

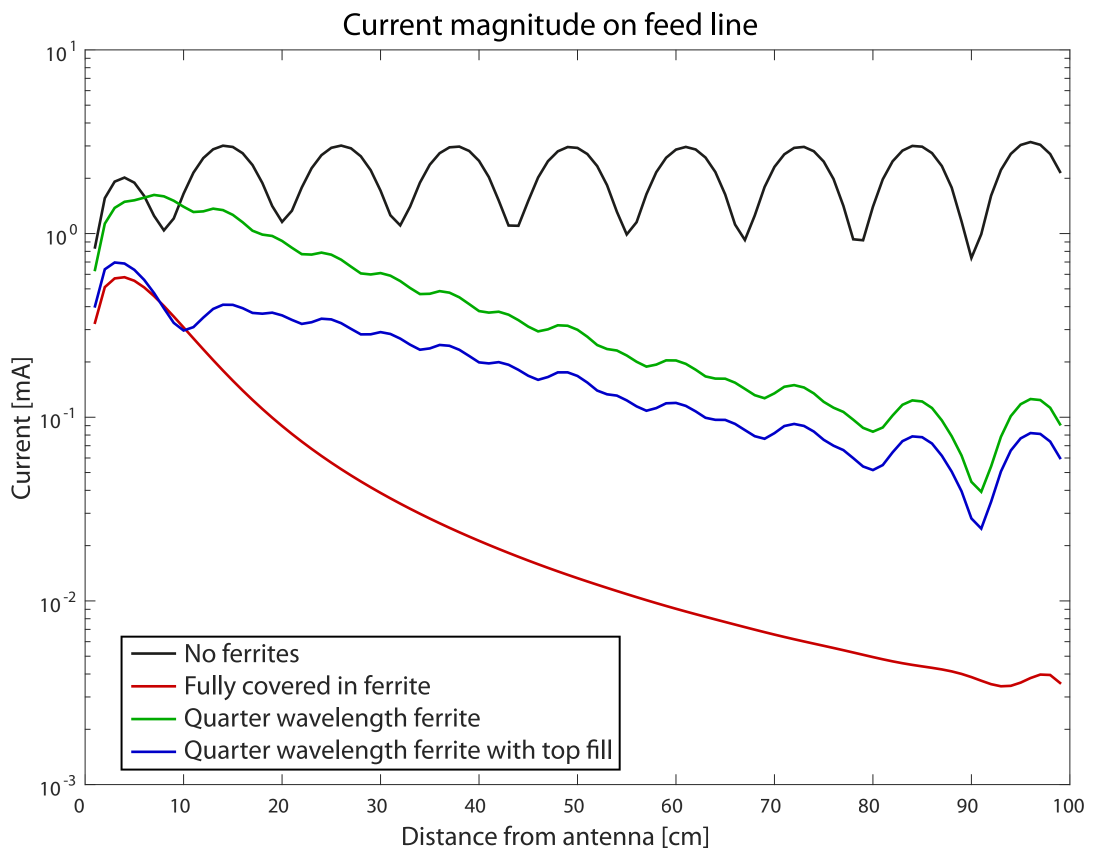


Figure 5.7: Current distribution on the feed lines of the antenna system

ically to be used for high frequency applications and it has a high nominal impedance for its length. This ferrite provides a nominal impedance per length of about 140 Ω/cm, which is higher than other available high frequency ferrites.

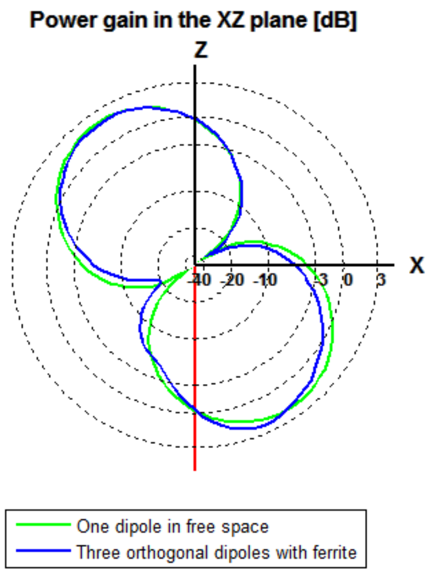
The influence of the ferrites has been simulated. In this simulation the feed lines of the antenna system has been fully covered with ferrites to achieve the maximum effect. The resulting current distribution on the feed line in the simulation is displayed by the red line in Figure 5.7.

The current distribution has clearly been improved compared to the baseline current distribution in black. The magnitude of the currents now only ranges from 0.003mA to 0.6 mA. However, the implementation of this solution requires a very large amount of ferrites. This will increase the cost of the testbed considerably. Thus, a trade-off between the number of ferrites and the error in the antenna pattern is desired.

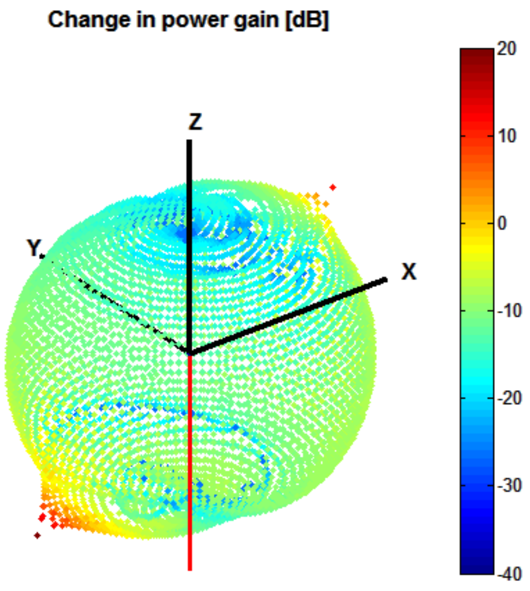
5.3.2 Distributed ferrites

The current distribution along the feed lines has a maximum every 12 cm. The ferrites will have the largest influence on the common mode current when placed on these points. However, the locations of these maxima are variable and are dependent on the length of the feed line and the placement of the ferrites. Thus, to ensure that the ferrites are always placed on the points of maximum magnitude, they will be placed evenly distributed along the feed line every half wavelength of the magnitude, which is every 6 cm. This situation has been simulated and the resulting current distribution is displayed by the green line in Figure 5.7.

The resulting current distribution ranges from 0.04 mA to 2 mA. The overall magnitude is, as expected, smaller than without ferrites and larger than the solution with fully covered feed lines. To verify if this trade-off produces adequate suppression of the common mode currents, the antenna pattern has to be inspected. The difference between the reference antenna patten and the pattern resulting from the antenna system when an even distribution of ferrites is used is provided in Figure 5.8.



(a) 2D



(b) 3D

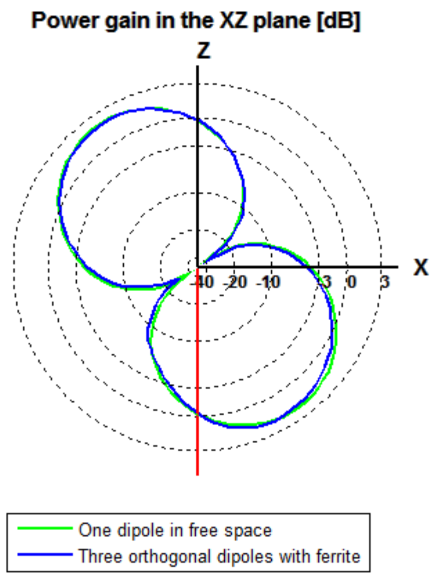
Figure 5.8: The difference in antenna pattern between one dipole in free space and three dipoles with feed lines with ferrites every quarter wavelength

Figure 5.8a clearly shows that the lobes have almost disappeared. However, irregularities still remain, especially towards the feed lines indicated in red. This conclusion is backed by the results of Figure 5.8b. The figure displays areas with low deviation from the reference but the major part of the figure displays relatively large differences. The difference is -10 dB on average, with larger values near the feed lines. These differences are relatively large and the antenna pattern does not adequately resemble the antenna pattern of the reference. Thus, more ferrites will have to be added than there have been used for this trade-off.

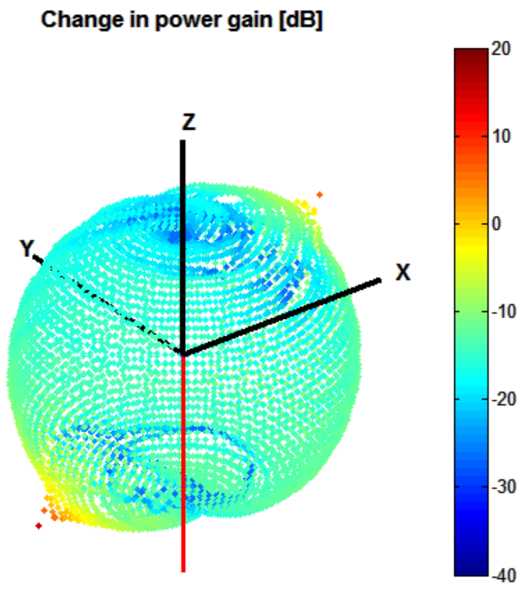
5.3.3 Smart use of ferrites

The field generated by the antenna is largest close to the antenna. The magnitude of the common mode current is largest there as well with the evenly distributed ferrites. This position is thus the logical choice to place the necessary additional ferrites. Additional ferrites have been added to the model to fill the first 12 cm of the feed lines, equal to one wavelength. A small length of feed line is necessary to maneuver the cable from the antenna through the ferrites. Thus, a small length of line cannot be covered with ferrites. The feed lines in the model have been fully covered with ferrites from 3 to 12 cm from the antennas. The remainder of the feed lines have been covered with ferrites which where placed every quarter wavelength, which is every 6 cm. The current distribution on the feed lines with this distribution of ferrites is provided by the blue line in Figure 5.7.

The resulting current distribution is very similar to the current distribution of the model with the evenly distributed ferrites which was presented by the green line in the figure. However, the magnitude is more than halved, it ranges from 0.02 mA to 0.7 mA. If this decrease is adequate, must be determined by comparing the antenna patterns. The comparison between the results of this model and the reference can be seen in Figure 5.9.



(a) 2D



(b) 3D

Figure 5.9: The difference in antenna pattern between one dipole in free space and three dipoles with feed lines with ferrites every quarter wavelength and a top fill

A large resemblance between the model and the reference can be seen in Figure 5.9a. The difference between the antenna patterns is provided in Figure 5.9b. This figure indicates the resemblance as well, the average difference is around -15 dB or 3%. This difference has been settled upon as being acceptable. Thus, this distribution of ferrites is considered a satisfactory trade-off between the number of ferrites used and the error present in the antenna pattern.

5.4 Construction

The design of the observational antenna system has been inspired by the patent of a field probe antenna [21]. This field probe possesses the same geometry as is necessary for the antenna system. It also combines manufacturability with sturdiness by using printed circuit boards (PCBs) as construction material. The design of the field probe antenna can be seen in Figure 5.10.

The design of one of the antennas of the observational antenna system can be seen in Figure 5.11. The antenna consists of a single-sided rectangular PCB with a width of 21 mm and a height of 55 mm. It will have a hole in the center for the feed line. Solder pads are placed on the PCB for the antenna elements, the balun and the feed line. The antenna elements will be constructed from solid copper wire and will be tilted 35.3° compared to horizontal. The balun will be a standard balanced 100 Ω to unbalanced 50 Ω version in a surface-mounted device (SMD) package. General RG-174 coaxial cable will be used for the feed line, due to its small diameter.

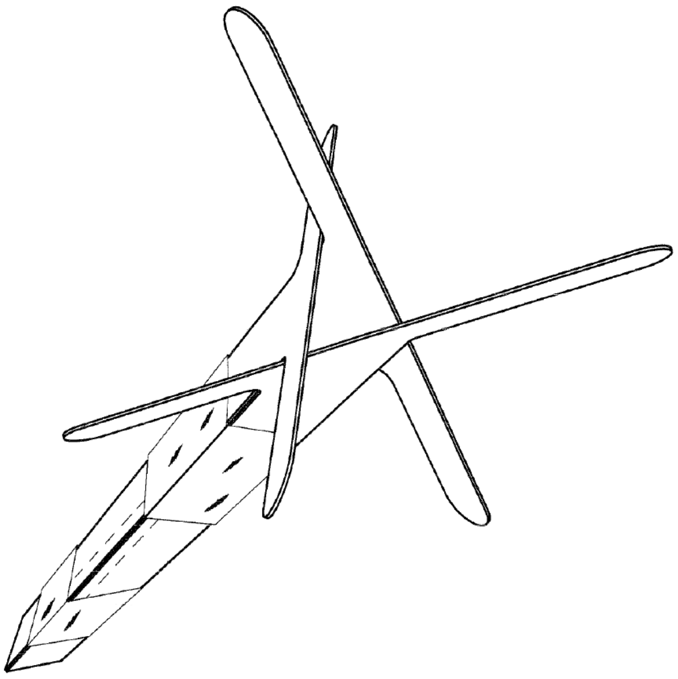
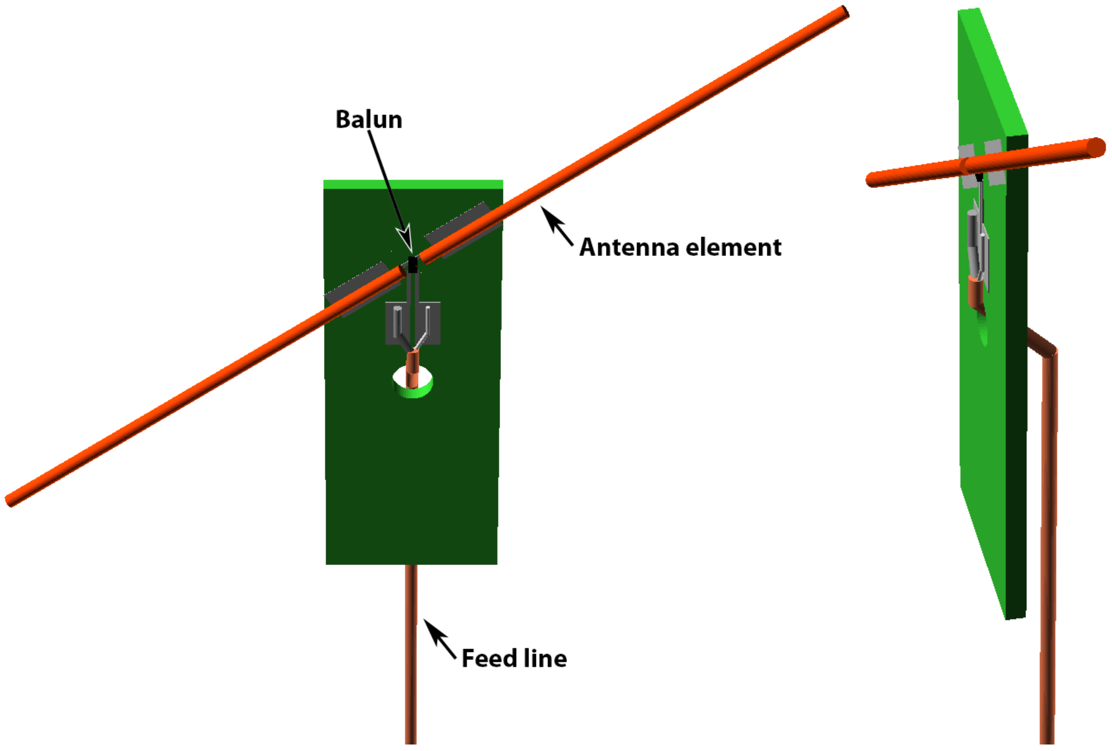


Figure 5.10: The design of the Stalk-type field probe antenna [21]

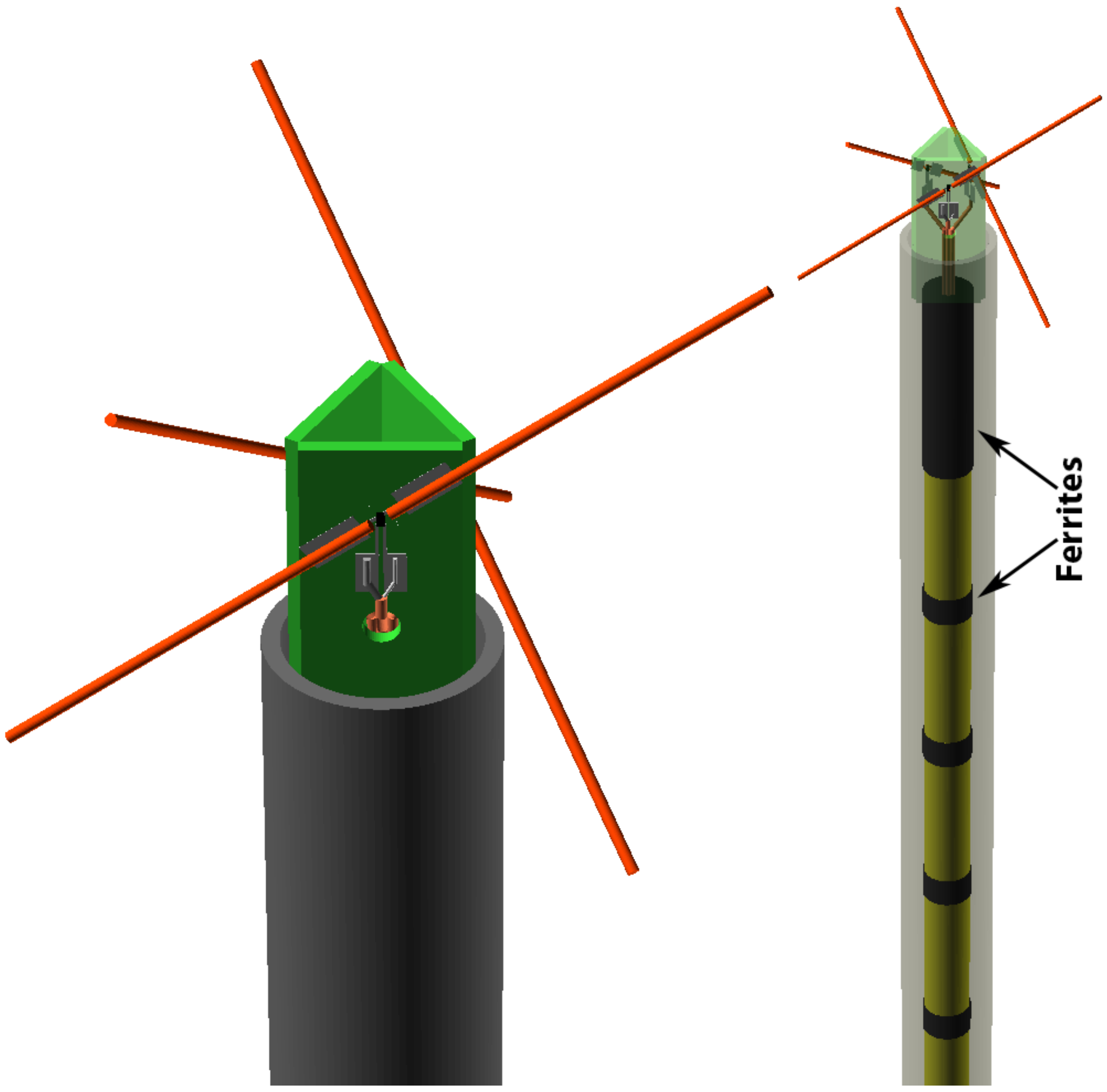


(a) Front (b) Side

Figure 5.11: The design of an astronomical antenna

Three antennas will be combined to create the observational antenna system. The antennas will each be rotated 120° around the feed line and will form a triangle. A model of the antenna system is displayed in Figure 5.12. As shown in the figure, the antenna PCBs will be inserted into a pipe. This pipe will be used for the mounting of the system. For this pipe general PVC plumbing pipe is used. This pipe has an outer diameter of 32 mm and an inner diameter of 27 mm.

A transparent view of the system is provided in Figure 5.12b. This view shows the placement of the ferrites along the feed line. The ferrites will be separated by pieces of PVC pipe, normally used for electrical installations. These pieces have an outer diameter of 16 mm and an inner diameter of 14 mm.



(a) Top of the antenna system (b) Transparent view

Figure 5.12: The astronomical antenna system

5.5 Conclusion

The designed observational antenna system corresponds perfectly with the determined specifications according to the simulations. It includes three orthogonal antennas which have their phase centers close together. Each antenna is linearly polarized has a dipole-like antenna pattern. The antenna pattern only differs around -15 dB or 3% from the ideal dipole-like antenna pattern.

The antennas have been designed to be used for a frequency of 1283 MHz and have an output impedance of 50 Ω. The antenna patterns will be equal for each antenna due to the symmetry of the construction.

The next step in the design of the observational antenna system is to build and test a prototype. This chapter should have provided an adequate amount of information and construction details to make this possible. Of this prototype, the antenna pattern of each antenna should be measured and compared to an ideal dipole-like antenna pattern.

Chapter 6

Receivers

The receivers of the testbed will convert the signals from the observational antenna system to digital data which can be interpreted by MATLAB. These receivers will have to filter, amplify and down-mix the analog signals. They also need to convert the analog signals to digital signals. However, all these operations are general practice for all receivers, and many forms of COTS hardware are therefore available. In this chapter these hardware options will be evaluated and the most suitable receiver is presented. Additional improvements have been made to this receiver, these will be presented in this chapter as well.

6.1 Specifications

Many of the specifications for the receivers were already mentioned in previous chapters. The spectral requirements where given in the functional design, as is the need for clock sharing and synchronization. The specifications for a high sample rate, high bandwidth and real-time processing where imposed by the methods for RFI mitigation.

A complete list of the specifications for the receivers can be compiled as follows:

- Frequency range from 1271 to 1295 MHz
- Bandwidth greater than 1 MHz
- 50 Ω input impedance
- Stable oscillator
- High sample rate
- Capable of clock sharing
- Capable of clock synchronization
- Capable of real-time data processing

6.2 Implementation

For the testbed, COTS SDR hardware has been be used to guarantee flexibility. With SDRs, most parameters of the receivers can be set in software. This makes it simple to alter the parameters according to the needs for the specific measurement. The data processing will also be performed in software. This results in a high adaptability of the used algorithms.

6.2.1 Hardware options

Eight capable COTS SDR hardware options have been selected for a thorough comparison. The specifications of each of the SDRs has been compared to the specifications which are stated for the receivers of the testbed. The full overview of the comparison can be found in Appendix C.

The comparison resulted in the selection of four SDRs which met almost all specifications. These where the BladeRF x40, HackRF One, USRP-2921 and the USRP B200. The NESDR was also included due to the very low cost of about 20 USD. This low cost might have made it justifiable to design or buy additional hardware to modify and improve the receiver. These five receivers have been compared and the USRP B200 and USRP-2921 where removed from the final selection due to their price. Procuring 15 of these SDRs would have resulted in unreasonably high costs for the testbed as these SDRs are respectively two and eight times more expensive than the other candidates.

The final selection for the receiver of the testbed consisted of the BladeRF x40, HackRF One and the NESDR. Of these three options, the BladeRF x40 has been selected as the most suitable candidate for the receivers of the testbed. The NESDR was discarded due to the large amount of time and effort that would have to be invested in improving this receiver. Without these improvements, the NESDR meets only half of the specifications and therefore is no viable candidate. The BladeRF x40 has been chosen in favor of the HackRF One mainly due to the presence of the field-programmable gate array (FPGA) in the receiver. The presence of this FPGA provides the receiver with more capabilities for real-time processing.

6.2.2 BladeRF

The BladeRF has been chosen as the most suitable device to be used as receiver for the testbed. The BladeRF is an open source device, for which the code can be freely adapted. Out-of-the-box the device has the specifications as listed in Table 6.1.

Table 6.1: BladeRF x40 specifications

Name	Nuand BladeRF x40
Price [USD]	420
Spectral specifications	
Frequency range [MHz]	300 to 3800
Instantaneous bandwidth [MHz]	28
Timing specifications	
Oscillator type	Factory calibrated 38.4 MHz VCTCXO controlled via a 16-bit DAC
Oscillator accuracy [PPM]	1
Signal processing specifications	
Sampling rate [MSPS]	40
Number of ADC bits	12
Effective number of bits	10
Spurious free dynamic range [dBc]	60
Technical specifications	
Input connector	SMA (50 Ω)
Output connector	USB 3.0
Onboard processing power	200 MHz ARM9 Processor, Altera Cyclone 4 FPGA
Software support	MATLAB, Simulink, GNU Radio, SDR#
Extras	Transmit capable (full duplex), External clock input, External clock output, Expansion header

With these specifications, the BladeRF almost meets the specifications of the testbed. The only specification which is not met, is the capability of clock synchronization. This specification is necessary for the testbed to ensure that all receivers start their measurement at the exact same moment, as was described in the functional design. However, as the BladeRF incorporates an FPGA of which the software is open source, this feature can still be implemented. A picture of the device is provided in Figure 6.1.

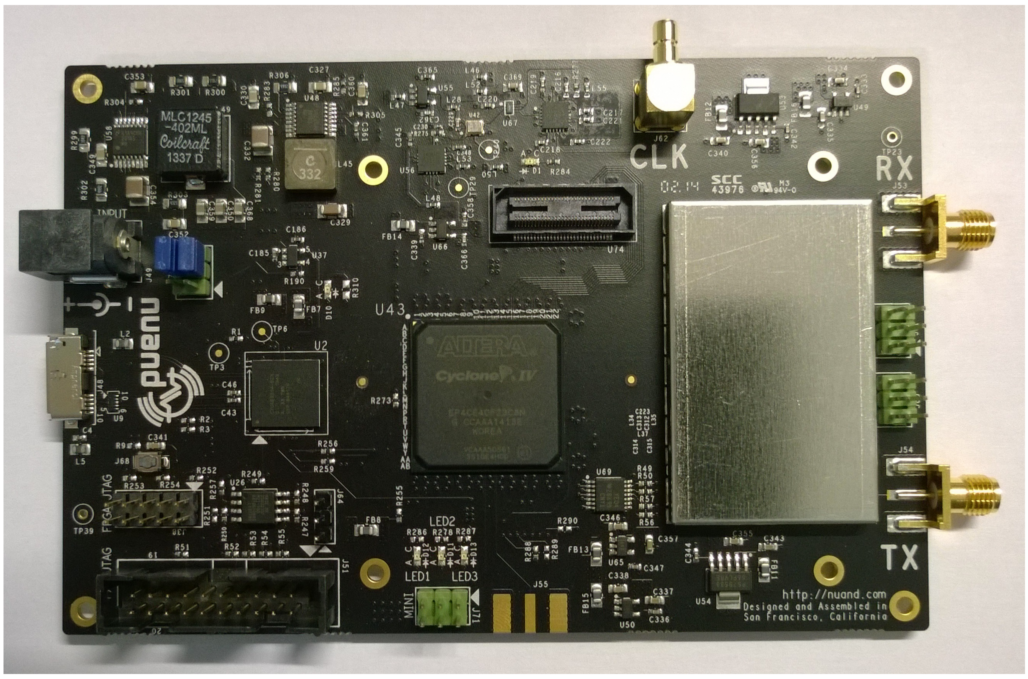


Figure 6.1: The Nuand BladeRF x40

6.3 Synchronization implementation

In OLFAR, the clocks of all satellites will be synchronized before the measurement takes place. This synchronization is necessary to perform the interferometry. Because the satellites are not connected, the synchronization will be performed via wireless communication. During the measurements, this wireless communication is not desirable as it can influence the measurements due to crosstalk. Thus, the synchronization will not be available during the measurements and the clocks of the satellites will start to drift with respect to each other.

The firmware of the BladeRF needs to be modified to incorporate this effect and to meet the clock synchronization specification. The firmware will be modified such that the clocks can be synchronized before the measurement. During the measurement, the synchronization has to stop and therefore the clocks will start to drift with respect to each other, just like will happen in OLFAR.

6.3.1 Clock distribution

An overview of the complete clock distribution network of the testbed is provided in Figure 6.2. The figure shows three different kinds of connections: The clock sharing within a satellite mock-up is indicated in red, the clock synchronization between the satellite mock-ups in blue and the connection to transmit a signal to indicate the start of the measurement is marked in green. The figure also reveals that three configurations are used for the receivers of the testbed: A MIMO slave, a MIMO master with PLL and a global/MIMO master. Each of these three different configurations will be described in more detail.

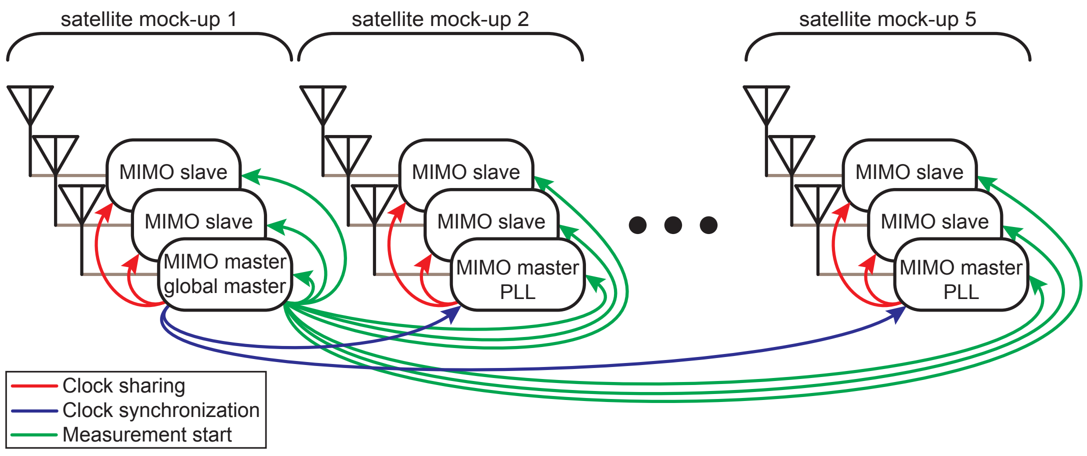


Figure 6.2: The clock distribution of the testbed

Global/MIMO master

Only a single receiver of the testbed will use this configuration. The receiver is configured as a multiple-input and multiple-output (MIMO) master, this means that it will be used as the master clock within its satellite mock-up. This satellite mock-up consists of three receivers grouped together, which represents the three receivers contained in a single OLFAR satellite. Within a satellite of OLFAR, the three separate receivers use the same master clock. The clock from this receiver will act as that master clock in the satellite mock-up.

The second purpose of this receiver is to act as the global reference clock for the entire testbed. The clocks from all satellite mock-ups will be synchronized to this clock in order to synchronize the entire testbed before the measurement starts.

MIMO master with PLL

Four receivers will have this configuration. They are configured as MIMO master, and will therefore be used as the master clock for their own satellite mock-ups. These four receivers will themselves synchronize their own clock to the reference clock of the 'global/MIMO master'. This synchronization is performed by using a phase-locked loop (PLL), which is locked to the signal from the reference clock. Details on this PLL and its implementation are provided in Section 6.4.1.

MIMO slave

The remaining ten receivers of the testbed will be configured as MIMO slaves. These receivers will use the clock provided by the MIMO master in their own satellite mock-up.

6.3.2 Measurement start

The green arrows in Figure 6.2 show the path of the signal which indicates the start of the measurement. This signal will also be transmitted by the 'global/MIMO master'. The signal indicates when all receivers should start the measurement phase.

6.4 FPGA firmware

The firmware of the FPGA is written in VHSIC hardware description language (VHDL). This language is specifically designed to program FPGAs. The firmware was modified to incorporate the synchronization of the start of the measurement. As indicated above, this feature consists of two parts: One part incorporates the PLL to synchronize the clocks of the receivers with the clock reference, the other part sends and receives the signal to start the measurement.

6.4.1 PLL

The PLL is necessary to synchronize the phases of the clocks of the MIMO masters to the reference clock of the testbed. The frequency reference of each BladeRF is a voltage controlled temperature compensated crystal oscillator (VCTCXO). The frequency of this oscillator can be adjusted with a voltage. This voltage is created by a 16-bit digital-to-analog converter (DAC) which is connected to the FPGA. In the original firmware the DAC is only used for calibration purposes and the output is fixed during operation. However, the presence of the voltage controlled oscillator and the DAC make it possible for a PLL to be implemented as shown by the block diagram of Figure 6.3 [24].

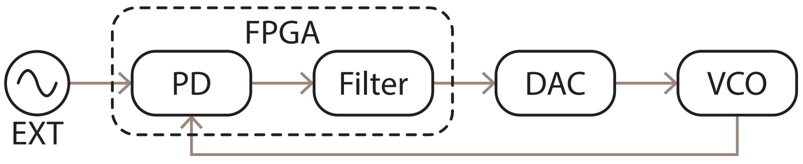


Figure 6.3: Block diagram of the PLL

The input signal for the PLL is an external reference signal. This signal will be generated by one of the SDRs which will be used as the master oscillator for the whole testbed. The phase detector is the first part of the PLL, it acts as the sensor of the PLL. It measures the difference in phase between the reference signal and the output of the VCO. This difference indicates if the phase of the VCO leads or lags compared to the reference, and therefore if the frequency of the VCO should be decreased or increased respectively. The output of the phase detector is not directly connected to the VCO, it is first filtered by the loop filter. This filter is a low-pass filter which removes the high frequency noise generated by the phase detector. It should also incorporate an integrating element which will reduce the steady-state error of the PLL.

Two of the blocks of the block diagram have to be added to the firmware to complete the PLL: the phase detector and the loop filter. The phase detector is implemented as an Alexander phase detector [25]. This detector has a topology which is very suitable to be implemented in the digital hardware of an FPGA. The Alexander phase detector has no linear relationship between the difference in phase and the output. The topology is a two state system, the phase detector only indicates if the VCO signal is leading or lagging. The gain of the phase detector is therefore very high for small differences in phase, which is useful to reduce the steady-state error of a PLL.

The implementation of the loop filter is less straightforward. This filter determines all loop dynamics. These include the stability, bandwidth, settling time and steady-state error. A lot of time and knowledge has to be invested to optimize this filter. During this master's assignment, this amount of time was not available. Instead, a crude but adaptable version of the filter has been implemented. The filter has been realized as a proportional controller of which the gain can be adjusted by the control software. The required integrating element to decrease the steady-state error has thus not been incorporated.

To partially compensate for the lack of an integrating element in the loop filter, the output of the loop filter will be averaged over about 20 seconds before the measurement starts. This averaging has an integrating effect and provides a decent output value for the DAC during the measurement. The averaging period of 20 second has been experimentally determined and provides a reasonable low steady-state error. However, a small steady-state error will always be present and it is essential that the loop filter is improved before the testbed will be used for actual measurements.

Combining all parts of the PLL is relatively straightforward, except for the communication between the FPGA and the DAC. This communication uses the serial peripheral interface (SPI) protocol. The original implementation of the SPI communication could not be used in the PLL, so a new open source module has been incorporated. The speed of the data communication between the FPGA and the DAC determines the upper limit of the update frequency of the PLL. This frequency has been set to 200 KHz, which is near its limit.

The PLL can be enabled and disabled by the control software and by other parts of the firmware. The PLL is controlled by the software to start the synchronization before the measurement, and the firmware will disable the synchronization while measuring. The entire measurement cycle will be elaborated in greater detail in Section 7.2.

6.4.2 Measurement start signal

The trigger to start the measurement is provided by a signal which is transmitted to all receivers. As was stated before, this signal will be transmitted by the 'global/MIMO master'. The expansion header of the BladeRFs will be used to transmit and receive this signal.

The receivers of the testbed will remain idle until they receive the measurement start signal. During this idle period, all receivers will discard all received data by disabling an internal buffer. This data has been received while the receivers where not yet properly synchronized, and is therefore of no interest for the testbed. In addition, the receivers with a PLL use this idle period to lock their PLLs to the reference clock.

When the receivers do receive the measurement start signal, they will not start the measurement immediately. Instead, the receivers will wait for 20 seconds. This delay is incorporated in all receivers of the testbed and is of equal length in all receivers. During this period the four receivers with a PLL will average the output value of the loop filter.

This averaging has been used as a crude substitute for a well-designed loop filter.

After the 20 second delay period, the four receivers with PLL will disable their PLL and will continue to use the averaged output value of the loop filter for the DAC. At this point, the MIMO master clocks will start to drift with respect to each other. At the same moment, all receivers of the testbed will enable their disabled internal buffer. Because all receivers re-enable this internal buffer at the same moment and because the clocks of all receivers are synchronized at the start of the measurement, the first output data of all receivers will be synchronized.

As stated before, during the measurement the clocks of the MIMO masters will slowly start to drift with respect to each other because the PLLs are disabled. Hence, the clocks used in each satellite mock-up will slowly start to drift compared to the other satellite mock-ups. This behavior corresponds nicely to that of the OLFAR satellites.

6.5 Conclusion

Out-of-the-box, the BladeRF x40 already met the specifications for the frequency range, the bandwidth, the high sample rate and the input impedance. It was already equipped with a stable oscillator and the hardware necessary for clock sharing. The capability of real-time processing was already integrated as well in the form of a fast processor and FPGA.

The capability of clock synchronization has been the only specification which was not satisfied out-of-the-box. By adapting the firmware of the device to implement the PLL and the measurement start signal this requirement has been met. However, the loop filter of the PLL which was used in the implementation is very primitive and should be replaced by a well-designed version. When the loop filter has been updated, the BladeRF x40 will satisfy all specifications and will be very suited as a receiver for the testbed.

Chapter 7

Software

The software is an important part of the testbed. The software is used for communication with the receivers, the control of the receivers and the data processing. The software is implemented in two parts: The communication between the computer and the receivers is implemented as a command line interface executable written in the language C. The second part is implemented entirely in MATLAB for easy compatibility with the algorithms which have to be tested. This latter part will control and set all parameters of the receivers, and it will perform all data processing. In this chapter, the functionality and implementation of the software is presented.

7.1 Communication software

The communication between the computer and the BladeRF is arranged by the program 'bladeRF-CLI.exe'. This is a command line interface program. The developers of the BladeRF have described how to use this program on their website and wiki [26]. The program itself includes a help function which provides information about all build-in functions and the required syntax to use those functions. The program is open source, this means that the developers enable the users to add extra functionality. In this section, the functionality which was added to the original program is described.

7.1.1 Added functionality

Four functions have been added to the communication software, which provide extra controls for the receivers. The extra functions are named: blink, dac, measurement and pll.

Function 'blink'

'Blink' has been implemented as a test function. It was used to verify the communication between the communication software and the SDR. One of the receivers is fitted with an XB-100 expansion board. This board adds extra buttons, LEDs and digital inputs and outputs. The board also includes a tri-color LED. This LED is controlled by the 'blink' code. In the code, the color of the LED and the number of times it will blink can be set. This function and the XB-100 expansion board are only used for debugging purposes, they are not required for the functionality of the testbed.

Function 'dac'

The function named 'dac' can be used to control the DAC of the BladeRF which is connected to the VCO. With this function a 16-bit value can be transmitted to the DAC. The DAC will output this value and thus change the frequency of the oscillator.

Function ‘measurement’

The ‘measurement’ function is intended for the SDR which is configured as the ‘global/MIMO master’. The function will command this BladeRF to transmit the signal to start the measurement. When this command has been given to the SDR, the SDR will wait for five seconds before it will transmit the measurement start signal to the other SDRs. This delay is incorporated to make sure that all other receivers have been configured and the PLLs are locked to the reference clock.

Function ‘pll’

The function ‘pll’ controls the PLL of the receivers. The function can be used to enable or disable the PLL and to change the properties of the loop filter. The current version of the firmware for the FPGA only uses a proportional controller, so only this parameter can be adjusted.

7.1.2 Implementation of the functions

All functions have been implemented in separate C code files. The code contains a lot of commentary which indicates the function of all parts of the code. This makes the code very comprehensible and therefore simple to adapt. The original ‘bladeRF-CLI.exe’ program includes help documentation which indicates the purpose of all functions and the required syntax to control these functions. This help documentation has been expanded to include the information regarding the four new functions.

7.2 Control and data processing

The whole testbed is controlled via MATLAB. Also, all data processing will be performed by MATLAB. By using one program for all controls and processing of the testbed, the whole system becomes more organized and simpler to adapt. Everything will be controlled from one main MATLAB script. To keep this script comprehensible, much of the functionality has been incorporated in functions which are stored in separate files. All MATLAB code contains commentary which indicates the functionality of the different parts of the code. In this section, first the main script of the control software is described, after which the functions used by this script are elaborated in more detail.

7.2.1 Main script

The main script of the control software of the testbed consists of multiple stages. The script will advance to the next stage when the previous stage has been completed for all receivers. The sequence of the stages in the main file is:

1. Create a clean workspace
2. Create the global variables and settings
3. Initialize and program all BladeRFs
4. Perform a measurement
    (a) Configure all BladeRFs
    (b) Store measurement settings
    (c) Prepare for sampling and start the measurement
    (d) Determine when the measurement is concluded

5. Perform the analysis
    (a) Data processing and algorithms under test

The first two stages are used for the initialization of the script. In these stages, the old variables will be deleted and the necessary global variables are defined.

The third stage is used to register which BladeRFs are connected, and to upload the firmware to all receivers.

During the fourth stage, the measurement will take place. In substage (a) all receivers will be configured and parameters like the center frequency, sample rate and gain will be set. Substage (b) is used to create a file where all metadata of the measurement is stored. This matadata contains all information about the connected BladeRFs and their settings, which can be used for future reference. The final configuration of the BladeRFs will be performed in substage (c). In this substage, all parameters regarding the sampling and data transfer are configured and the sampling process is started. Finally, substage (d) will determine when the sampling process is finished and the next stage can be performed.

Stage five is reserved for all data processing. In this stage, the algorithms which should be tested by the testbed will be implemented. Currently, this stage is only used to generate a plot of the acquired data to provide a demonstration of the functionality of the testbed.

All six stages of the script have been illustrated. Now that the functionality of these stages is clear, the implementation of this functionality will be described in the following sections.

7.2.2 Implemented functions

Most operations are performed by seven MATLAB functions. These functions are stored in separate files and are called by the main script. All functions possess a help text and use input argument validation to make the implementation of these functions straightforward and clear. The help text will indicate precisely how each function can be used. These help texts are provided in the manual of the testbed in section D.2.3 of the appendix. The operations performed by each of the created functions will be described below:

Function 'initialize'

This function will initialize all connected BladeRFs. During the initialization, the hardware information of all connected BladeRFs is stored in a variable. Also, the FPGAs of the BladeRFs will be loaded with the firmware and this process will be verified.

Function 'set'

With this function, all parameters of a BladeRF can be set. These include the standard parameters like center frequency, sample rate and gain, but it also selects the input source of the BladeRF. It can select to use the standard external input connector as the input for the BladeRF. Or, an internal counter of the receiver module on the BladeRF, which is very useful for testing purposes and is used by the 'verify_data_rate' function. If the external input is selected, the set function will calibrate the input of the BladeRF.

Function ‘print’

All parameters which have been set for a specific BladeRF can be retrieved from the SDR with this function. It will provide the parameters like the center frequency, sample rate and gain of the specific BladeRF, and can be used to verify if the BladeRF was configured properly.

Function ‘start_sampling’

The ‘start_sampling_function’ is used to set all parameters regarding the sampling and data transfer and the function will start the process. The function will open separate command line interfaces for each of the attached BladeRFs. These interfaces are used to transfer the sampling commands to the BladeRFs. After the interfaces have been opened, they will remain open during the entire measurement. The interfaces will sustain the USB connection and therefore enable the transfer of data from the BladeRFs to the computer. The windows of the interfaces are minimized and will only be visible in the taskbar of the computer.

The ‘start_sampling_function’ also commands one of the BladeRFs to transmit the measurement start signal. As described in Section 7.1.1, this BladeRF will transmit this signal after a 5 second delay. This delay has been incorporated to make sure that all BladeRFs are ready to receive the start measurement signal.

Function ‘import_data’

All measurement data has been stored in a folder by the command line program. Within this folder, each receiver has stored its data in a different comma separated file. This function will import the data from each of those files and will convert this data to a MATLAB data structure. The process of reading these files and importing the data has been highly optimized for speed.

Function ‘plot_data’

The ‘plot_data’ function will plot the data from a single receiver. The function will use the ‘import_data’ function to import all data. It will also use the stored metadata of the measurement to determine the settings of the BladeRF during the measurement. The function will plot the data in the time and the frequency domains. In the time domain, the measured voltage and corresponding power will be displayed for both the I and the Q channels. In the frequency domain, the plots will display the input power in watts and dBm. The function will also calculate and display the maximum power which was received in a small bandwidth, this is used for testing purposes to display the power of an incoming unmodulated carrier.

Function ‘verify_data_rate’

This function is used to verify if the computer is able to handle the data output of the testbed via the USB connections. For this function, the testbed needs to perform a measurement with the internal counter as input for all BladeRFs. This measurement results in a known output for all BladeRFs. This function will then compare the results of this measurement with the desired outcome if no USB packages would have been lost. If no packages have been lost during this measurement, the computer has proved to be able to cope with this data rate.

7.3 Conclusion

The created software provides a complete package to control the testbed. The commands which where needed to control the added firmware functions of the BladeRFs have been added to the communication software. Also, seven MATLAB functions have been written to provide all necessary control functions and data processing capabilities.

By controlling the entire testbed from a single MATLAB script, the testbed has become very flexible and simple to adapt. All changes to the actions and parameters of the testbed can be made at a single location, which should make the control of the testbed very organized and comprehensible.

Chapter 8

Physical construction

Four of the five main components of the testbed have been described in the previous chapters. Some research has also been devoted to the fifth element of the testbed: the physical construction of the testbed.

The specifications of OLFAR which are of importance for the testbed have been provided in Appendix A. When comparing this list of specifications with the specifications used by the four components described in the previous chapters, it becomes clear that some have not been addressed. The missing specifications are some spatial and some receiver specifications, but they all relate to spatial stability and accuracy. As will be shown in this chapter, the physical construction of the testbed is of importance to ensure the accuracy and repeatability of the measurements performed with the testbed.

The construction is also of importance for the user-friendliness of the testbed. It will largely determine how much effort the user has to invest in each measurement. For example, if the observational antenna systems are placed at fixed or known locations, the user will not have to measure all their locations before each measurement. Another example is the relocation of the satellite mock-ups or the astronomical source simulator between subsequent measurements. This movement could be automated, which will spare the user a lot of time and effort between the measurements.

In this chapter, the specifications for the physicals construction will be provided and elaborated. In addition, the research performed to meet these specifications and a draft concept will be presented.

8.1 Specifications

The specifications which have not been addressed in the previous chapters are:

- Baseline determination accuracy of $\lambda/10$
- Baseline stability smaller than $\lambda/10$ during the measurement
- Antenna orientation accuracy of 1°
- Antenna spinning rate smaller than 1°/s

As mentioned, these specifications are all connected to the physical construction of the testbed. Because of the chosen center frequency of 1283 MHz, the baseline accuracy and stability must be smaller than 2.4 cm. This distance is the maximum error which is possible while still maintaining the required accuracy of the testbed.

However, this error is not only determined by the inaccuracy in the location of the satellites. The timing inaccuracies in the testbed will also contribute to this error. This is due to the fact that light has a certain speed. For example, if the clock of one of the receivers is lagging by a picosecond, the data gathered by this receiver would seem to have been sampled 0.3 mm closer to the source than it actually was.

8.2 Construction

During this master's assignment, not enough time was available to design the physical construction. However, some research has performed which can provide a useful basis for future work.

8.2.1 Accuracy and stability

As described in the specifications, the construction should be very accurate and stable. This accuracy is necessary to enable exact measurements and the stability is necessary for repeatability of these measurements. The positional information of the antenna systems can be used as a reference to be compared with the outcome of the calibration algorithms. This comparison is also only possible if the accuracy of the testbed is adequate.

8.2.2 Satellite constellations and movement

The testbed would be much more flexible if the construction supports multiple different satellite constellations. In OLFAR, all satellites will be moving slowly with respect to each other. The constellation of the satellites is therefore constantly changing and each constellation will produce different measurement results. In the testbed, this can be incorporated by making the positions of the satellites adaptable. The largest amount of flexibility is obviously obtained when the positions of the satellites can be adjusted in all six degrees of freedom: X, Y, Z, pitch, yaw and roll. However, not all these degrees of freedom have to be incorporated to obtain insightful results. The movements of the satellites of OLFAR will be so slow that the satellites are assumed stationary during the measurements, the testbed therefore only has to incorporate the movement of the satellites between the measurements and not during the measurements.

Between measurements, the observational antenna systems or the astronomical source simulator will be relocated. This will simulate the movement of the astronomical source compared to the satellite constellation. For the performance of the testbed it is best to relocate the antenna systems, instead of the source. This is due to the possible presence of RFI sources. If the antenna systems are relocated, the source simulator and the source of the RFI will remain stationary compared to each other. These will therefore show up as two static sources in the measurement results.

A second possibility is to relocate the source simulator instead of the antenna systems. In this case, the RFI source will move with respect to the source simulator. In the measurement results, this still results in a static source caused by the source simulator. But, the influence of the RFI source will be smeared out and it will taint a large part of the field of view.

8.2.3 Materials

The construction of the testbed should be accurate, flexible and adaptable. Therefore, metal seems like a logical choice for the construction material. However, this is not the case for the testbed. This is because the antennas should be placed as far away as possible from any conductors. These conductors would otherwise influence the antenna pattern just like the feed lines as has been described in Chapter 5. A lot of effort was put into the reduction of the influence of these feed lines on the antenna pattern. This would also be necessary for the construction if it would be build from conductive materials.

8.3 Concept

During the assignment, a concept for the construction has been created. This concept applies many of the above described aspects. A 3D model of the concept can be found in Appendix E. However, this concept is the only draft design that has been developed during this assignment for the construction of the testbed. It is therefore recommended that additional designs will be created such that the most suitable concept can be selected.

Chapter 9

Conclusion

This chapter provides the conclusions of all work performed during the master's assignment. The chapter will provide the answer on the research question: "How to create a realistic testbed for the calibration and RFI mitigation algorithms used in the interferometer OLFAR with its arbitrary three-dimensional antenna distribution?". The conclusion will be split into two sections: The individual results of each of the components of the testbed and the results of the testbed as a whole. This chapter will finish with a listing of essential and nonessential recommendations for future work.

9.1 Individual results

The five functional components of the testbed have largely been designed separately. Therefore, each part had a separate evaluation of the results of the design. The conclusions of these evaluations will be provided below.

9.1.1 Astronomical source simulator

The astronomical source simulator is capable of generating and transmitting the signal required for operation of the testbed. The source simulator has met all specifications which where stated in Chapter 4. However, the source will transmit circularly polarized signals instead of unpolarized signals. Therefore, the source is not suitable to verify algorithms which calculate the polarization of the source. In addition, the quality of the transmitted signal remains unknown because the axial ratio of the antenna has not been measured.

9.1.2 Observational antenna system

The design of the observational antenna system has resulted in a satisfactory model. This model meets all stated mechanical requirements and, in simulation, also meets the electrical specifications. The simulated antenna pattern only deviates 3% from the ideal dipole-like antenna pattern. In addition, due to the use of PCBs and standard PVC pipes as construction materials, the antenna system should be straightforward to construct. Due to time constraints however, a prototype of the antenna system has yet to be created and measured.

9.1.3 Receivers

The Nuand BladeRF x40 has been selected as the receiver for the testbed. By using this feature-rich COTS receiver, almost all of the required specifications where met by the device out-of-the-box. One specification was not met out-of-the-box: the capability of clock synchronization. This functionality has been incorporated by adapting the firmware of the BladeRF. However, the loop filter in the PLL which has been added to the firmware by this adaptation provides inadequate results. Therefore, the specification for the capability of clock synchronization has been deemed unfulfilled until this loop filter has been improved.

9.1.4 Software

The software is used to control the entire testbed, and it will perform all data processing. The software that has been created is fully capable of performing these operations. The software has been created in C and MATLAB code. Functions have been created to command the added functionality of the SDRs, and comprehensive code has been written to control the testbed. The software has become flexible and simple to adapt. The algorithms for calibration and RFI mitigation, which will be verified by the testbed, can be effortlessly inserted in the MATLAB code of the software.

9.1.5 Physical construction

The specifications for the construction where reviewed, and recommendations for the physical construction have been provided. No complete design for the construction has been created due to time constraints. However, a draft design was developed and has been presented.

9.2 Combined result

For four of the components of the testbed a design has been presented. When combined, these four components will result in a functional testbed. With an adequate physical construction, this testbed can serve as a representation of actual OLFAR-hardware and should be able to conduct accurate and repeatable measurements. The testbed is very flexible and adaptations to the parameters and processing can be made with ease. Thus, the testbed can easily be reconfigured to the specific needs of each measurement.

Provided that the loop filter of the PLL has been improved and a suitable construction has been created, the testbed will be able to adequately represent all relevant specifications of OLFAR as stated in Appendix A. Therefore, the testbed is expected to be suitable for the verification of the calibration and RFI mitigation algorithms used in OLFAR.

9.3 Recommendations

The design of the testbed has almost been completed. Only the design of the physical construction has to be finished. On the other hand, not enough time was available to construct many of the parts during the master's assignment. Therefore, the fabrication and testing of these parts is future work. A list of essential items for future work is provided below:

- Measure the axial ratio of the antenna of the astronomical source simulator
- Build a prototype of the observational antenna system and verify its performance
- Construct 5 observational antenna systems
- Replace the loop filter of the PLL with an improved version
- Complete the testbed by adding 12 BladeRFs
- Design and build the physical construction of the testbed

Optional recommendations for future work have been conceived as well. These are not required to be implemented, but they will improve the testbed:

- Add a second antenna and signal generator to the astronomical source simulator to generate actual unpolarized signals, or any other form of polarization.
- Implement cheaper signal generators for the astronomical source simulator to decrease the costs of the testbed, and to reduce the financial risks involved when using the testbed outdoors.
- Automate the movement of the satellite constellation between subsequent measurements.

Bibliography

[1] R. Rajan, S. Engelen, M. Bentum, and C. Verhoeven, "Orbiting low frequency array for radio astronomy," in *Aerospace Conference, 2011 IEEE*, March 2011.

[2] S. V. Engelen *et al.*, "The road to OLFAR - a roadmap to interferometric long-wavelength radio astronomy using miniaturized distributed space systems," in *Proceeding of the 64th IAC International Astronautical Congress, Beijing, China*. Beijing, China: IAF, August 2013, pp. 1–7.

[3] J. D. Kraus, *Radio astronomy*, 2nd ed. Cygnus-Quasar Books Powell, Ohio, 1986.

[4] T. L. Wilson, K. Rohlfs, and S. Hüttemeister, *Tools of Radio Astronomy*. Springer Berlin Heidelberg, 2009.

[5] A. R. Thompson, J. M. Moran, and G. W. Swenson Jr, *Interferometry and synthesis in radio astronomy*. John Wiley & Sons, Inc., 1998.

[6] S. J. Wijnholds, *Fish-eye observing with phased array radio telescopes*. S.J. Wijnholds, 2010.

[7] A. Leshem, A.-J. van der Veen, and A.-J. Boonstra, "Multichannel interference mitigation techniques in radio astronomy," *The Astrophysical Journal Supplement Series*, vol. 131, no. 1, p. 355, 2000. [Online]. Available: http://stacks.iop.org/0067-0049/131/i=1/a=355

[8] J. E. Noordam, "LOFAR calibration challenges," in *SPIE 5489 Proceedings, Ground-based Telescopes*, vol. 5489, September 2004, pp. 817–825. [Online]. Available: http://dx.doi.org/10.1117/12.544262

[9] K. van de Schaaf and R. Nijboer, "LOFAR calibration implementation global design of the major cycle," ASTRON, Tech. Rep. 2.0, March 2007, LOFAR-ASTRON-SDD-050. [Online]. Available: http://www.lofar.org/wiki/lib/exe/fetch.php?media=public:documents:23_lofar_calibration_implementation.pdf

[10] A.-J. Boonstra, S. Wijnholds, S. van Der Tol, and B. Jeffs, "Calibration, sensitivity and rfi mitigation requirements for LOFAR," in *Acoustics, Speech, and Signal Processing, 2005. Proceedings. (ICASSP '05). IEEE International Conference on*, vol. 5, March 2005, pp. 869–872.

[11] P. A. Fridman and W. A. Baan, "RFI mitigation methods in radio astronomy," *Astronomy & Astrophysics*, vol. 378, no. 1, pp. 327–344, October 2001. [Online]. Available: http://dx.doi.org/10.1051/0004-6361:20011166

[12] OLFAR & DCIS design team, "OLFAR strawman design report," ASTRON, Tech. Rep. 0.4, January 2013.

[13] Electronic Communications Committee (ECC), "A european table of frequency allocations and applications for the frequency range 8.3 khz to 3000 ghz (ECA Table)," European Conference of Postal and Telecommunications Administrations (CEPT), ERC REPORT 25, May 2014.

[14] VERON, *Vademecum*, 15th ed. Stichting Servicebureau VERON, May 2010.

[15] ARRL, *The ARRL UHF/microwave experimenter's manual*. The American Radio Relay League Inc, 1993.

[16] C. Balanis, *Modern Antenna Handbook*. Wiley, 2008.

[17] ARRL, *The ARRL antenna book*. The ARRL Inc, 2007.

[18] J. Wong and H. King, "Broadband quasi-taper helical antennas," *Antennas and Propagation, IEEE Transactions on*, vol. 27, no. 1, pp. 72–78, January 1979.

[19] M. R. Andrews, P. P. Mitra, and R. deCarvalho, "Tripling the capacity of wireless communications using electromagnetic polarization," *Nature*, no. 409, pp. 316–318, January 2001. [Online]. Available: http://dx.doi.org/10.1038/35053015

[20] M. Klein Wolt, A. Aminaei, P. Zarka, J.-R. Schrader, A.-J. Boonstra, and H. Falcke, "Radio astronomy with the European Lunar Lander: Opening up the last unexplored frequency regime," *Planetary Space Science*, vol. 74, pp. 167–178, December 2012. [Online]. Available: http://arxiv.org/pdf/1209.3033v1.pdf

[21] J. Galluppi, "Field probe," November 2012, US Patent 8,305,282. [Online]. Available: http://www.google.com/patents/US8305282

[22] A. Voors, "4nec2 antenna modeler and optimizer 5.8.14," December 2013. [Online]. Available: http://www.qsl.net/4nec2/

[23] The ITU Radiocommunication Assembly, "Electrical characteristics of the surface of the earth," International Telecommunication Union Radiocommunication Sector (ITU-R), RECOMMENDATION P.527-3, March 1992.

[24] J. Gaither, "Digital phase-locked loop (dpll) reference design," XILINX, Tech. Rep. XAPP854, October 2006. [Online]. Available: http://www.xilinx.com/support/documentation/application_notes/xapp854.pdf

[25] B. Razavi, "Challenges in the design high-speed clock and data recovery circuits," *Communications Magazine, IEEE*, vol. 40, no. 8, pp. 94–101, Aug 2002.

[26] "BladeRF wiki," November 2014. [Online]. Available: https://github.com/Nuand/bladeRF/wiki

[27] R. Rajan, M. Bentum, and A.-J. Boonstra, "Synchronization for space based ultra low frequency interferometry," in *Aerospace Conference, 2013 IEEE*, March 2013, pp. 1–8.

[28] A. Budianu, A. Meijerink, M. J. Bentum, D. M. Smith, and A.-J. Boonstra, "Antenna architecture of a nano-satellite for radio astronomy," in *Aerospace Conference, 2014 IEEE*, March 2014, pp. 1–10.

Appendix A

Specifications of OLFAR

Table A.1: The relevant specifications of OLFAR for the testbed

Property	**Specification OLFAR**
Spectral specifications	
Frequency range	0.3–30 MHz [12]
Instantaneous bandwidth	10 MHz [12]
Spectral resolution	1 kHz [12]
Timing specifications	
Snapshot integration time	$\geq$ 1 s [12]
Allan deviation (1 s integration time)	$\leq 10^{-8}$ [27]
Deployment specifications	
Deployment location	Earth-Moon Lagrangian 2 point [2]
Deployment configuration	3D Lissajous swarm [12]
Spatial specifications	
Number of satellites	50 (minimum $\geq$ 5) [12]
Baseline length	≤ 100 km [12]
Baseline stability during snapshot	$\lambda/10$ m [12]
Baseline determination accuracy	$\leq \lambda/10$ m [12]
Spatial resolution	Diffraction limited [12]
Aperture filling distribution, snapshot	3D even distribution [12]
Aperture filling distribution, integrated	3D even distribution [12]
Receiver specifications	
Receiver sensitivity	Sky-noise limited [12]
Number of receivers per node	3 [12]
Antenna concept	Active short antennas [28]
Antenna length	9.6 m [28]
Antenna orientation	Arbitrary, but known within $^1/_{60}$ rad [12]
Antenna spinning rate	$\leq {}^1/_{60}$ rad/s [12]
Signal processing specifications	
Imaging procedure	Aperture synthesis [12]
Imaging signal processing	Correlations followed by spatial Fourier transforms [12]
Delay compensation	No [12]
Phase compensation	Yes, off-line for all-sky images [12]
Real-time processing bandwidth	$\geq$ 1 MHz [12]
Required No. ADC bits	$\geq$ 1 bit (dependent on RFI levels) [12]
No. signal bits at correlator	$\geq$ 1 bit (after RFI mitigation) [12]

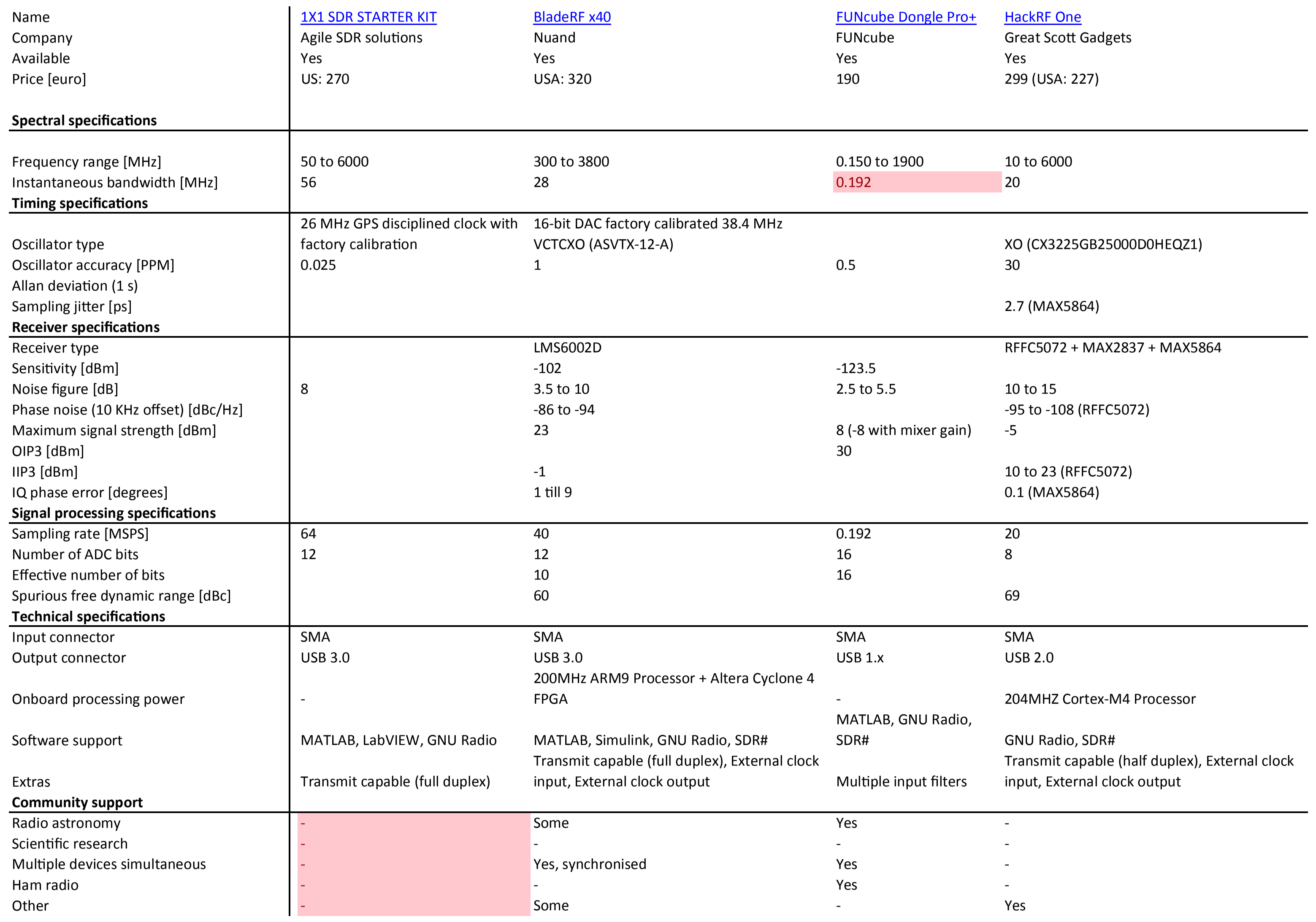
Appendix B
Antenna simulations

```
CM One dipole in free space
CE
SY cos_30=cos(30)    'The outcome of cos(30)
SY sin_30=sin(30)    'The outcome of sin(30)
SY cos_35=cos(35)    'The outcome of cos(35.3)
SY sin_35=sin(35)    'The outcome of sin(35.3)
SY pole_length=0.055    'Length of one pole of the dipole
SY height=1    'Height of the antenna
SY offset_feed=0.000    'Distance between the feed and the centers of the antennas
SY offset=0.010    'Distance from the center of the antennes to the feedpoint
GW    1    35    -cos_35*pole_length    -offset    -sin_35*pole_length+height    cos_35*pole_length
    -offset    sin_35*pole_length+height    0.0008    'Antenna 1 (active)
GE    0
LD    5    1    0    0    58000000    'Antenna 1
GN    -1
EK
EX    0    1    18    0    1    0    0    'Voltage source (1+j0) at wire 1 segment 5.
FR    0    0    0    0    1283    0
RP    0    61    121    1003    -180    0    3    3
EN
```

Figure B.1: The NEC code for the simulation of a single dipole in free space

Appendix C

Receiver comparison



Name	1X1 SDR STARTER KIT	BladeRF x40	FUNcube Dongle Pro+	HackRF One
Company	Agile SDR solutions	Nuand	FUNcube	Great Scott Gadgets
Available	Yes	Yes	Yes	Yes
Price [euro]	US: 270	USA: 320	190	299 (USA: 227)
Spectral specifications				
Frequency range [MHz]	50 to 6000	300 to 3800	0.150 to 1900	10 to 6000
Instantaneous bandwidth [MHz]	56	28	0.192	20
Timing specifications				
Oscillator type	26 MHz GPS disciplined clock with factory calibration	16-bit DAC factory calibrated 38.4 MHz VCTCXO (ASVTX-12-A)		XO (CX3225GB25000D0HEQZ1)
Oscillator accuracy [PPM]	0.025	1	0.5	30
Allan deviation (1 s)				
Sampling jitter [ps]				2.7 (MAX5864)
Receiver specifications				
Receiver type		LMS6002D		RFFC5072 + MAX2837 + MAX5864
Sensitivity [dBm]		-102	-123.5	
Noise figure [dB]	8	3.5 to 10	2.5 to 5.5	10 to 15
Phase noise (10 KHz offset) [dBc/Hz]		-86 to -94		-95 to -108 (RFFC5072)
Maximum signal strength [dBm]		23	8 (-8 with mixer gain)	-5
OIP3 [dBm]			30	
IIP3 [dBm]		-1		10 to 23 (RFFC5072)
IQ phase error [degrees]		1 till 9		0.1 (MAX5864)
Signal processing specifications				
Sampling rate [MSPS]	64	40	0.192	20
Number of ADC bits	12	12	16	8
Effective number of bits		10	16	
Spurious free dynamic range [dBc]		60		69
Technical specifications				
Input connector	SMA	SMA	SMA	SMA
Output connector	USB 3.0	USB 3.0	USB 1.x	USB 2.0
Onboard processing power	-	200MHz ARM9 Processor + Altera Cyclone 4 FPGA	-	204MHZ Cortex-M4 Processor
Software support	MATLAB, LabVIEW, GNU Radio	MATLAB, Simulink, GNU Radio, SDR#	MATLAB, GNU Radio, SDR#	GNU Radio, SDR#
Extras	Transmit capable (full duplex)	Transmit capable (full duplex), External clock input, External clock output	Multiple input filters	Transmit capable (half duplex), External clock input, External clock output
Community support				
Radio astronomy	-	Some	Yes	-
Scientific research	-	-	-	-
Multiple devices simultaneous	-	Yes, synchronised	Yes	-
Ham radio	-	-	Yes	-
Other	-	Some	-	Yes

Figure C.1: Receiver comparison part 1

Name	NooElec NESDR	SDR Play	USRP-2921	USRP B200
Company	NooElec	SDRplay	National Instruments	Ettus research
Available	Yes	Yes	Yes	Yes
Price [euro]	14	220	2430	600
Spectral specifications				
Frequency range [MHz]	25 to 1750	0.100 to 380 and 430 to 2000	2.4 to 2.5 and 4.9 to 5.9	70 to 6000
Instantaneous bandwidth [MHz]	2.4	8	36 (8 bit) or 19 (16 bit)	56
Timing specifications				
Oscillator type	XO 28.8 MHz	XO (NX2520SA)	TCXO	TCXO
Oscillator accuracy [PPM]		15	2.5	2
Allan deviation (1 s)				
Sampling jitter [ps]				
Receiver specifications				
Receiver type	R820T + RTL2832U	MSI001 + MSi2500		AD9361
Sensitivity [dBm]				
Noise figure [dB]	3.5 (R820T)	3.5 to 12.5	5 to 7	8
Phase noise (10 KHz offset) [dBc/Hz]	-98	-100		
Maximum signal strength [dBm]	10		0	2.5 (AD9361)
OIP3 [dBm]				
IIP3 [dBm]	35	-15		-20
IQ phase error [degrees]				1.5
Signal processing specifications				
Sampling rate [MSPS]	2.4	12	50 (8 bit) or 25 (16 bit)	61.44
Number of ADC bits	8	10	14	12
Effective number of bits		9.5	8 bit (36 Mhz) or 16 (19 MHz)	
Spurious free dynamic range [dBc]			88	78
Technical specifications				
Input connector	MCX	F	SMA	SMA
Output connector	USB	USB	Gigabit Ethernet	USB 3.0
Onboard processing power	-	-	-	200MHz ARM9 Processor + Spartan 6 XC6SLX75 FPGA
Software support	MATLAB, GNU Radio, SDR#	SDR#	LabVIEW, GNU Radio	GNU Radio
Extras	-	Multiple input filters	Transmit capable (half duplex), External clock input, Pulse per second input	Transmit capable (full duplex), External clock input, Pulse per second input
Community support				
Radio astronomy	Yes	-	-	-
Scientific research	-	-	Yes	-
Multiple devices simultaneous	Yes	-	Yes	Yes
Ham radio	Yes	-	-	-
Other	Yes	-	-	Some

Figure C.2: Receiver comparison part 2

Appendix D

Manual

This chapter will provide more practical information about the testbed. The first section of this chapter is meant as a manual to demonstrate the use of the testbed. This demonstration is used to display the functionality of the testbed and the software, it can be used for verification of the workings of the testbed. In addition, the remainder of this chapter will provide more insight in the software functions of the testbed and it provides a helping hand if the user wishes to expand the testbed.

This chapter will not provide a guide for novices on how to install the required software for general use of the testbed. These users are referred to the abundant amount of information available on the website of the developers of the BladeRF: `http://nuand.com/windows.php`.

Throughout this chapter, the text "[PATH]" is used to indicate the main directory of the DVD which is provided with this thesis. This DVD will contain all files and data which has been gathered or created during this master's assignment.

D.1 Demonstration

To verify the performance of the receivers and software of the testbed a small demonstration has been devised. The goal of this demonstration is to show that the improved firmware of the receivers and the created software are functional. In this section, first the setup of the demonstration will be described such that the demonstration can be performed by other users. Next, it will be explained what will happen during the demonstration. And finally, the results of the demonstration will be interpreted to indicate the conclusion of the demonstration.

D.1.1 Setup

For this demonstration three BladeRFs are available, these are named: "BladeRF_00", "BladeRF_01" and "BladeRF_02". Each of these BladeRFs will be configured as one of the three receiver configurations as where mentioned in Section 6.3.1. "BladeRF_00" will be configured as "Global/MIMO master", "BladeRF_01" will be configured as "MIMO slave" and "BladeRF_02" will be configured as "MIMO master with PLL". The part of the testbed which is thereby constructed can be seen in Figure D.1.

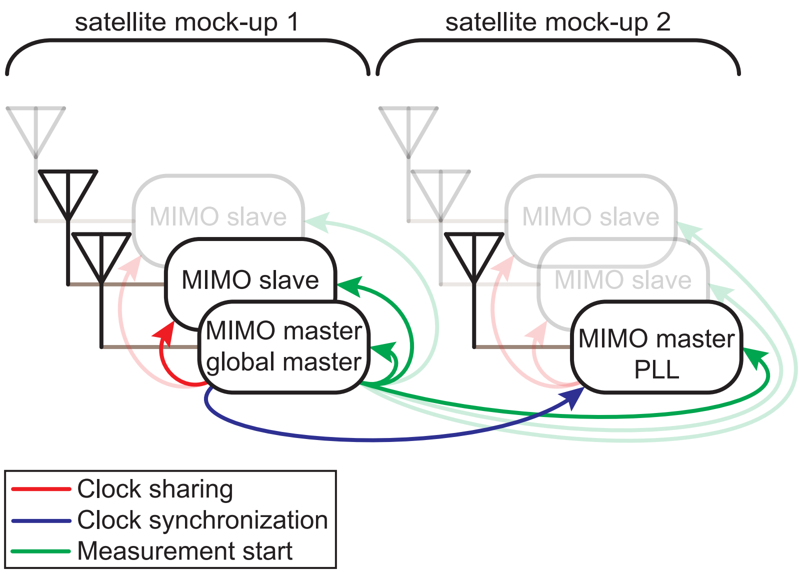


Figure D.1: The clock distribution of the demonstration

This figure also displays how the clock distribution of the demonstration should be arranged. Next to this clock distribution, the demonstration also requires a RF input signal for the receivers and a oscilloscope to verify the clock distribution. The entire connection diagram of the demonstration is provided in Figure D.2.

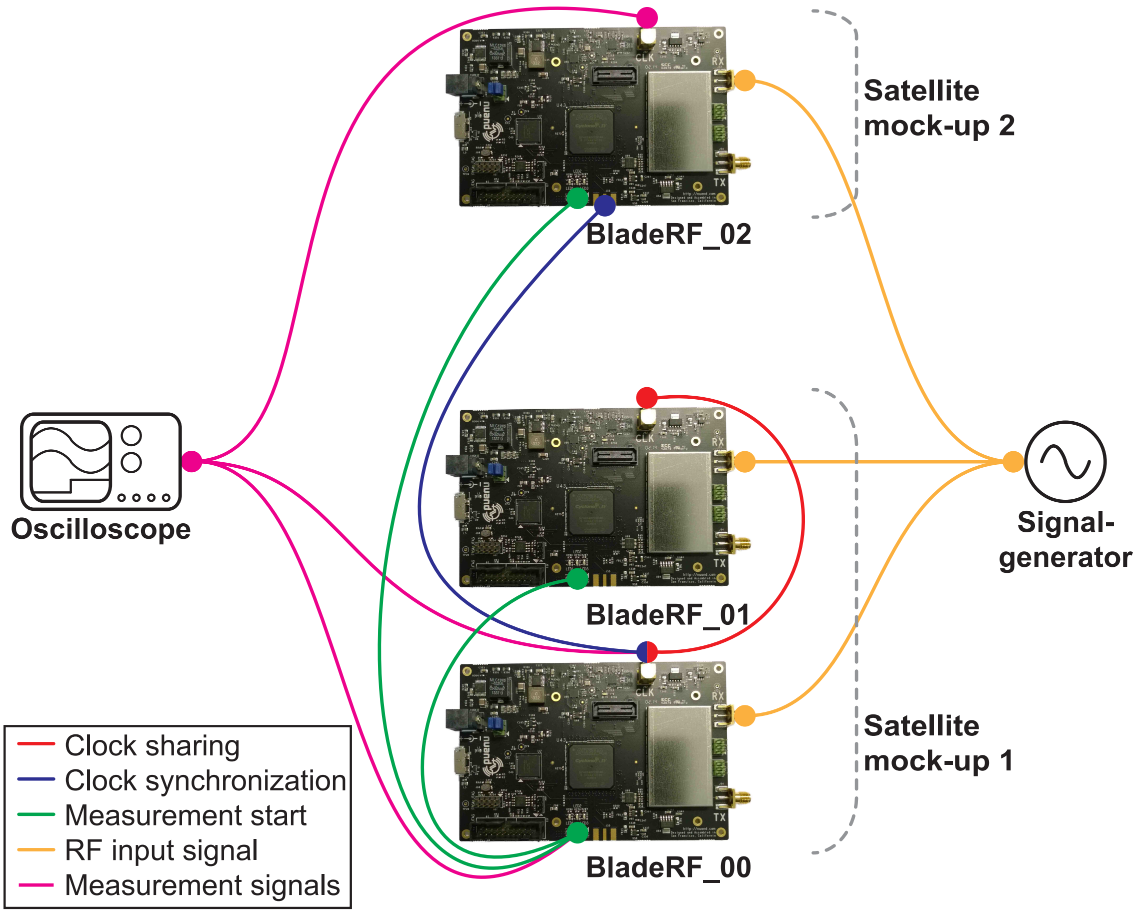


Figure D.2: Connection diagram of the demonstration

All connections of the connection diagram can be made with general purpose coaxial cables, with as exception the measurement start cable indicated by the green line. This cable should be designed to fit the expansion headers of the receivers. For the demonstration, a cable has been created and is provided with the other parts necessary for the demonstration. This cable has specific connectors which will only fit the correct receiver.

After all connections have been made, the signal generator and the oscilloscope can be setup. The signal generator needs to be set to generate a 1 MHz wide noise signal with a center frequency of 1283 MHz. The output power of the generator should be set to -30 dBm. The oscilloscope should be set to capture the clock signals of the receivers. These clock signals have a frequency of 38.4 MHz and an amplitude of 1.8 V_{PP}. The required settings for the oscilloscope to capture these signals is dependent on the type of oscilloscope that is used.

D.1.2 Measurement

The demonstration is started by running the "BladeRF_main.m" file with MATLAB. This file can be found in the folder "[PATH]\Matlab".

The control of the demonstration is fully automated. The demonstration will display the same step-by-step behavior as the full testbed will, when performing a measurement. The following 8 stages will be performed during the demonstration:

1. Create a clean workspace
2. Create the global variables and settings
3. Initialize and program all BladeRFs
4. Configure all BladeRFs
5. Store measurement settings
6. Prepare for sampling and start the measurement
   (a) Set all sampling parameters of BladeRF_00, start its data acquisition and send the measurement start trigger to the BladeRF
   (b) BladeRF_00 will start to wait for five seconds before it will transmit the measurement start signal
   (c) Set all sampling parameters of BladeRF_01 and BladeRF_02 and start their data acquisition
   (d) All three BladeRFs are acquiring, but they discard all data
   (e) The 5 second delay for BladeRF_00 is over and the start measurement signal is transmitted via the mini_exp1 pin
   (f) All three BladeRFs receive the start measurement signal at the same moment via the mini_exp2 pin
   (g) BladeRF_00 and BladeRF_01 will remain idle for 20 seconds, but BladeRF_02 will average the output of its PLL during these 20 seconds
   (h) After the 20 second delay, BladeRF_02 will disable its PLL. Furthermore, all three BladeRFs will stop discarding their input data and start to stream this data to the computer
   (i) The data streams are stored on the computer
7. Determine when the measurement is concluded
8. Process the data and create plots

The first two stages will be performed very quickly and unnoticeable by MATLAB. The third stage will make the LEDs on all three BladeRFs blink, this indicates that the BladeRFs have been programed. In stage four the LEDs will blink again to indicate the communication between the computer and the BladeRFs. Stage five will be performed unnoticeable.

In stage six, the functional part of the demonstration will take place. The steps which are performed here are described in the enumeration above. During this process the LEDs on the BladeRFs will blink continuously because they will remain in constant connection with the computer. During this stage, the oscilloscope is used to verify the workings of the testbed. On the oscilloscope it should be visible that the PLL of BladeRF_02 is functioning, and therefore that both clock signals are phase-locked. The phase difference between the clocks is dependent on the length of the cable used for the clock synchronization. When the demonstration has reached step (h), the PLL is disabled and the clock signals visible on the scope should start to drift compared to each other. The moment when this should happen can be found with the third channel of the oscilloscope, which measures the measurement start signal.

Stage seven will check if the measurement is finished. After which stage eight will be performed where the acquired data is processed. In this stage time domain plots will be created from the acquired data. In these plots it should be visible that the acquired data is the same for all three BladeRFs at the start of the measurement. When time passes the data from BladeRF_00 and BladeRF_01 should continue to be the same, as they use the same clock and sample at the same time. The data from BladeRF_02 should slowly deviate from the data of the other two BladeRFs, this is due to the difference in phase between the clocks. Due to this difference, BladeRF_02 will sample at different moments and therefore gather different data compared to the other two BladeRFs.

D.1.3 Outcome

As described above, the demonstration will provide two outputs, the measurement with the oscilloscope and the graphs on the computer. Both should indicate that the clocks are synchronized before the measurement, and start to drift during the measurement. This proves that the added firmware and software functions of the testbed are working. It indirectly also indicates that the out-of-the-box functionality of the BladeRFs is working. However, this of course expected for COTS hardware.

D.2 Software functions

In this part of the manual, more insight in the used software is provided. The software consists of three levels: the low-level firmware of the receivers, the communication software and the high-level MATLAB code. Each of the three levels will be described separately.

D.2.1 Firmware

The lowest level of the software is the firmware of the BladeRFs. This firmware is written in VHDL. For the testbed the original firmware of the receivers has been adapted. The original firmware of the BladeRFs is constantly updated and is provided on the GitHub of the developers at: `https://github.com/Nuand/bladeRF/tree/master/hdl`.

This firmware has been adapted to incorporate the extra features which where described in Chapter 6 of this thesis. All changes to the firmware have been made in plain text, and no software with graphical interface has been used. Therefore, the difference between the original firmware and the altered version can easily be examined by using a side-by-side comparison. This comparison can be performed with the software of GitHub, or the “compare” plug-in of notepad++ for example.

IP cores

For the firmware, additional IP cores have been downloaded or created. These IP cores can be found in the directory: “[PATH]\Receivers\BladeRF\Github\Testbed_bladeRF\hdl\fpga\ip\other”. In this directory, three newly added IP cores can be found.

The code in the “debounce” core is used to debounce and synchronize the input signals from the buttons of the XB-100 expansion board. The phase detector which is used by the PLL is described in the core “phase_detector”. The code which is necessary for the SPI communication between the FPGA and the DAC is provided in the core “spi_master(rtl)”.

Top level design

The top level of the design is provided in the file: “[PATH]\Receivers\BladeRF\Github\Testbed_bladeRF\hdl\fpga\platforms\vhdl\bladerf-hosted_testbed.vhd”. Numerous small and large changes have been made to the original code of this file. The large changes have been incorporated in so-called ‘processes’, these contain the large functions of the top level design.

Four processes have been added to the top level design to incorporate the necessary functionality of the testbed. The reception of commands from the communication software on the computer is performed by the code in the “PC_Communication” process. The functionality of the PLL is incorporated in the process “external_pll”. This PLL communicates with the DAC, this communication is controlled by the “dac_program” process. The process “blink” has been used for testing purposes, and is used to control the tri-color led on the XB-100 expansion board.

D.2.2 Communication software

The communication software will be running on the computer and is used as the connection between the low-level firmware and the high-level MATLAB code. This code is written in the C programming language. All software created by the developers for the communication software can be found on GitHub at: `https://github.com/Nuand/bladeRF/tree/master/host`.

The most interesting part of the software is the command line interface program used for the communication between the computer and the BladeRF: “bladeRF-cli”. The source code of this program can be found in a subsection of the above mentioned GitHub. It is available at: `https://github.com/Nuand/bladeRF/tree/master/host/utilities/bladeRF-cli`.

Four functions have been added to this program, these are: “blink”, “dac”, “measurement” and “pll”. These four functions are used to control the new functionality of the

BladeRF firmware, which has been described in the previous section. The communication software has a build-in help. This help provides the exact purpose of all functions together with the required syntax to use these functions. The four functions which where added to the library are also listed in this help. The help can be reached by using the command line program with the "-h" option. The information provided by this help can be seen in Figure D.3.

```
[PATH]\Receivers\BladeRF\Github\Testbed_bladeRF\host\build\output\Debug>bladeRF-
cli.exe -h

Usage: bladeRF-cli.exe <options>
bladeRF command line interface and test utility (0.12.0-git)

Options:
  -d, --device <device>          Use the specified bladeRF device.
  -f, --flash-firmware <file>    Write the provided FX3 firmware file to flash.
  -l, --load-fpga <file>         Load the provided FPGA bitstream.
  -L, --flash-fpga <file>        Write the provided FPGA image to flash for
                                 autoloading. Use -L X or --flash-fpga X to
                                 disable FPGA autoloading.
  -p, --probe                    Probe for devices, print results, then exit.
  -e, --exec <command>           Execute the specified interactive mode command.
                                 Multiple -e flags may be specified. The
                                 commands will be executed in the provided order.
  -s, --script <file>            Run provided script.
  -i, --interactive              Enter interactive mode.
      --lib-version              Print libbladeRF version and exit.
  -v, --verbosity <level>        Set the libbladeRF verbosity level.
                                 Levels, listed in increasing verbosity, are:
                                 critical, error, warning,
                                 info, debug, verbose
      --version                  Print CLI version and exit.
  -h, --help                     Show this help text.
      --help-interactive         Print help information for all interactive
                                 commands.

Notes:
  The -d option takes a device specifier string. See the bladerf_open()
  documentation for more information about the format of this string.

  If the -d parameter is not provided, the first available device
  will be used for the provided command, or will be opened prior
  to entering interactive mode.

  Commands are executed in the following order:
  Command line options, -e <command>, script commands, interactive mode commands.

  When running 'rx/tx start' from a script or via -e, ensure these commands
  are later followed by 'rx/tx wait [timeout]' to ensure the program will
  not attempt to exit before reception/transmission is complete.
```

Figure D.3: The help text for the communication software

Much of the functionality of the communication software can only be used when a BladeRF is connected. For many options it is also required for the FPGA to be programmed. Connecting to a BladeRF and programming its FPGA requires a number of steps. The step-by-step instructions are provided below:

1. Connect the BladeRF to the computer with the USB 3.0 cable
2. Open a command line interface
3. Enter: “cd [PATH]\Receivers\BladeRF\Github\Testbed_bladeRF\host\build\output\Debug”
4. To start interactive mode:
   (a) For any device enter: “bladeRF-cli.exe -i”
   (b) For device # enter: “bladeRF-cli.exe -d "libusb: instance=#" -i”
5. Load the firmware on the FPGA with the command: “load fpga [PATH]\Receivers\BladeRF\Github\Testbed_bladeRF\hdl\quartus\work\output_files\hosted_testbed.rbf”
6. Verify that LEDs 1 and 3 are burning, and that LED 2 is flashing

The LEDs of the BladeRF will indicate if the programming of the FPGA has finished. If the FPGA has been programmed successfully, all three LEDs will be turned on. However, if the BladeRF is in interactive mode LED 2 will be flashing instead. The BladeRF is now ready to be used.

D.2.3 MATLAB

MATLAB is used to control the testbed. The code consists of a main script, and seven large functions which are each stored in a separate file. The build-up of the main script and these seven functions will be provided in this section.

Main script

The MATLAB script which controls the testbed is called “BladeRF_main”. Throughout this script a lot of commentary is available which indicates the function of each section of code. The program currently is divided in ten stages, each of those stages has its own functionality. The basic purpose of each stage is indicated by their name, and a more detailed understanding of the functionality contained in each stage can best be obtained by reading the commentary in the stage. A list of these ten stages is provided below:

1. Create a clean workspace
2. Create global variables and settings
3. Initialize BladeRFs
4. Measurement start
5. Prepare for sampling
6. Store measurement settings
7. Start sampling
8. Determine when the sampling is done
9. Analysis start
10. Script end

To keep this main file comprehensible, a large part of the functionality is performed by separate functions. These functions where specifically created for the testbed and are described below.

Functions

For the testbed seven functions have been created: “BladeRF_import_data”, “BladeRF_initialize”, “BladeRF_plot_data”, “BladeRF_print”, “BladeRF_set”, “BladeRF_start_sampling” and “BladeRF_verify_data_rate”. Each function is stored in a separate file. Each file starts with a clear help text and the MATLAB code is thoroughly explained by commentary throughout the code.

To inform the reader of all functionality contained in these seven functions, the help text of these functions are provided in Figure D.4 to D.10.

```
([..])BladeRF_import_data([..]) Imports all sampled data in matlab

  Data_struct = BladeRF_import_data(Folder_name) Imports all sampled data
  stored in the folder and converts it to a structure containing all data.

  * Folder_path => The full path to the folder which contains all data files.

  Example:
  Data_struct = BladeRF_import_data('C:\Data\')
```

Figure D.4: The help text for the BladeRF_import_data function

```
[...] = BladeRF_initialize([...]) Initializes all connected BladeRF's

  Device_info = BladeRF_initialize(Exe_location,Firmware_location, Force_load)
  Initializes all connected BladeRF's by finding the number of connected
  BladeRF's, storing all device information about the connected
  BladeRF's. If necessary or requested the FPGA's will be loaded.

  * Exe_location => The full path to the host software 'bladeRF-cli.exe'.
  * Firmware_location => The full path to the FPGA image 'hosted.rbf'.
  * Force_load => If Force_load=1 all connected BladeRF's will be
  programmed independent of there load status.

  Example:
  Connected_devices = BladeRF_initialize('C:\bladeRF-cli.exe','C:\hosted.rbf',0);
```

Figure D.5: The help text for the BladeRF_initialize function

```
BladeRF_plot_data([..]) Creates plots of the data

  BladeRF_plot_data([Data_struct,BladeRF_name]) Will create time and
  frequency domain plots of the data sampled by the BladeRF with the name
  "BladeRF_name"..

  * Folder_path => The full path to the folder which contains all data files.
  * BladeRF_name => The name of the BladeRF of which the signal should be
  plotted.

  Example:
  BladeRF_plot_data('C:\Data\','BladeRF_00');
```

Figure D.6: The help text for the BladeRF_plot_data function

```
[...] = BladeRF_print([...]) Provides some device parameters

  [Frequency,Bandwidth,Samplerate,Lnagain,Rxvga1,Rxvga2] = BladeRF_print(
  Exe_location,Libusb_instance)
  Provides the most important set device parameters of the BladeRF with
  the provided libusb instance.

  * Exe_location => The full path to the host software 'bladeRF-cli.exe'
  * Libusb_instance => The libusb instance number of the BladeRF

  * Frequency => The set frequency in Hz.
  * Bandwidth => The set bandwidth in Hz.
  * Samplerate => The set samplerate in sps.
  * Lnagain => The set gain of the LNA in dB.
  * Rxvga1 => The set gain of RX VGA1 in dB.
  * Rxvga2 => The set gain of RX VGA2 in dB.

  Example:
  [Frequency,Bandwidth,Samplerate,Lnagain,Rxvga1,Rxvga2] = BladeRF_initialize(
  'C:\bladeRF-cli.exe',0);
```

Figure D.7: The help text for the BladeRF_print function

```
BladeRF_set([...]) Sets some device parameters

  BladeRF_set(Exe_location,Libusb_instance,Frequency,Bandwidth,Samplerate,
  Lnagain,Rxvga1,Rxvga2)
  Sets the most important set device parameters of the BladeRF with the
  provided libusb instance.

  * Exe_location => The full path to the host software 'bladeRF-cli.exe'
  * Libusb_instance => The libusb instance number of the BladeRF

  * Frequency => The to be set frequency in Hz. Range: [232.5e6, 3800e6]
  * Bandwidth => The to be set bandwidth in Hz. Valid values: [1.5e6, 1.75e6,
  2.5e6, 2.75e6, 3e6, 3.84e6, 5e6, 5.5e6, 6e6, 7e6, 8.75e6, 12e6, 14e6, 20e6,
  28e6]
  * Samplerate => The to be set samplerate in sps. Range: [80e3, 40e6]
  * Lnagain => The to be set gain of the LNA in dB. Valid values: [0, 3, 6]
  * Rxvga1 => The to be set gain of RX VGA1 in dB. Range: [5, 30]
  * Rxvga2 => The to be set gain of RX VGA2 in dB. Range: [0, 30]
  * Counter => If Counter=1 the internal 32-bit counter of the LMS6002D will be
  selected as the data source

  Example:
  BladeRF_set('C:\bladeRF-cli.exe',0,1e9,28e6,10e6,6,30,3,0);
```

Figure D.8: The help text for the BladeRF_set function

```
BladeRF_start_sampling([...]) Configures the BladeRF and starts the
sampling process

  BladeRF_start_sampling(Exe_location,Libusb_instance,File_name,Number_samples,
  View_status) or
  BladeRF_start_sampling(Exe_location,Libusb_instance,File_name,Number_samples,
  View_status,Buffer_samples,Number_buffers,Transfers,Timeout,Master)
  Configures the BladeRF for the sampling of data and starts the process
  in a new minimized command screen.

  * File_name => The full path and filename to which the received samples
  will be written.
  * Number_samples => Number of samples to be received.
  * View_status => If View_status=1 the command screen will remain open
  to verify the output of the command
  * Buffer_samples => Number of samples which will be stored per buffer
  to use in the asynchronous stream. Must be an integer multiple of 1024
  * Number_buffers => Number of sample buffers to be used in the
  asynchronous stream. Must be 4 or larger.
  * Transfers => Number of simultaneous transfers to allow the
  asynchronous stream to use. Must be less than the buffers parameter.
  * Timeout => Data stream timeout in ms.
  * Master => If Master=1 the BladeRf will transmit the start_measurement
  signal after 5 seconds

  Example:
  BladeRF_start_sampling('C:\bladeRF-cli.exe',0,'C:\Data\Sampled_data.csv',1e6,0)
  BladeRF_start_sampling('C:\bladeRF-cli.exe',0,'C:\Data\Sampled_data.csv',1e6,0,
  32768,32,16,1000,1)
```

Figure D.9: The help text for the BladeRF_start_sampling function

```
BladeRF_verify_data_rate([..]) Creates plots of the data

  Blade_verify_data_rate(Data_struct) or
  Blade_verify_data_rate([Data_struct,BladeRF_name]) Will compare the data
  transmitted by the connected BladeRF's with a known array to verify if
  USB data packages have been lost. If requested a plot will be genetated
  from the named BladeRF's data.

  * Folder_path => The full path to the folder which contains all data files.
  * BladeRF_name => The name of the BladeRF of which the signal should be
  plotted.

  Example:
  BladeRF_plot_time('C:\Data\','BladeRF_00');
```

Figure D.10: The help text for the BladeRF_verify_data_rate function

D.3 Expanding the testbed

This section provides the user of the testbed a helping hand to expand the firmware and software of the testbed. Next to the information in this section, a lot of information can be found on the wiki of the developers of the BladeRF at: https://github.com/Nuand/bladeRF/wiki. Sometimes changing software and firmware can me a daunting task, but with the guides provided in this section a user with some programming experience should be capable to adapt the firmware and software of the testbed. This section provides a list of programs required for adapting the software of the testbed, the section also provides the instructions for how to use these programs.

This section will pinpoint the locations of important folders and files on the included DVD. The path to the main directory of the DVD, for example "F:\", is indicated by the text "[PATH]" throughout this section. All files connected to the receivers and the software are contained in the folder "[PATH]\Receivers\BladeRF\Github\Testbed_bladeRF".

D.3.1 BladeRF firmware

A lot of information about the firmware of the BladeRF can be found at https://github.com/Nuand/bladeRF/tree/master/hdl. The firmware is written in the VHDL language, and to compile new firmware only a single program is required:

- Quartus II version 13.1 with update 4

The firmware of the BladeRF consists of three parts, which are stored at different locations.

Top level design

The top level design is a single file. This file is the main file of the firmware, here the full behavior of the receiver is described. The file can be found at this location: "[PATH]\Receivers\BladeRF\Github\Testbed_bladeRF\hdl\fpga\platforms\bladerf \vhdl \bladerf-hosted_testbed.vhd". This file can be edited in plain text with any general text-editor. However, Notepad++ is recommended due to the build-in syntax highlighting.

IP cores

The IP cores are stored in the folder: "[PATH]\Receivers\BladeRF\Github \Testbed_bladeRF\hdl\fpga\ip\other". Newly created IP cores can be stored in the same folder, but they must also be added to the list in the file "[PATH]\Receivers\BladeRF \Github\Testbed_bladeRF\hdl\fpga\platforms\bladerf\bladerf-hosted_testbed.qip" before they will be included by the compiler.

Pin assignments

The assignment of all pins of the FPGA is done in a separate file. In this file the settings of all pins can be adapted. For example, their name can be changed or they can be set to be used as an input or output. These pin assignments are stored in the file: "[PATH]\Receivers\BladeRF\Github\Testbed_bladeRF\hdl\fpga\platforms\bladerf \constraints\pins.tcl".

Compile firmware

When changes have been made to any of the previously mentioned firmware files, the firmware should be recompiled. Compiling the firmware can be performed by following these instructions:

1. Open the “Nios II 13.1 Command Shell”
2. Enter: “cd [PATH]\Receivers\BladeRF\Github\Testbed_bladeRF\hdl\quartus”
3. Enter: “./build_bladerf.sh -s 40 -r hosted_testbed”
4. Wait
5. The created firmware file is called “hosted_testbed.rbf” in the “[PATH]\Receivers\BladeRF\Github\Testbed_bladeRF\hdl\quartus\work\output_files” directory
6. The hardware description can be viewed in Quartus by opening “bladerf.qpf” in the “[PATH]\Receivers\BladeRF\Github\Testbed_bladeRF\hdl\quartus\work” directory

D.3.2 Communication software

The communication software is written in C code. To compile a new version of this software, multiple programs and libraries are required. The full list is:

- Visual Studio Express 2013 for Windows Desktop
- libusb
- pthreads-win32
- CMake

How to find these resources and how to install them is described in detail on the wiki at: https://github.com/Nuand/bladeRF/wiki/Getting-Started%3A-Windows.

Adding commands

Adding commands to the communication software requires a couple of steps. First the C code for the new command should be written. Hints on how to build-up these files can best be acquired by examining the files created during this master's assignment: “blink.c”, “dac.c”, “pll.c”, and “measurement.c”. This C code file should be stored with the other source files at: “[PATH]\Receivers\BladeRF\Github\Testbed_bladeRF\host\utilities\bladeRF-cli\src\cmd”.

The command should then be added to the list in the file “cmd.c” which can be found in the same folder, and to the list in the file “CMakeLists.txt” which can be found in the grandparent directory. As a final step, a help text for the command should be written and added to the file: “[PATH]\Receivers\BladeRF\Github\Testbed_bladeRF\host\build\utilities\bladeRF-cli\src\cmd\doc\cmd_help.h”.

Because adding commands to the communication software is cumbersome, a how-to has been created by the developers of the BladeRF. This how-to is called: “ADDING_COMMANDS.txt” and it can be found in the folder: “[PATH]\Receivers\BladeRF\Github\Testbed_bladeRF\host\utilities\bladeRF-cli\src\cmd”. The instructions contained in this file explain in much more detail how to add commands to the communication software.

Compile communication software

When new commands have been added, or if commands have been edited, the software needs to be recompiled. The step-by-step instructions to perform this compilation are provided here:

1. Open “CMake (cmake-gui)”
2. At “Where is the source code” enter: “[PATH]\Receivers\BladeRF\Github\Testbed_bladeRF\host”
3. At “Where to build the binaries” enter: “[PATH]\Receivers\BladeRF\Github\Testbed_bladeRF\host\build”
4. Click the “Configure” button and select “Visual Studio 12 2013 Win64” and “Use default native compilers”, then click “Finish”
5. Wait
6. Enable “BUILD_DOCUMENTATION” and “BUILD_LIBBLADERF_DOCUMENTATION”
7. Click the “Configure” button
8. Click the “Generate” button
9. Close “CMake”

10. Open “bladeRF.sln” which was created in the build directory
11. Click “Build -> Build solution”
12. Wait
13. The generated software can be found in “[PATH]\Receivers\BladeRF\Github\Testbed_bladeRF\host\build\output”

Controlling the BladeRF

An introduction on how to use the communication software for basic device operations is provided at: `https://github.com/Nuand/bladeRF/wiki/Getting-Started%3A-Verifying-Basic-Device-Operation`.

D.3.3 Control software

The control software has been fully coded in MATLAB. All files can be found in the folder: “[PATH]\Matlab”, and start with the prefix: “BladeRF_”. The code does not need to be compiled and is edited and run in MATLAB itself. The only required program to adapt the control software of the testbed is therefore:

- MATLAB

The MATLAB files contain a lot of commentary. This commentary should provide the user with enough guidance to be able to adapt the software.

Appendix E

Testbed platform concept

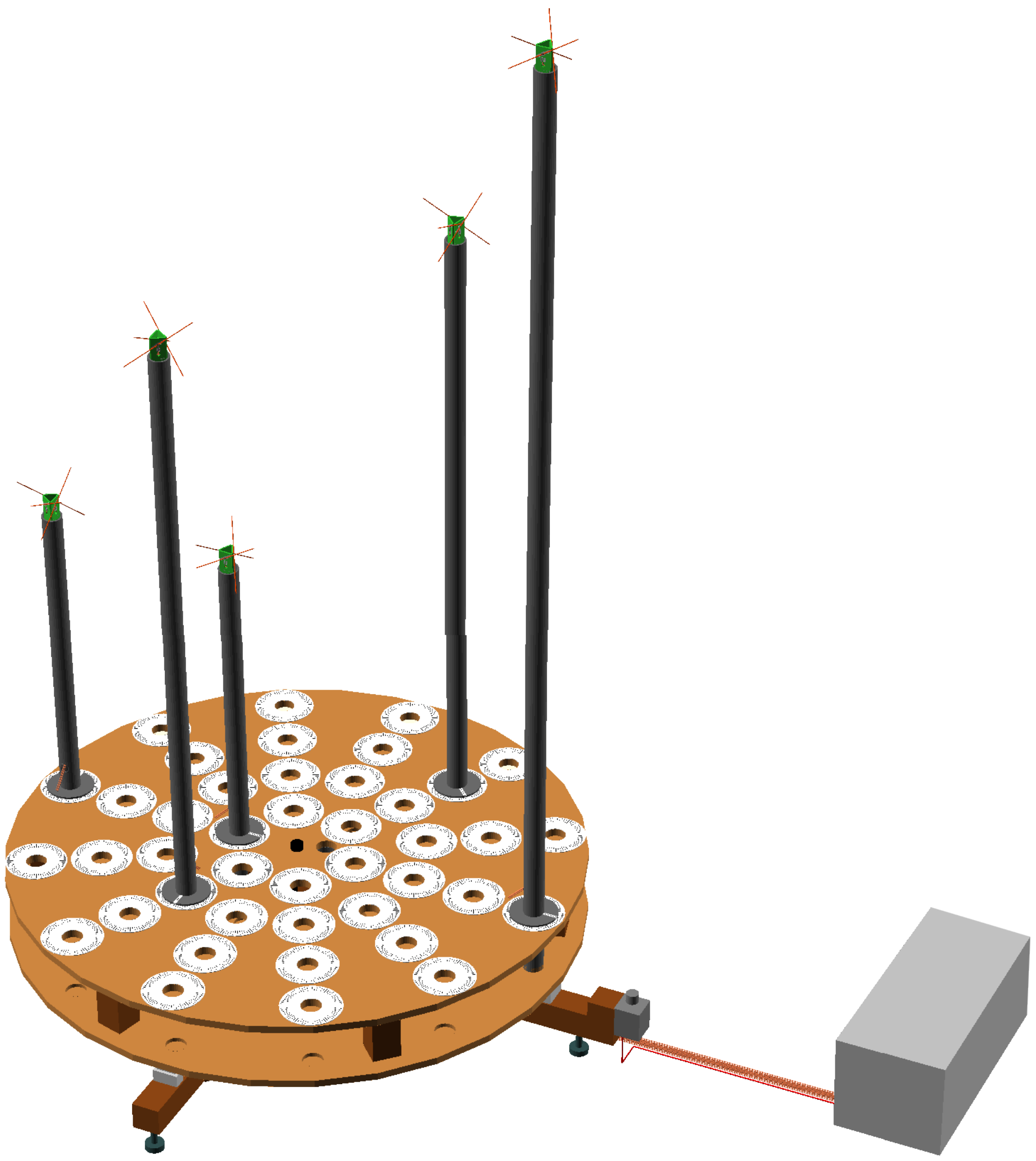


Figure E.1: Overview of the platform with an external box for all receivers and other electronics

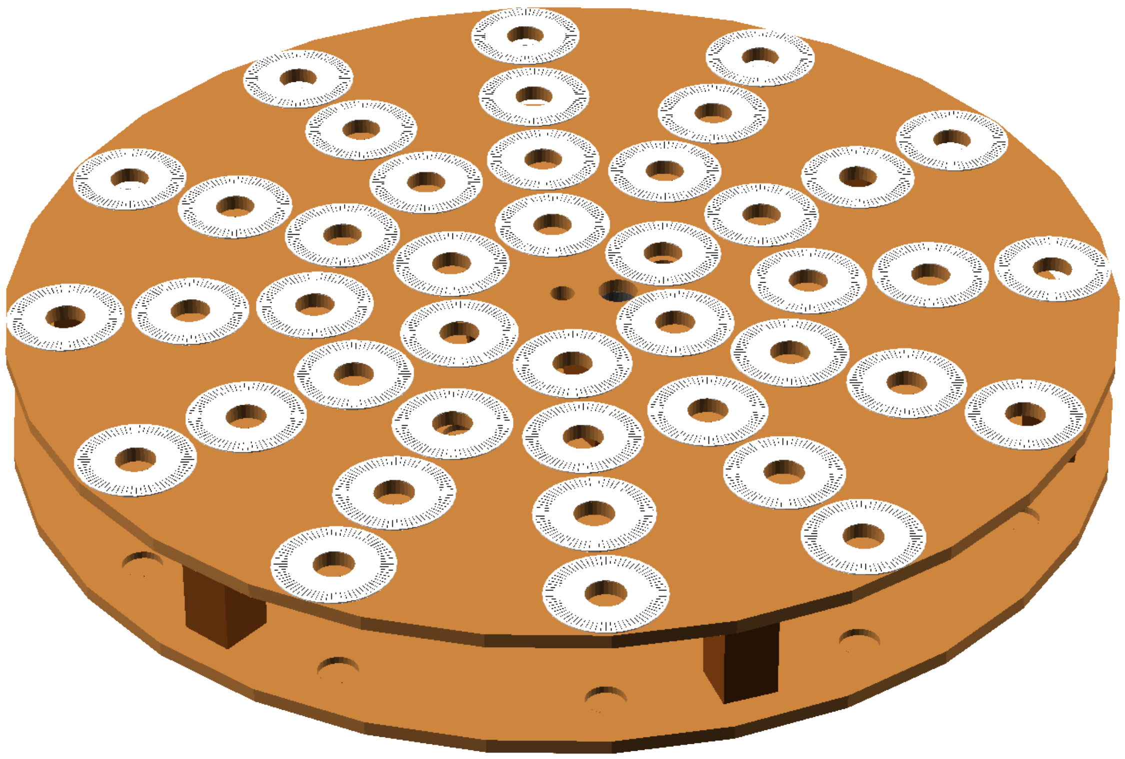


Figure E.2: Top view of the rotating disk with angle indicators around each hole for the antenna system poles

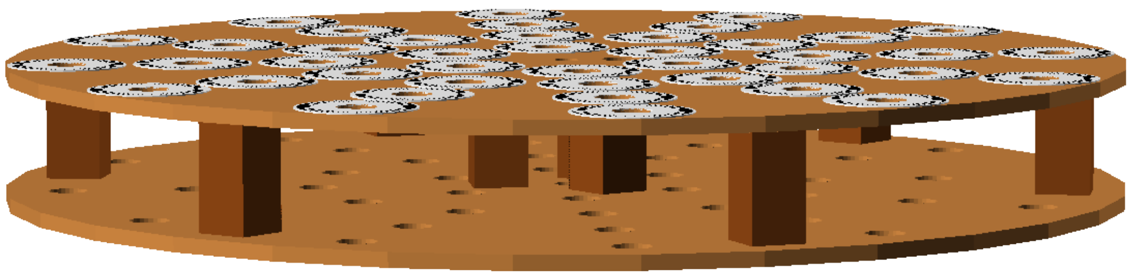


Figure E.3: Side view of the rotating disk with secondary holes in the bottom for the antenna system poles

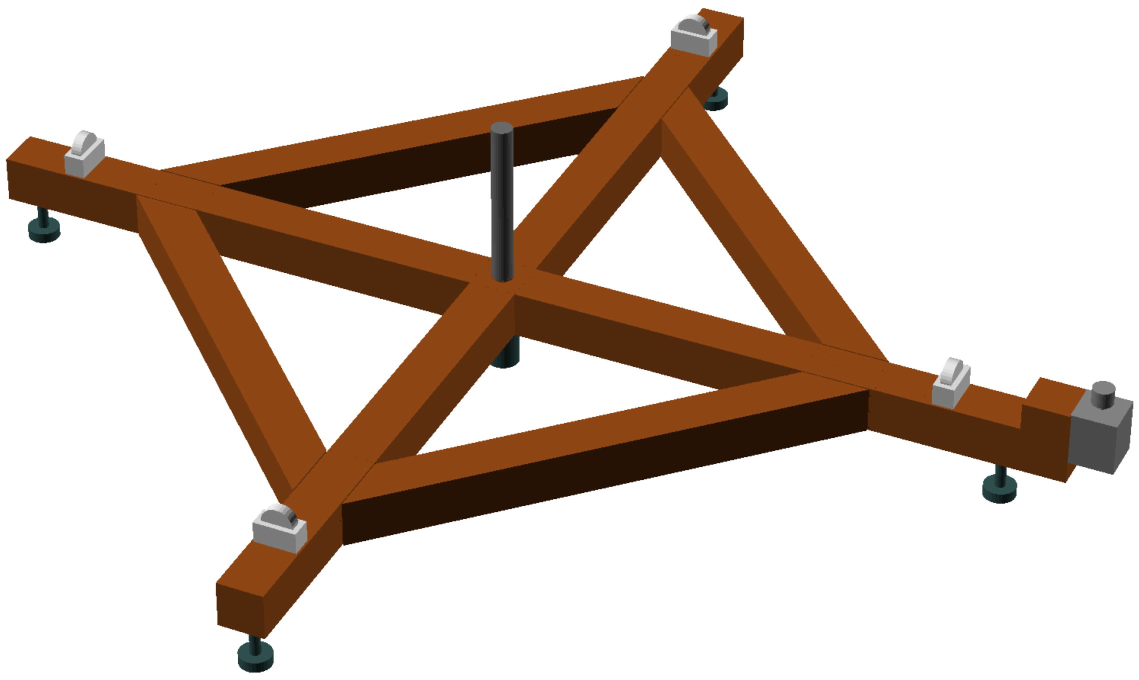


Figure E.4: Base of the platform with stepper motor to rotate the platform

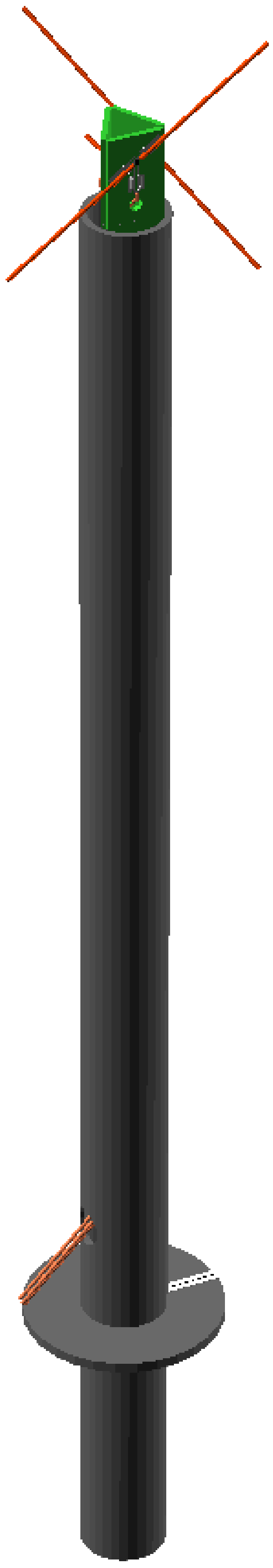


Figure E.5: Observational antenna system pole with angle indicator