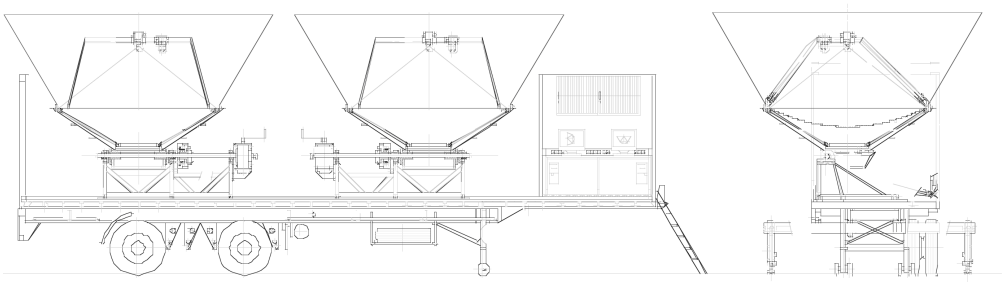
Delft University of Technology

Bachelor Thesis

Designing for TARA

The radar control unit



Authors:
Jeroen van Gemert
Martijn Janssen
Satoshi Malotaux

Supervisors:
Yann Dufournet
Tobias Otto
Simone Placidi
Christine Unal
Fred van der Zwan

June 19, 2011

Preface

As a consequence of global warming, an intensification of regional extreme weather phenomena (heavy precipitation, storms and flooding) is predicted in the coming years by climate models such as the one running at the Royal Netherlands Meteorological Institute (KNMI). Compared to worldwide climate changes such extreme weather events are often sudden, intense and localized in a specific area. In densely populated countries/cities they are considered as a real threat since the reaction time required to prevent devastating consequences is very short.

New atmospheric observation techniques are currently developed to increase the reaction time to anticipate on these weather events. Radars have been proven to be well suited for studying and monitoring these atmospheric phenomena in real-time. In the near future, more and more atmospheric radar systems will be deployed. Three different radars have been designed and built at the Telecommunications Department of Delft University of Technology in order to observe the atmosphere. They all have a leading position as state-of-the-art atmospheric radars and could be used as test radars for a future radar network deployment. Their data is currently used and tested among a large community of scientists, meteorologists and hydrologists for operational purposes. One of these radars is the Transportable Atmospheric RAdar (TARA) which profiles the atmosphere in height up to 15 km measuring precipitation and clouds. TARA is situated on the Cabauw Experimental Site for Atmospheric Resesearch (CESAR) at Cabauw, a small village near the village of Lopik.

TARA has now reached the age of 10 years. Considering the developments in processing power of computers in the past decade, this allows for a new approach in the design of the radar control and processing. Therefore, the main goal of the project is to create a completely new design aiming at controlling and processing the TARA data to maintain its leading position.

This project was done by five students as their Bachelor graduation project, it was done for the ATMOS group, part of Remote Sensing of the Environment (RSE). The students were divided into two subgroups. the control group of three students that had to implement the radar control unit and the process group of two people which had the task to build data processing, visualization and storage. This thesis describes the work done by the control group in designing a control unit for the TARA.

The control group of BAP group 6: TARA radar, would like to thank everyone at the ATMOS group for their valuable contributions to our work. First of all, Yann Dufournet, for his supervision and advises in design. Secondly, Tobias Otto, for his valuable presence in our room and direct help with the LabVIEW software; Thirdly, Christine Unal, for reading and suggesting changes for this thesis; Fourthly, Simone Placidi, for his supervision and advises and finally Fred van der Zwan and Paul Hakkaart, for their advises and help with the used hardware.

This thesis is written for readers with a background in electrical engineering. It can be used as a part of the documentation for building a system that controls an FM-CW weather radar with the use of LabVIEW.

Readers interested in the background of the research related to the project will probably like to read chapter 3. Chapters 4, 5 and 6 are in particular of interest to readers that like to know in which way the project was completed. Readers with an interest for the final results should read chapters 7 and 8, in which the actual test results are presented and discussed.

Contents

Summary

This thesis describes the work done on a system for the Transportable Atmospheric RAdar (TARA), which is at the Cabauw Experimental Site for Atmosperic Research (CESAR). It reveals the way in which, making use of a high-level programming language, a radar control unit is implemented in the system.

This project was set in a framework with a PXI of National Instruments, LabVIEW software, a DDS and provided Matlab code. An objective of the project was to deliver a working prototype. The TARA itself is a FM-CW radar with the possibility of sending and receiving in multiple polarizations and orientations for weather measurements. The design process started with the creation of a main block diagram, which defined certain sub-blocks. Those blocks were implemented and put together in the LabVIEW software.

The system was tested and most functions worked out properly. Actually, the same results of the measurements that the new system performed were found as the result during the measurements earlier in the morning, using the old TARA equipment. This proved that the system is working correctly, which is of great importance for the ATMOS group, because they are planning transport the TARA to France in 2012 for measurements using the new system.

List of Acronyms

ADC	Analog-to-Digital Convertor
BAP	Bachelor Afstudeer Project
CESAR	Cabauw Experimental Site for Atmospheric Research
CPU	Central Processing Unit
CW	Continuous Wave
DDS	Direct Digital Synthesiser
DIO	Digital Input Output
DSP	Digital Signal Processor
EM	Electro-Magnetic
FFT	Fast Fourier Transform
FM	Frequency Modulation
FM-CW	Frequency Modulated Continuous Wave
GUI	Graphical User Interface
HH	Horizontal receive, Horizontal transmit
HV	Horizontal receive, Vertical transmit
IC	Integrated Circuit
IDRA	IRCTR Drizzle RAdar
IF	Intermediate Frequency
I/O	Input/Output
ISA	Industry Standard Architecture
IRCTR	International Research Centre for Telecommunications and Radar
KNMI	Royal Netherlands Meteorological Institute
LF	Low Frequency
OB1	Offset-Beam 1
OB2	Offset-Beam 2
PC	Personal Computer
PFI	Programmable Function Logic
PXI	PCI eXtensions for Instrumentation
RAM	Random-Acces-Memory
RADAR	Radio Detection And Ranging
RF	Radio Frequency
RSE	Department of Remote Sensing of the Environment
TARA	Transportable Atmospheric RAdar
TCXO	Temperature Compensated Crystal Oscillator
TM	Timing Module
TU	Technical University
USB	Universal Serial Bus
UTC	Universal Time, Coordinated
VCO	Voltage-Controlled Oscillator
VI	Virtual Instrument
VH	Vertical receive, Horizontal transmit
VV	Vertical receive, Vertical transmit

Chapter 1

Introduction

With ever increasing computing power, the development of weather radars still is an ongoing process. The Transportable Atmospheric RAdar (TARA) is an example of such a radar. This system was built by the Department for Remote Sensing of the Environment at the Delft University of Technology and became operational in April 2000 [4]. The TARA is used for observing the atmosphere in order to better understand the behavior of clouds and their impact on the world's climate.

The aim of the department for remote sensing of the environment is to maintain the leading position of the TARA in order to continue in doing high-level research. As a result of the developments in computers systems regarding processing speed and data storage in the past decade, a potential increase of performance is possible. With this increase, TARA can maintain its leading position because more complex calculations can be performed and new atmospheric parameters can be estimated. This makes an update to the system necessary.

The main question addressed in this thesis is in which way the radar control unit could be implemented in the new computer system to enhance performance. The project started with establishing the program of requirements. This gave direction to which goal the process should have. This thesis will focus on a flexible solution that can process the data in a high level programming language for easy adaptability. This is done to be able to easily implement future developments in radar signal processing.

The only restriction for the project consists of the amount of time available. Because of this, the decision was made to make use of tools that allow for rapid prototyping in order to be able to give a proof-of-concept in a short time.

This thesis has been divided into nine chapters. Chapter 1 is a short introduction. Chapter 2 deals with the description of the project. The following chapter deals with the system requirements. Then, in Chapter 4 the related research is outlined and a comparison between solutions for rapid prototyping will be given. This chapter is followed by chapter 5, containing the design process. It will then go to chapter 6 about the implementation of the hardware and software. Chapter 7 reveals the results of the on-site testing of the system at the KNMI CESAR site at Cabauw. Chapter 8 contains the discussion. Chapter 9 evaluates the progress made during the project. In chapter 10, conclusions will be presented. In the last chapter, recommendations are made.

Chapter 2

Problem Description

At this moment, TARA has been in use for more than a decade. When the construction of the system was finished in April 2000, it was state of the art technology and put to good use in climate research. At the time, this task required the collection and storage of large amounts of data. During this period developments in the semiconductor industry have increased processing speed and data storage in an extent that it has allowed for a significant increase of performance. An upgrade of the TARA allows the systems to make full use of the current capabilities of electronics. This makes it possible for the radar to process and store more data to maintain its leading position. In particular, the radar data should be acquired and processed using the multi-polarization and the multi-beam capabilities of the TARA. This chapter will first define the problem; then define the objectives and ends with the framework of the project.

2.1 Problem definition

The main question addressed in this thesis is how the radar control unit can be implemented to enhance the overall performance of the system. The tasks of this control unit are to configure the hardware to generate the signal to be transmitted by the radar. This signal is defined by user input and generated by a Direct Digital Synthesizer (DDS). Another task is to digitize the signal from the receiver and to transfer the data to processing blocks, which will be discussed later. The difficulties with these objectives concern the timing of the processes and the sequence in which they occur.

2.2 Objectives

To start, the first objective was to establish a main block diagram, which was needed to design a common basis for the project. This diagram was needed to define all the interconnecting signals and subsystems of the overall system. This has been done by writing down the interface signals that are used to communicate between the different blocks designed by the two groups. In this way, definition problems were prevented when connecting the entire system.
The second objective was to define the tasks the radar needs to perform. These tasks consisted of designing software that can configure the new hardware while remaining compatible with the hardware currently present in the TARA.
The third objective was to construct a subsystem called 'system control', necessary to guide the subsystems from the two subgroups the project was divided in properly. and to anticipate possible occurring errors. Besides, this control block is responsible for resolving possible occurring problems by giving warnings to the user via the Graphical User Interface or even in the worst case scenario, by stopping the transmission.

The fourth objective was to create a radar control block. This block covers the creation of the transmit signal according to the settings of the user and covers the digitization of the received signal. Three things need to be done to accomplish this. First, a DDS needs to be configured to produce the desired signal. Second, an Analog-Digital converter needs to be set up to convert the signal. Third, a timing module needs to be present and configured to keep these two systems working synchronously.

2.3 Framework of the project

The project is set in a framework where certain hardware choices were already made by the supervisors, because of the equipment they purchased for the project.

One of the main requirements is that the design is compatible with the existing TARA. This implies that the old interface had to be used to control the radar. Because of the past experiences with National Instruments in reliability and service, the choice for using a PXI interface with the TARA was already made. The PXI is provided with LabVIEW software. This software uses a graphical programming language. A DDS was provided for generating the signal to be transmitted. This device from Analog Instruments can digitally generate the sweep needed. In order to acquire the information of interest from the measured data, the measured data is processed using provided Matlab code. All this hardware, software and algorithms were made available to us in our project room.

It is a goal of the project to deliver a working prototype. The little amount of time available plays a big role in the design choices to achieve this goal, because only seven weeks are available in which the system has to be designed and implemented.

Chapter 3

TARA as Research Instrument

In order to understand the assignment and to be able to put it into its proper context some explanation is needed for the existence of the TARA and its intended use. This chapter starts with explaining why the intended use is cloud research and why this is considered beneficial to society in the first section. The next section delves into why atmospheric radars are the instrument of choice when researching clouds. First an overview will be given of what radar types are available and then an answer will be provided to the question why the preference at RSE is given to the development of FM-CW radars. In addition, more information can be found in the Appendix History of TARA.

3.1 Atmospheric radars

The last few decades more and more people have become worried that the climate is changing and that this change will have dramatic effects on our lives and that of future generations. The problem is that the climate is a complex system with many variables and that there is very little knowledge about the impact many of these variables have on the climate. One of these phenomena of which there is still a lot of uncertainty are clouds. In order to gain a better understanding of the impact clouds have more knowledge needs to be gained of the behavior of clouds. That is why there is a need for instruments to be able to study them. Currently one of the most common ways to study clouds is by the use of Radar (Radio detection and ranging). A radar is a device that transmits signal in the form of an electro-magnetic wave and then receives and interprets the part of the signal that is reflected back to the radar by objects in its path. The radars that are used for studying atmospheric phenomena are commonly called weather radars.

3.2 FM-CW

The commonly used weather radars are the pulse radars. These radars simply transmit a pulsed signal and waits for its return. With the time difference the distance of the object can be calculated.

Another type of radar that can be used is the continuous wave radar (CW). This radar continuously transmits and receives a signal simultaneously. Range and Doppler information is, however, more difficult to extract . In order to overcome this issue, the frequency is also modulated in time (FM). The period of the modulation is called the sweep of the signal and the difference between the lowest and highest frequency is called the frequency excursion. From this modulation, range and Doppler information can finally be computed using two successive FFTs performed on the beat signal which is also obtained from this signal modulation and will be explained in Chapter 6.2.

3.3 Properties of atmospheric radars

3.3.1 Range

There are two reasons for developing FM-CW radars in Delft. These reasons are described here. The first reason is that a pulse radar has to transmit a short pulse with high power. This is done because as short pulse is needed for a good range resolution ΔR. This follows from the following formula:

$$\Delta R = \frac{c \times \tau}{2} \tag{3.1}$$

with c the speed of light and τ the pulse duration. This caused technological problems in the past, which is why the choice was made for FM-CW technology. Lots of power needs to be transmitted, because the reflection of the atmospheric particles is small. The second reason is that with FM-CW radars, the frequency excursion can be adjusted easily, which means that it is easy to adjust the range resolution ΔR. With pulse radars, only one or two range resolutions are possible, depending on the pulse length τ.

3.3.2 Doppler

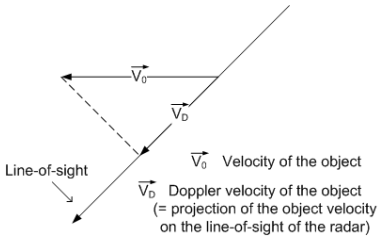


Figure 3.1: Doppler measurement

The Doppler effect is defined as a frequency shift from the original signal if a echo is reflected from a object moving along the line-of-sight of the radar. This is shown in figure 3.1. When this frequency shift is detected, the component of the velocity of the target can be computed. This component does not tell what the actual velocity of a particle is, but only the speed in one direction, also known as mean radial Doppler velocity components. In order to get the actual direction of this velocity, more components need to be acquired from a different angle. This is done by having three feeds at each radar (figure 3.2a). The central feed is used for the polarimetric transmission and is called MB, the other two which differ in transmission angle are called Offset-Beam 1 (OB1) and Offset-Beam 2 (OB2) as shown in figure 3.2b. Hence, three velocity components of a particle can be acquired enabling the calculation of the mean radial velocities of each beams. Using this information from a distribution of particles, a 3D wind figure can be retrieved from geometrical computations.

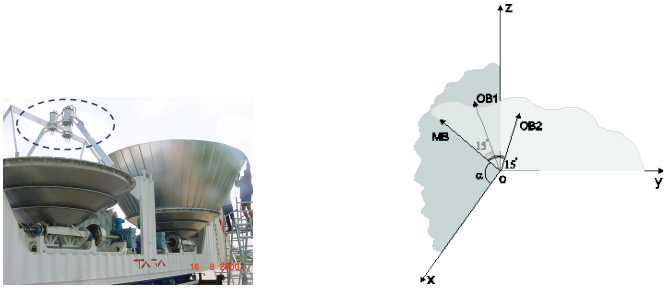


Figure 3.2: Three feeds, Main-Beam (MB), Offset-Beam 1 and 2 (OB1 and OB2)

3.3.3 Polarimetry

Polarimetric radars have the ability to transmit electromagnetic waves in specific orientations, e.g. horizontal or vertical. The reflection of these waves can also be received in horizontal or vertical orientation as can be seen in figure 3.3. With a combination of these orientations the shape of the target can be acquired. In addition the dimension of the target (whether it is small or large) is calculated using the magnitude of the received power. Atmospheric radars can interpret this to the type of particles that are present in a cloud (i.e. water or ice).

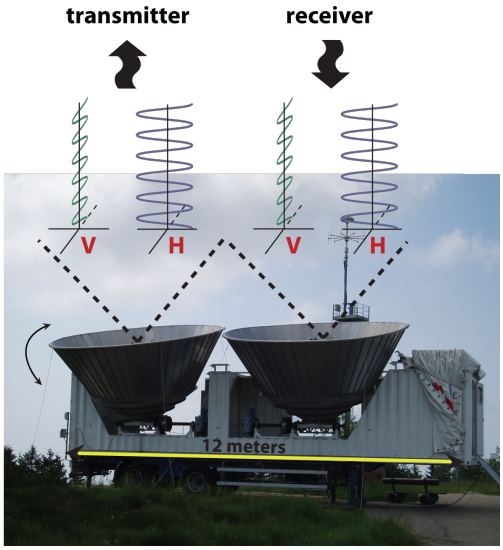


Figure 3.3: Polarimetry of TARA

Chapter 4

Program of Requirements

4.1 Introduction

For weather observations, atmospheric radars can be used. TARA is such a type of radar, with transportability as an extra feature. This radar uses continuous waves to scan columns of air in the troposphere for cloud particles and precipitation (usually referred to as hydrometeors).

The radar is built to measure with the radar beams in real-time the location and the properties of the particles in the atmosphere, their reflectivity (reflected power) and their mean Doppler velocity. Furthermore diverse polarimetric parameters are calculated for shape and classification purposes. All this information has to be visualized and stored in real-time. The TARA is owned by the TU Delft and placed at the Cabauw Experimental Site for Atmospheric Research (CESAR), a site where several types of weather measurement equipment are installed [14].

The current TARA system suffered from a lack of computation power required to correctly process the radar signal in real time. Modern computers are capable of doing complex calculations in parallel at a much higher rate than the current dedicated hardware and PC's present in the radar. This allows for a new design to improve measurement speeds as well as processing speeds. To make this new design, a computer is used that is connected to a data acquisition card together with a Direct Digital Synthesizer (DDS). This computer needs to be programmed in such a way, that it can replace the function of the existing dedicated hardware and has a design to allow for new functions to be implemented easily.

In this chapter, the requirements for the products that will be made by the control group are given. The remaining of this chapter has been divided into four parts. The first section concerns the product use. In the second section, environment issues related to the products are listed. In the third section, the design requirements of the system will be given. The fourth section provides an overview of the way in which the system is produced.

4.2 Product use

1. The system should be able to do weather measurements in real-time by controlling the transmission and reception of EM-waves and data processing.
2. The user has to be able to select diverse measurement specifications.
3. At the beginning of a sweep, change of polarization or beam has to occur with a sequence selected by the user.
4. The system should supply real-time data to the data processor.
5. A time stamp related to the data has to be created. This has to be done in the UTC time format.

6. At the start of the day or when the measurement specifications are changed, the system has to carry out a noise measurement.
7. The system should make a header with the measurement specifications.
8. The system should be able to acquire raw data when it is necessary.
9. The system should be able to measure without an operator.
10. The system should allow for continuous measurements.
11. The system should have the ability to save comments in the header.
12. The system should be able to display and anticipate errors it experiences.

4.3 Environment

1. The software has to run on a normal PC under the Windows operating system.
2. For the other parts of the system the product should be easy to communicate with.
3. The product should run stable without causing any instability to the PC running it.
4. The system should be user-friendly.

4.4 System design

1. Usage features
   (a) The system should be designed using an open approach, allowing for easy implementation of new functions and simple debugging.
   (b) The design of the software should be modular so that it is easily adjustable.
   (c) The block 'user input' should be able to select the different options in transmitting and receiving using well-defined signals.
   (d) The block 'user input' should be able to choose which data should be stored using well-defined signals.
   (e) The code written by the user containing the processing algorithms, is in Matlab and should be easy to implement.
   (f) If applicable, the system should require very little maintenance.
2. Production, commissioning and liquidation
   (a) The software should be modular so that it is easily adjustable to other systems.
   (b) The user should be able to update the code easily.
   (c) The software should be well-documented.
   (d) After the software is installed it should be ready to use immediately.
   (e) The system should be operational before the 7th of June 2011.
   (f) The install and un-install processes should be done within 60 minutes.
   (g) The personnel available at the CESAR site and the people from RSE should be able to install and un-install the software.

4.5 Production

1. The user interface as well as the processing software should be built in standard software.
2. The PXI system of National Instruments should be used for data collection.
3. A DDS should be used for generating the transmitted signal of the radar.

Chapter 5

Design Process

5.1 Project Organization

Because the supervisors of the project considered the task of upgrading the Control and processing of the TARA in seven weeks as challenging, they already provided a frame work for us to work in. They made the Gant-chart that is shown in figure 5.1. Because there were still a lot of unknowns on how to begin and proceed with and during this project. Weekly meetings where planned on all Monday mornings to asses the progress, discuss the weekly tasks and communicate possible problems.

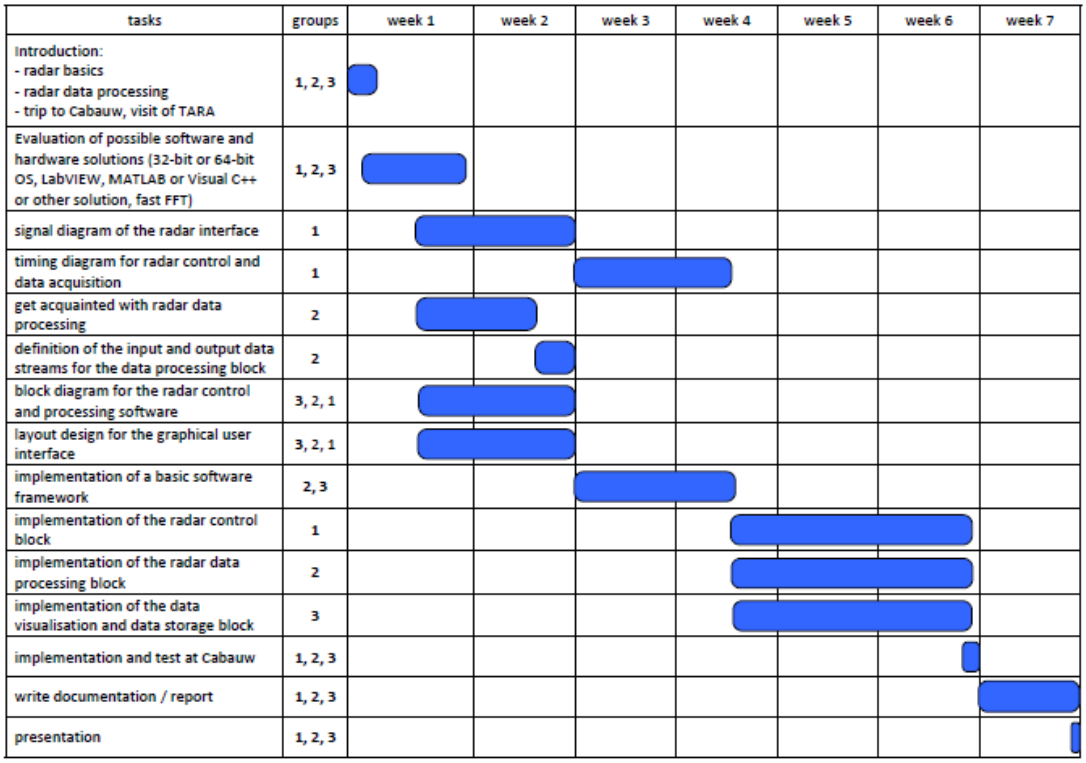


Figure 5.1: Gant chart

5.2 The Design Process

5.2.1 Journey Top-Down

In order to identify all the tasks the complete system needs to perform and to group similar tasks and to identify possible sub-blocks, a top-down design approach was chosen for the initial part the design process. By this method a general overview was obtained. This overview was needed to be able to divide the task between the control unit group and the processing group. In figure 5.2 the block diagram of the complete system is shown, a larger version of this picture can be found in the Appendix General Block Diagram together with a description of the different sub-blocks.

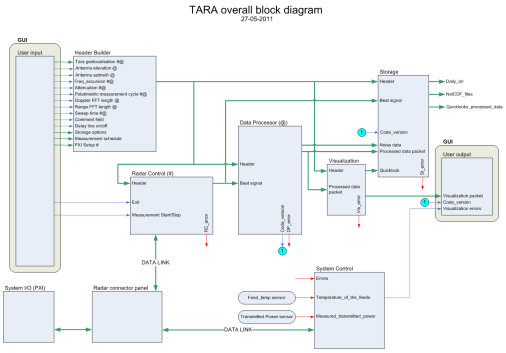


Figure 5.2: Implications of design choices

5.2.2 Detours going Bottom-Up

The Top-Down approach is a good conceptual exercise to identify the needed sub-blocks and system tasks; there are however some problems with this design approach, when used in a project with unfamiliar hardware and software. In such a project many unforeseen details may pop-up that will force the project to deviate from their original design. The sources for many of these problems came from LabVIEW, the PXI-system and the DDS.

In order to see how these different systems could be properly integrated in to a single system. Some time has been spend on finding out how to implement the devices by doing bottom-up design. Beginning with functionality that would be critical to the system as a whole, examples of this where communication with the DDS, PXI and TARA or signal generation by the DDS.

This bottom-up design was done by first indentifying critical functionality and that setting weekly tasks to implement and test the system and interfaces that would produce them. In this way we managed to implement all the absolutely needed functionality and in the final weeks to integrate them in a slightly changed original design.

5.3 Design Choices

Impact of choices

Figure 5.3 shows the way the design space, that represents the set of possible designs becomes smaller with every design choice taken. In this figure ican be seen how design choices impact the final design. For the radar control of course the design space was already confined by the choice of the Hardware and the construction of the TARA.

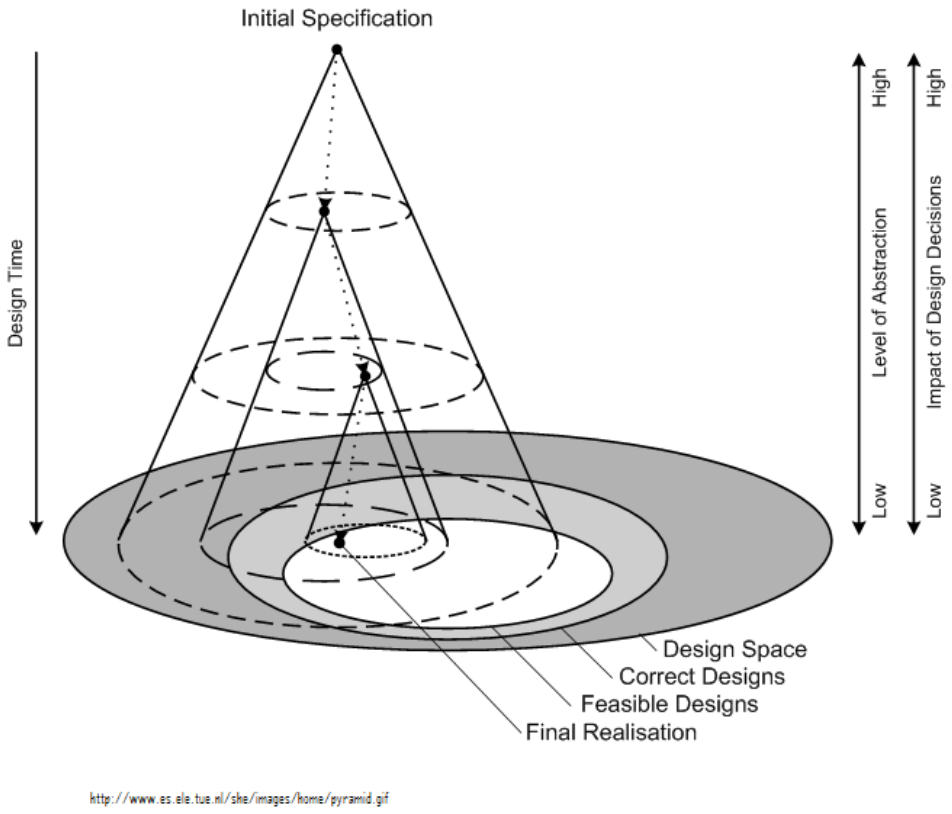


Figure 5.3: Implications of design choices [16]

Main considerations

As can be read in chapter four this assignment should result in a design of a working system with the available hardware, that is flexible and integrated as much as possible in one package. Although there where more design consideration then can be seen in figure 5.4. The ones that are given in the figure represent the main design considerations together with the given hardware. The concept that this figure is trying to convey is that these design considerations can only result in a limited set of solutions represented by the purple part of the figure where all consideration overlap.

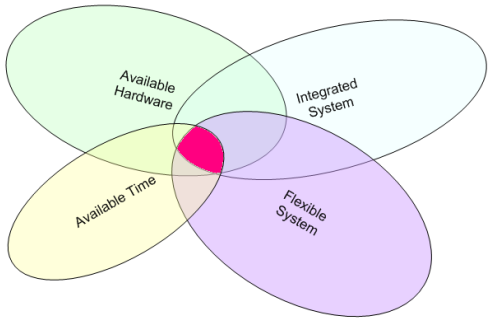


Figure 5.4: Main considerations

5.3.1 Choice for LabVIEW

The main design choice that had the biggest implication for the rest of the project was choosing LabVIEW as the language in which to write your program. Things that need to be considered when choosing are programming language are productivity, maintainability, efficiency, portability, tool support, and software and hardware interfaces [13].

With this project the main considerations are productivity, support and software and hardware interfaces. Productivity is important because the time for this project is seven weeks, support because no one in both teams is familiar with the language. The software interfacing is needed to be able to use Matlab in order to ensure the flexibility needed for atmospheric research. And as last the hardware interfaces are needed to communicate to all external devices.

The programming languages that were considerate are C, Matlab and LabVIEW. Of these languages the best suited was LabVIEW. The reasons for this are that Matlab can be integrated within a LabVIEW program, the interface elements to communicate with all PXI-systems are well documented and plug-and-play and it is a graphical language which makes it relatively easy to learn quickly. Especially the fact that the needed interfaces are readily available would save much time [13]. in implementation which would have a big impact on group productivity.

5.3.2 Choices in LabVIEW

There where two main problem with the use of LabVIEW. The first problem was that it is impossible in LabVIEW to communicate with data lines when a process is running and the second is how to distribute the needed variables to processes that are running in parallel.

Functional Global Variables

For the variable distribution problem there where two alternatives that could distribute the needed variables, Global Variables and Functional Global Variables [1].

Global variables are variables in LabVIEW that can be accessed and passed between all processes in the program at any time. This property makes these variables vulnerable to race conditions, which occur when two different processes try to write to the same global variable. If this happens only one will be successful and the information from the other process will be lost. The use of global variables will also affect performance, when a global variable is called by a process it will be copied in memory at another location, so that when they are used by multiple functions and are used over and over again, the system memory will fill up with copies the same variables.

Functional Global Variables are similar to global variables in there ability to distribute variables throughout a program with multiple running processes. The main difference is that functional global variables are initialized by linking variables to them after which they are stored in a shift register. Then if the Functional Globals are called by a process this process is allowed to access the functional global as long as is necessary without other processes being allowed access. This property resolves the race issues that global variables have. Another property is that Functional Globals are only stored in one place and so the needed memory for the program is stable when using Functional Globals.

To summarize, Functional Globals and Globals solve the same problem but Functional Globals solve the problem without affecting reliability and performance.

Notifiers

Notifiers can also be used to communicate between the different parallel-running processes in a program just like Functional Globals and Globals however notifiers take less time implement correctly and result in a more efficient program. There implementation is easier because this element was added to LabVIEW for this purpose by National Instruments. Efficiency is better because a process that needs a notifier will only run if this notifier is available and else be paused. If either one of the globals would be used to transfer this variable it would mean the same process would continually check if the global is available. The design

decision made was to implement the signals that are needed to synchronize and control the order of processes in the program with notifiers and to still distribute data variables with Functional Global Variables. This was done to implement the program in a way that the overview remains on the screen while programming. The computational variables are nicely collected in one place in the program and can be accessed easily. Also, the notifiers are the only signals on screen that are controlling the sequence of events.

5.3.3 Hardware versus Software Triggers

In the system there are important hardware components that are controlled by trigger signals. On every rising flank of one of these triggers starts a certain action, e.g. starting the sweep. When all the systems operate independently of each other, there would not be a problem. However in radar systems this is not the case. All the systems need to be well synchronized. When one system acts out of sequence, the whole system will get out of line and the acquired data will be worthless. Therefore all triggers should be as precise as possible in order to ensure a proper sequence of operation between all subsystems within the radar.

There are two ways of producing the needed triggers:

- With the software
- With the hardware

Software has three advantages and one disadvantage as a trigger source. Triggers generated by the software are easily implemented, can be monitored to insure proper operation and gives exact control over the sequence of operation. This makes generating the triggers with the software to look very attractive on first sight. The disadvantage comes from the fact that LabVIEW is a high level language that runs in its own environment that again runs on Windows. This is a problem because when running on Windows you have no control of the scheduler of this operating system. The timing is then dependent of the priorities and order in at which this scheduler operates and although priorities can be influenced with LabVIEW, exact timing control can never be implemented or guaranteed. So with this implementation the sequence of event can be easily controlled in a transparent way, but exact timing remains an issue.

Hardware triggers have two disadvantages and one advantage over their software equivalents. Disadvantages are that the triggering process will be completely done within the timing module so that this process will be hidden for the main LabVIEW program. For this reason it becomes very difficult to see at what stage the process is running and this makes it difficult to check if the system will stay synchronized with the other subsystems in the radar, especially for the data processor which needs to know the polarization state to process the data generated by radar. Furthermore there is a chance that the ADC trigger will be missed and the operation of the entire systems gets out of order. However,by adding safety margins into the trigger delays the probability of such unwanted behaviour can be reduced. The advatage is that it gives is exact control of the timing between TARA, the ADC, the DDS and the TM.

Because timing and synchronization is such a critical issue within our system, the choice was made to implement the generation of the trigger signals in hardware. This would gives the most stable and accurate system with exact timing control. The drawback of the hardware implementation was resolved by bigger margins and by calculating the optimum configuration of all different subsystems involved before every measurement.

5.3.4 Pin Choices

The hardware of the TARA that is responsible for the transmission is connected to the Radar Control Unit with three signals. These signals are RF_enable, RF_fast and RF_ack. The RF_fast and the RF_ack are used to trigger events when a measurement is being taken. RF-fast switches on the transmission power of the TARA transmitter and RF_ack detects the presence of power on the transmission line. There are two paths available in the PXI-system to connect these signals from the TARA to the Radar Control Unit.

- Via a PXI Digital I/O
- Via the PXI Timing Module

In the current system the TARA in connected to the Prodrive I/O with a standard 40 wire ribbon cable. The PXI digital I/O uses the same connector so the same cable can be used so the result is easy to install and nicely integrated, with only a limited amount of cables. But because signal RF_fast has to have a specific form to perform its function, it became a problem to use Digital I/O.

The reason for this is that although it is easy to make a signal with a specific form with Timing Module in LabVIEW. There is no possibility to transfer a exact copy of this signal to the DIO. This is because when this signal is transferred over the trigger line from the TM to the DIO. It can only use DIO this signal as a trigger to create a signal by reacting on the high or the low flanks. So a simple exact transfer of the signal can not be accomplished. This means that the only possible signal that can be transferred is to the DIO in this manner is a signal with a 50 % duty-cycle.

The other path available, is by directly transferring the RF_fast signal from the Timing Module to the TARA by the PF pins. This has the disadvantage that an extra cable will be needed and that changes will have to be made to the connector of the TARA. But this path gives the possibility to transfer the signal exactly as needed.

Because the RF_fast trigger signal needs to have a very specific shape there we choose for the path via the PXI Timing Module. This leads to a modification of the connections and extra cables that and therefore strays from a fully integrated solution, but it makes it much easier to implement the needed shape of RF-fast and therefore much less time consuming.

5.3.5 DDS Serial I/O

Direct Digital Synthesizer has to be programmed to be set in the proper modulation mode needed. There are two ways to program the DDS.

- Using the Analog Devices Evaluation Software provided with the DDS
- Write a VI in LabVIEW to program DDS via Serial I/O.

Using the software provided by Analog Devices has an advantage and a disadvantage. The advantage is that it gives a plug and play option to program the DDS using the USB connection. In this way the programming of the DDS is simple, fast and it works. Using the Analog Devices software would mean that our clients need to set the measurement setup values in two different software interfaces which is a possible source of human error. This implementation goes against the wish of our clients for a system that is integrated in one single program.

Programming the DDS with a VI in LabVIEW is time consuming. First, it is necessary to understand the way the registers have to be set and how to communicate to the IC. Secondly, a program has to be written which can easily be used and which is flexible for possible future adjustments. The advantages are that a program written in LabVIEW can be easily integrated in the rest of the system. This leads to one single program controlling the whole system which complies to the need for a integrated system . Another advantage is that when the DDS is implemented in LabVIEW it is easy to switch between the different modes because interfacing is custom made to the needs of the radar. For instance, for a noise measurement the single tone mode will be set and for a data measurement the frequency modulation mode will be set. This will be done with a single instruction in LabVIEW.

The choice was made to write a VI that programs the DDS. This will make it possible to easily switch between radar operating modes, give one program controlling the system that is flexible and safe to use.

5.3.6 Transmission

Before the start of a measurement the user selects the wanted polarization cycle that will be used to set polarization. Nowadays there are two cycles that can be used for a measurement, e.g. VV-VH-HH or VV-VH-HH-OB1-OB2. To implement such a cycle a system has to be made. One way is to make a program from completing a cycle as in the state diagram that can be seen in figure 5.5. It starts at the noise state, after this is completed it continues to VV, goes to the next state VH and then arrives at HH. At this state there are two paths that can be chosen, VV and OB1. The path chosen will make the distinction between a 3 and a 5 cycle measurement. The advantage of such a system is that it is clear and that there are only two possible routes that can be taken. The disadvantage however is that this system is not flexible, a cycle other then 3 or 5 is not possible.

Another approach is to make a flexible system as can be seen in figure 5.6, here an arbitrary cycle can be used as polarization cycle. The measurement starts in the Idle/Noise state and reads the value of the first desired state from the cycle and will jump into this state. When this state is completed it will go back to the initial state, reads the next value and jumps into the next state and so on. The advantage is that there is not a limitation on the polarization cycle anymore, the disadvantage is that it takes some time to make such a system. Since the processing block is not ready to process a cycle different than the 3 and 5 cycle measurement, it would not be necessary to make a system that is able to set polarization other than those two cycles. However, since the system needs to be flexible so that future upgrades of the system can be easily implemented, the second system, following an open approach, has been created.

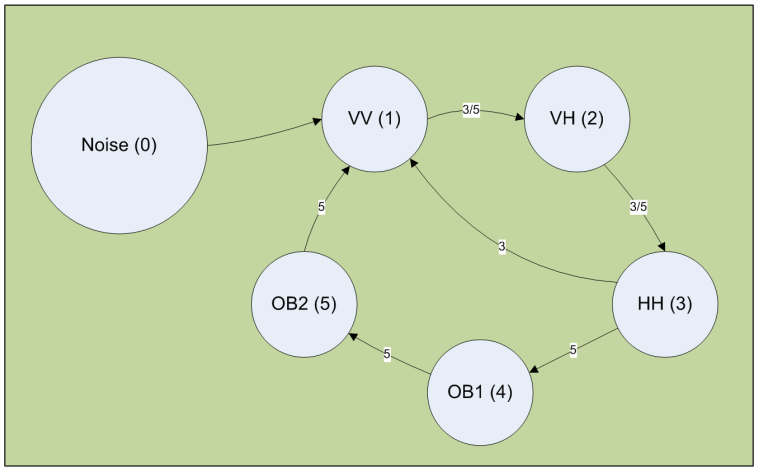


Figure 5.5: Old transmission scheme

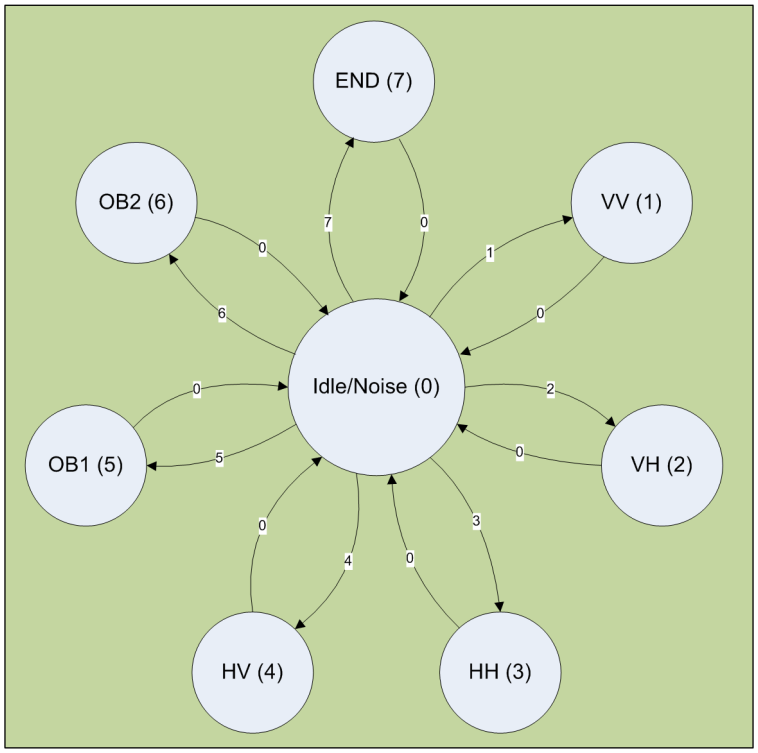


Figure 5.6: New transmission scheme

Chapter 6

Implementation

In this section, the implementation of the entire system is described. This is done by first describing the way the hardware is implemented and finally describing the way everything was implemented in LabVIEW.

6.1 Hardware

6.1.1 TARA

The set the TARA radar in the proper setting a connection needs to be made with the two I/O slots. The connections are shown in figure 6.1 (A bigger version can be found in Appendix TARA I/O). For the inputs of the TARA a high signal mean that system will be active an low that system is inactive. For the output signal a high signal means that system is active and low that system is inactive. The way to use this interface is to set all pins in the wanted setting and then the Radar will configure it self in the right way. Only remark is that a change of setting will have a delay less than a 1 μs, so a certain time interval is needed before one can assume that the radar in the proper setting.

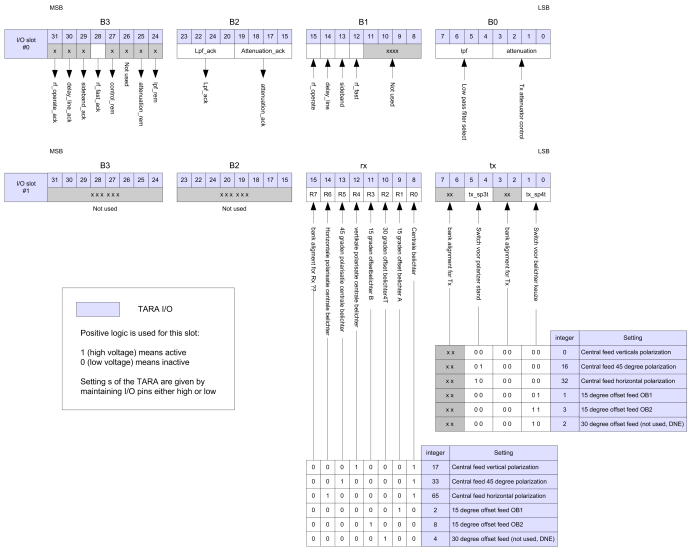


Figure 6.1: TARA I/O

6.1.2 PXI Chassis NI PXIe-1082

The PXI chassis [26] used for TARA is from National Instruments. This systems has slots available for 8 cards which can be PXI and PXI express cards. The system consists of the following cards:

Slot	Card	Name	Function	Data sheet reference
1	NI PXIe-8360	MXI-express	Connectivity to PC	[25]
2	NI PXIe-6535	10MHz digital I/O	Interfacing	[24]
3	NI PXIe-6535	10MHz digital I/O	Interfacing	[24]
4	empty	-	-	-
5	NI PXI-6682H	Timing module	Synchronization	[22]
6	NI PXI-5922	16-bit to 24-bit ADC	Analog-to-digital	[23]
7	empty	-	-	-
8	empty	-	-	-

NI-PXIe-8360 MXI-express
The chassis must be connected with a computer in order to be able to program the cards. This is done via this MXI-express which is connected with the PC using a serial connection.

NI-PXIe-6535 10MHz digital I/O
This PXI express card has a clock rate of 10MHz which is impossible to phase lock with an external clock. It supports 32 channels to which data can be send to and received from. This device is used as a communication link to TARA and the DDS, which is the reason there are two digital I/O cards in the chassis.

NI-PXI-6682H Timing Module
For the control of the radar it's important to have one reliable module that's responsible for all the timing and synchronization of the system. The Timing Module that is used for this task is the National Instruments PXIe-6682H. This module has a GPS input which can be used to read out the time and location. In this device there is a temperature-compensated crystal oscillator (TCXO) present which can be used to send a sharp 10 MHz clock to the clock out pin.

This module has internal logic which can be programmed via for example LabVIEW. Hence, it is possible to set up signals which can be outputted on one of the 7 trigger lines at the backplane or on one of the 3 PFI pins at the front panel. These PFI pins can be used as input as well as output, which makes it an easy way to communicate with other external devices, e.g. TARA.

NI-PXI-5922 16-bit to 24-bit ADC
The data that is received from the radar is in an analog form. To be able to process the data later on in the process it has to be digitalized using and analog to digital converter. The condition of this converter is that it has a fast data acquisition and a high resolution.

The National Instruments PXI-5922 is the digitizer that is used. This device is a 24 bit flexible resolution digitizer and comes with a high speed 256 MB memory. The sample rate that can be used is up to 15 megasamples per second. This rate cannot be picked arbitrary but has to be a value calculated with formula 6.1.

$$Samplerate = \frac{60\frac{MS}{s}}{n}, 4 < n < 1200 \tag{6.1}$$

This resolution and sample rate are closely related to each other. When selecting a sample rate lower than 500 kS/s a resolution of 24 bit is reached. The highest sample rate of 15 MS/s causes the resolution to fall back to 16 bit.

On the front panel of the digitizer there is an analog input available which can be used for digitalizing a signal. Another input is the reference clock, with this input the internal clock can be phase locked with another clock which enables a good synchronized system.

One handicap of this piece of hardware is the time it takes to reinitialize the trigger after each data acquisition. This amount of time is called the rearm time and is related as in formula 6.2.

$$t_{rearm} = 144 \times T_{sample\_clk} \tag{6.2}$$

6.1.3 DDS

The DDS is an IC that has several different operating modes [27]. Because the TARA is a radar that uses frequency modulation, the DDS had to be set into frequency modulation mode. This can be done by filling the internal control registers with the proper bit sequence. In this frequency modulation mode, the parameters of the frequency excursion can be set by filling the internal channel registers with the wanted values.

These registers can be programmed by a USB connection or by making use of a Serial I/O. For programming with the USB an easy to use program is provided by Analog Devices. When making use of the serial I/O connection a program has to be written to provide the communication protocol. This protocol needs to provide an update signal so the registers are set to the new values and its needs to give instruction to either read or write to a given register.

For synchronization purposes the DDS has to be connected to a reference clock generator by the oscillator so that the internal clock of the DDS can be phase locked to it.

6.1.4 Hardware connections

All this hardware had to be connected in a proper way to get the system working. In order to do so, a connection scheme was drawn, which gave an overview of the connections of the system. This overview is shown in figure 6.2. The computer connects to the PXI via the MXI-express interface. The PXI is connected to the DDS via the PXIe-6535. The signal that is received by the TARA is digitized using the ADC in the PXI-5922.

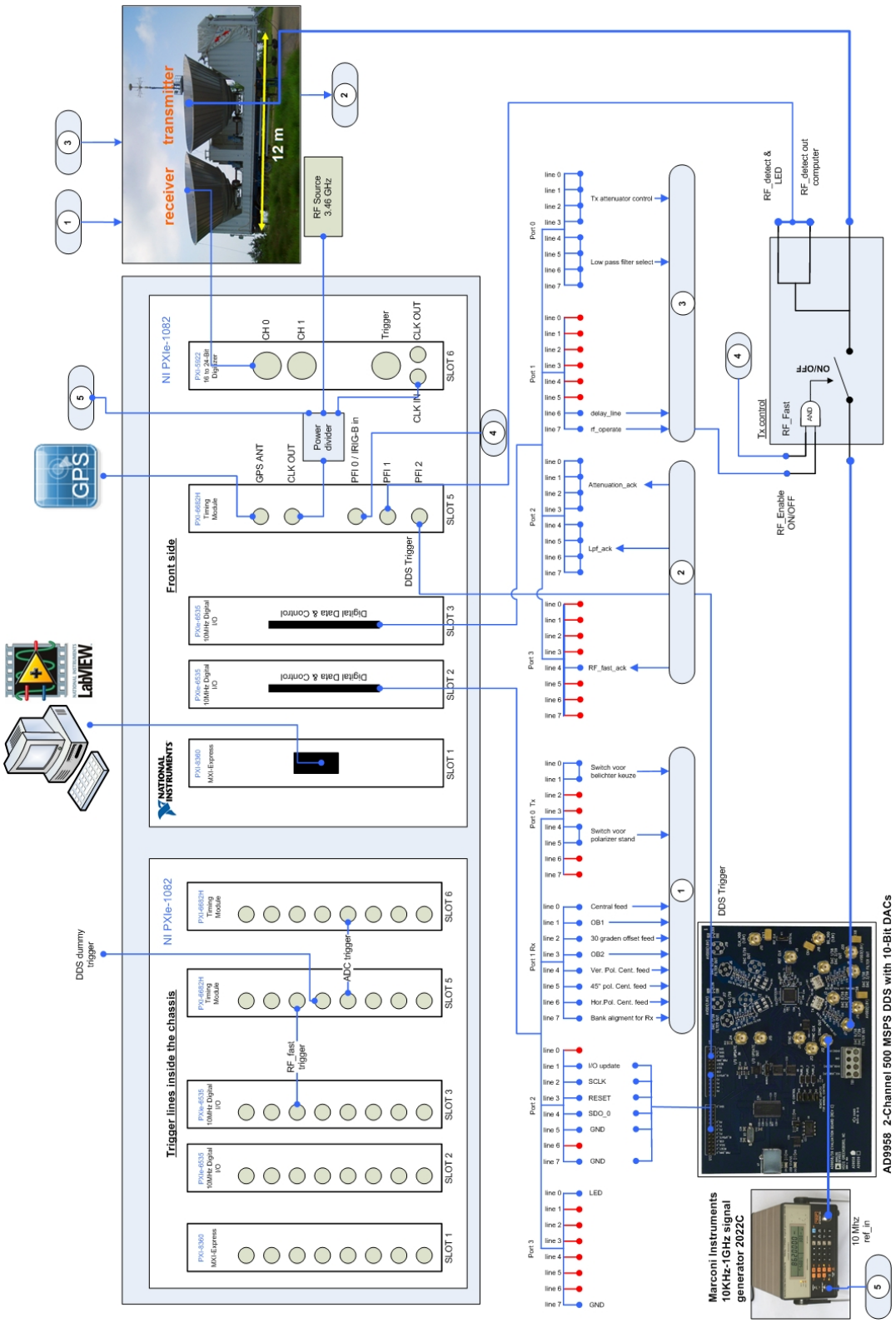


Figure 6.2: Hardware interconnections

6.2 LabVIEW

LabVIEW is the programming language used in this project. And because the reader might not be familiar with the specifics of this graphical programming language this section explains some of the more crucial LabVIEW concepts used in the implementation of the Radar Control Unit.

VIs

A VI stands for Virtual instrument and this is one of the basic building blocks for programs written in LabVIEW. It is similar to a function or subroutine in other programming languages. The term Virtual Instrument comes from the fact that LabVIEW applications are designed for writing software that simulates the functionality of instruments, they are virtual because they exist in software [18].

Notifiers

Notifiers are used to communicate between the different parallel-running sub-VIs in a program. These notifiers are used in our program to control the order of events or sub-VIs in the program.

Functional Global Variables

Functional Globals are VIs that allow controlled access to data or resources. The primary use of Functional Globals is to provide data sharing across an entire project, both horizontally and vertically. Horizontal sharing means that multiple top-level VIs running autonomously can all have access to the same data. Vertical sharing means that sub-VIs at any depth have access to the data without requiring explicit passing of parameters through VI connector pane terminals which is depicted in [19]. The Header Builder in the radar control is a Functional Global VI and is thus the VI that provides all the needed variable throughout the program

State Machines

The state machines in LabVIEW are all implemented as a case structure is contained within a while loop which is also shown in [20]. Execution of particular cases in the structure is determined by the output from the previous case (or in the instance of the first execution) by the control selector input. The control of the order of case execution is controlled through the use of a shift register. By wiring the output from one case as the determinate to the subsequent case (wiring to the shift register on the right hand side) and wiring the input to the case selector from the left hand side shift register, operations can be ordered according to which state is executed.

Radar Control VI

The Radar Control is the Top level VI of the program. When this VI is started, it first constructs the needed notifiers. Then it starts the System Control VI. The System Control is a VI that performs two tasks. The first task is handling and logging system errors and the second is to prevent damage to the system in the case of malfunctions. After producing the notifiers and the System Control, the Radar Control State Machine is started. The task of this state machine is to control the synchronization and sequence of all VIs used in this program. The state diagram of this state machine can bee seen in figure 6.3.

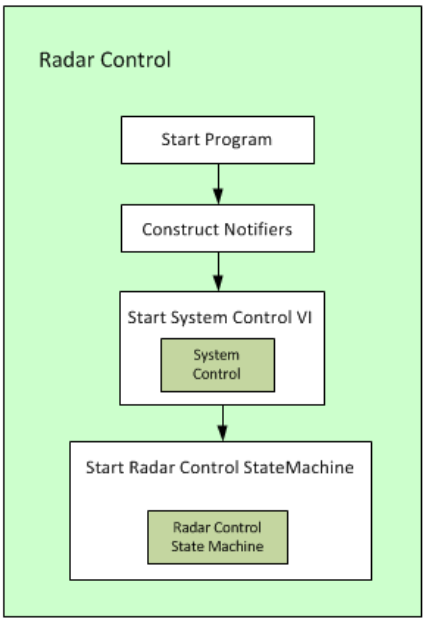


Figure 6.3: Radar control VI

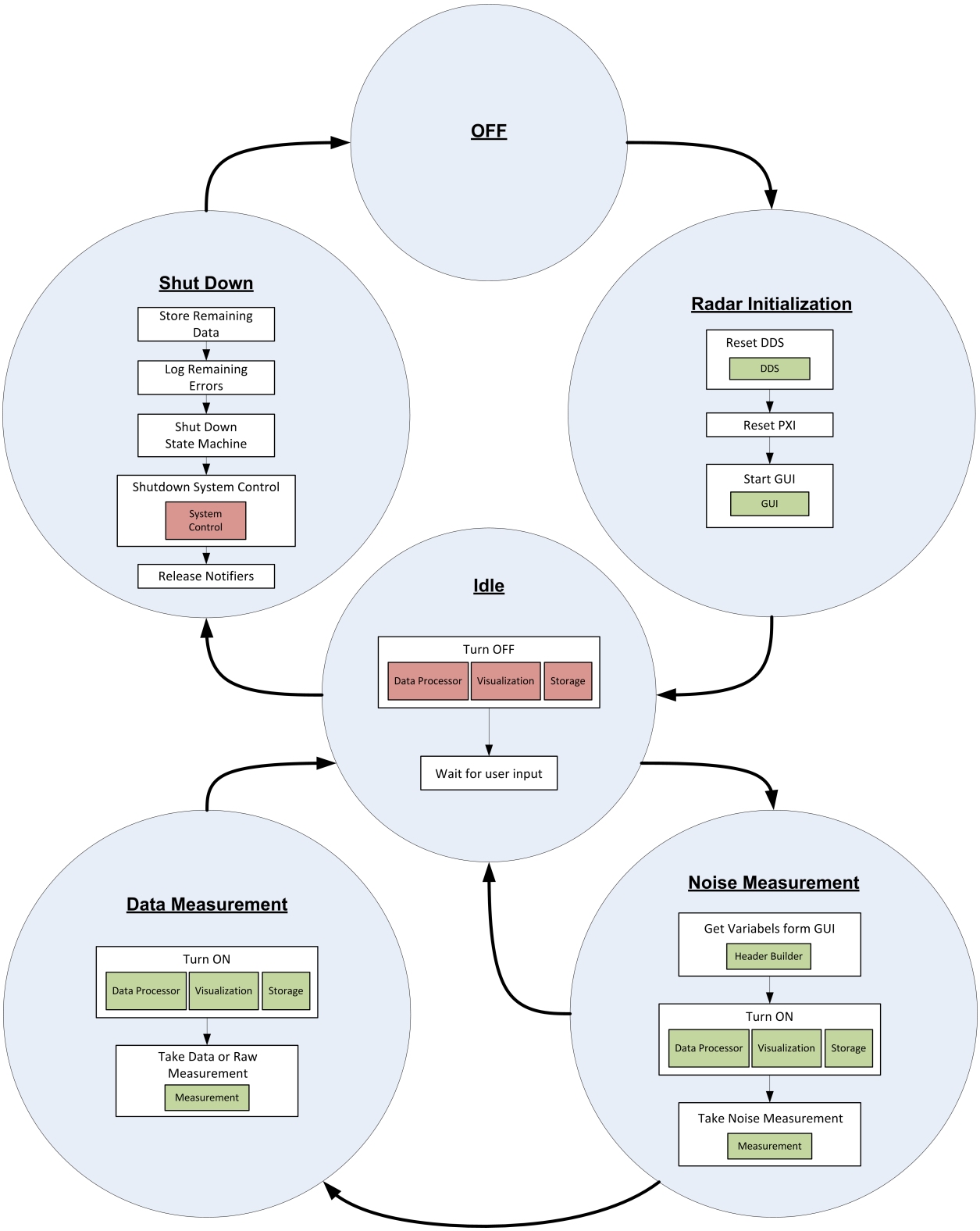


Figure 6.4: Radar control state machine

Radar Control State Machine

As can be seen in figure 6.4 the Radar Control State Machine consists of six states. These states are named OFF, Radar initialization, Idle, Noise Measurement, Data Measurement and Shut Down. The OFF state represents the state machine before it is started. Then when the State Machine is started it goes to the Radar Initialization state, in this state the DDS and the PXI are reset, and the Graphical User Interface (GUI) is initialized. The state machine moves to the Idle State when these tasks are completed, where it waits for the user to start the measurement. When a measurement is started it moves to the Noise Measurement State. In this state, the Data Processor, Visualization and Storage are turned on and the measurement VI that is described in the next section is used to generate the wanted beat signal that is provided to Data-Processor and/or Storage. Hereafter, depending on the measurement type, the state machine will go to either the Data-Measurement-State or the Idle-State. If it goes to the Idle-state the processor, visualization and storage will be turned off and the system will wait for the start measurement again. But if the measurement type is of the data-type, the state machine will transfer to the Data-Measurement-State. In this state the processor, visualization and storage are still on and the measurement VI takes the data measurement. After the measurement is finished the state machine will return Idle-State. That leaves the Shut-Down-State. The radar will go into this state if the user shuts down the program. In this state all remaining processes like processing, storage and error login are finished. After which the System Control is shutdown and the notifiers are released. The State Machine is able to provide the user with four kinds of measurements. These are shown in figure 6.5.

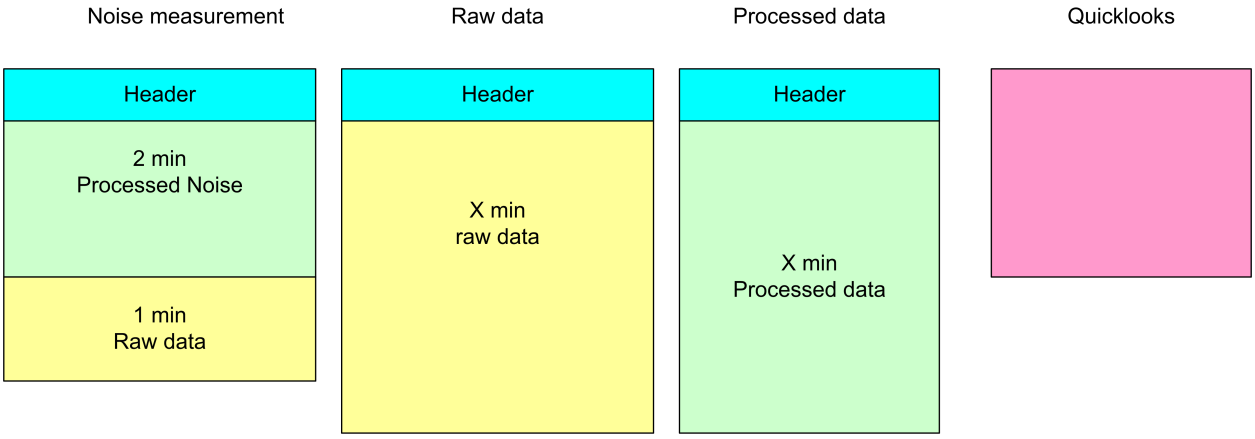


Figure 6.5: Four kinds of measurements

Beginning from left to right, the Noise-Measurement is done to obtain the information about the properties of the radar and environment that is needed in the processing and to be able to compare the measurement taken by TARA with other radars. To obtain this information the radar collects data without transmitting a signal. The measurement contains a header, processed-noised, and one minute of raw data. The processed noise consists of variables calculated for two minutes of data and the raw-data is simply another minute of unprocessed data.

During the Raw-Data-Measurement the radar will transmit the frequency sweep. This measurement contains a header and unprocessed (raw) data. This measurement collects a huge amount of data and is used when the user wants to collect all the information contained in the received signal. This is done because when raw data is processed some information content is highlighted at the cost loosing other content. So if the full content is wanted a Raw-Data-Measurement is needed. This measurement will take as long as the user wants.

Just like in the previous measurement, a signal is being transmitted during a Processed-Data-Measurement. The attributes of this measurement also have a header but for the rest only contains the collected information

that has been processed by the data-processor. The data collected by these measurements contain only the content extracted by the data-processor; therefore the size of this measurement in storage space is less than that of a raw measurement. Then as last there is the Quicklook. This contains a simple picture file which can be used to quickly lookup a measurement when needed. Of these four measurements the Noise-Measurement, Raw-Data-Measurement and Processed-data-Measurement all have a header. This header contains the reference information about the location and radar settings of the measurement. This is done so that people are able to interpret the data from a measurement a long time after the measurement. The Quick looks have no need for a header because they consist of a simple picture file containing the reference information in the picture.

The Beat-Signal

As discussed before the processing and the combining of the different data packets are done by the processing-group of the project. However the data that the processing group needs to process and store is generated by components within the Radar-Control. This signal is supplied in the form of the Beat-signal, of which the content can be seen in figure 6.6.

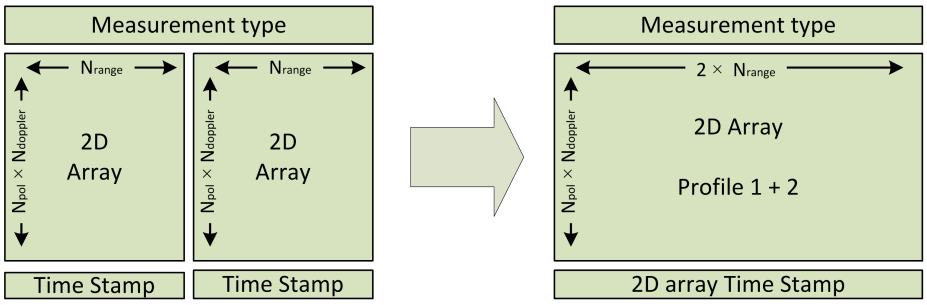


Figure 6.6: Beat-Signal content

The VI responsible for creating the beat-signal is the ADC that collects this signal in a profile. A profile contains a time stamp that gives the time of its creation and a 2-dimensional array filled with the values of the raw data in a specific way. The size of each profile depends on the length of N_{range}, $N_{doppler}$ and the number of transmission cycles and the number of polarization states used in one measurement cycle (N_{pol}). After generating one profile the ADC generates another profile and combines both of these and sends the one to the data-processor and/or storage.

The main difficulty for the radar-control in this context is to produce correct profiles. To do this a measurement VI was designed which synchronizes the TARA, the PXI, ADC and the DDS in an exact way so that together they manage to perform the different measurements above.

The Measurement VI

This Block Diagram seen in figure 6.7 is the Measurement VI and is responsible for controlling the DDS-Mod, Timing-Module, ADC, and TARA-Lines-Setup VIs. The functions of these VIs are as following:

- the DDS-mod configures the DDS in the wanted setting
- the Timing-Module generates the triggers to synchronize the transmission sequence and the Analog-to-digital conversion
- the ADC configures the hardware of the ADC and generates the Beat-signal

- the TARA-Lines-Setup sets RF_enable in the TARA to enable or disable transmission.

More details about these VIs are explained in the Appendix Block Diagrams.

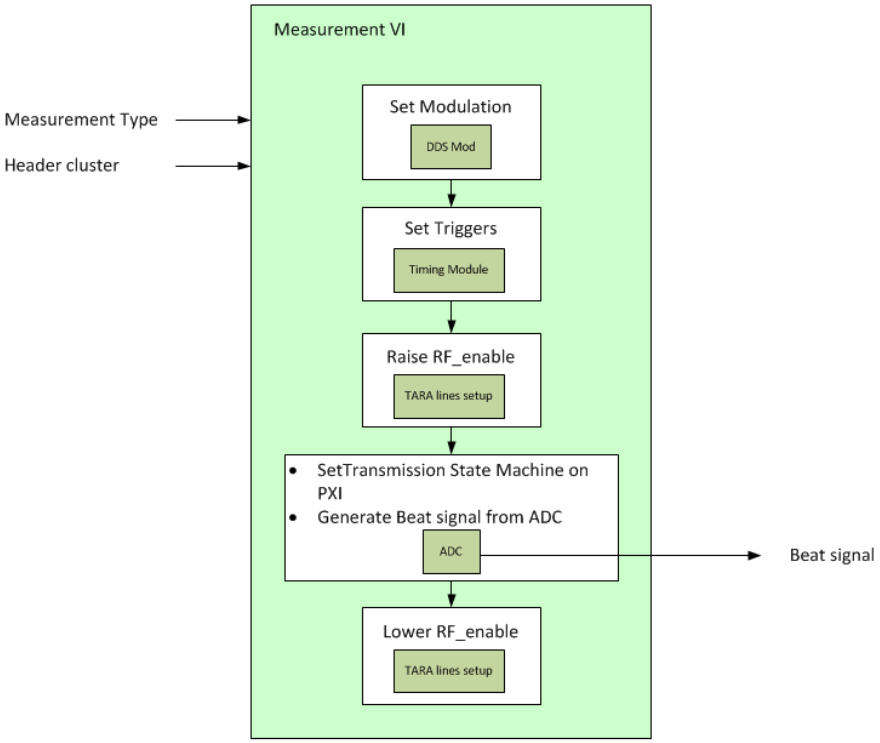


Figure 6.7: The measurement VI

In Figure 6.8 the three measurement modes that this VI can produce are shown together with the configuration of the DDS, Timing-Module, ADC and TARA-Lines-Setup during the measurement.

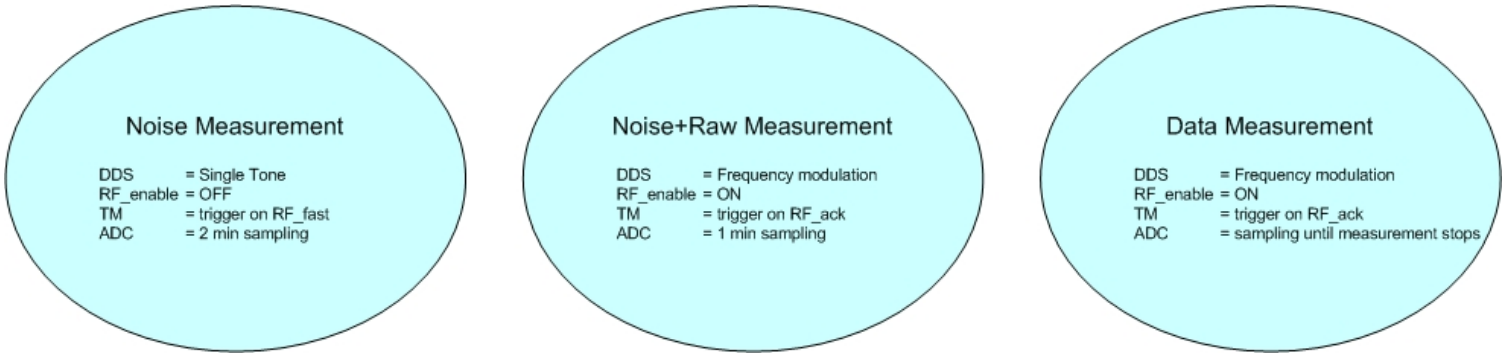


Figure 6.8: The three measurement modes

As shown at the left in figure 6.8, during a Noise-Measurement, the DDS is in single tone mode and the RF_enable is not raised as in 6.7 but remains low, meaning that no transmission occurs during this

Measurement. The Timing-module is generating the needed triggers and the end of a sweep is detected by RF_fast instead of RF_ack, both signals will be explained in a later section. To be able to switch on transmission both RF_enable and RF_fast have to be high. Because RF_enable remains low during this entire measurement, the fact that RF_fast is switching no problem. The ADC is continuously triggered until 2 minutes of profiles have been gathered. The trigger used for this is a signal called the ADC trigger. This signal will be generated in such a way as to ensure a good measurement. The details of the trigger signal will be explained in the next section. The number of profiles that will be generated is calculated in advance with the following formula:

$$N_{profiles} = \frac{t_{noise\_desired}}{N_{dopp} \times N_{pol} \times t_{sweep}} \tag{6.3}$$

This ensures that the ADC VI knows how many profiles need to be constructed to have a measurement of a certain time.

The Noise-Raw-Measurement is used to collect the data needed for the Raw-data part of a noise measurement. During this measurement the DDS is in frequency modulation mode and will start a frequency excursion when triggered to do so. The signal used to start this frequency excursion sweep is called the DDS trigger and is generated by the Timing Module. Because the frequency sweep is transmitted, TARA-Lines-Setup will set the RF_enable to high after the DDS is set so that transmission can be started by raising RF_fast to high voltage. The ADC will be triggered as long as it takes to generate 1 minute of profiles.

The Data-Measurement is the measurement used for collecting real data about the atmosphere. These measurements are continuous and take as long as the user has set with user input. During the DDS is sweeping and the sweep is being transmitted just like in the Raw-Noise-Measurement, the ADC will be triggered continuously until the end of the measurement.

Triggers

In the last section a lot is said about the different triggers produced within the system to indicate when a process should be started or stopped. Figure 6.9 gives an overview of the triggers used in the system.

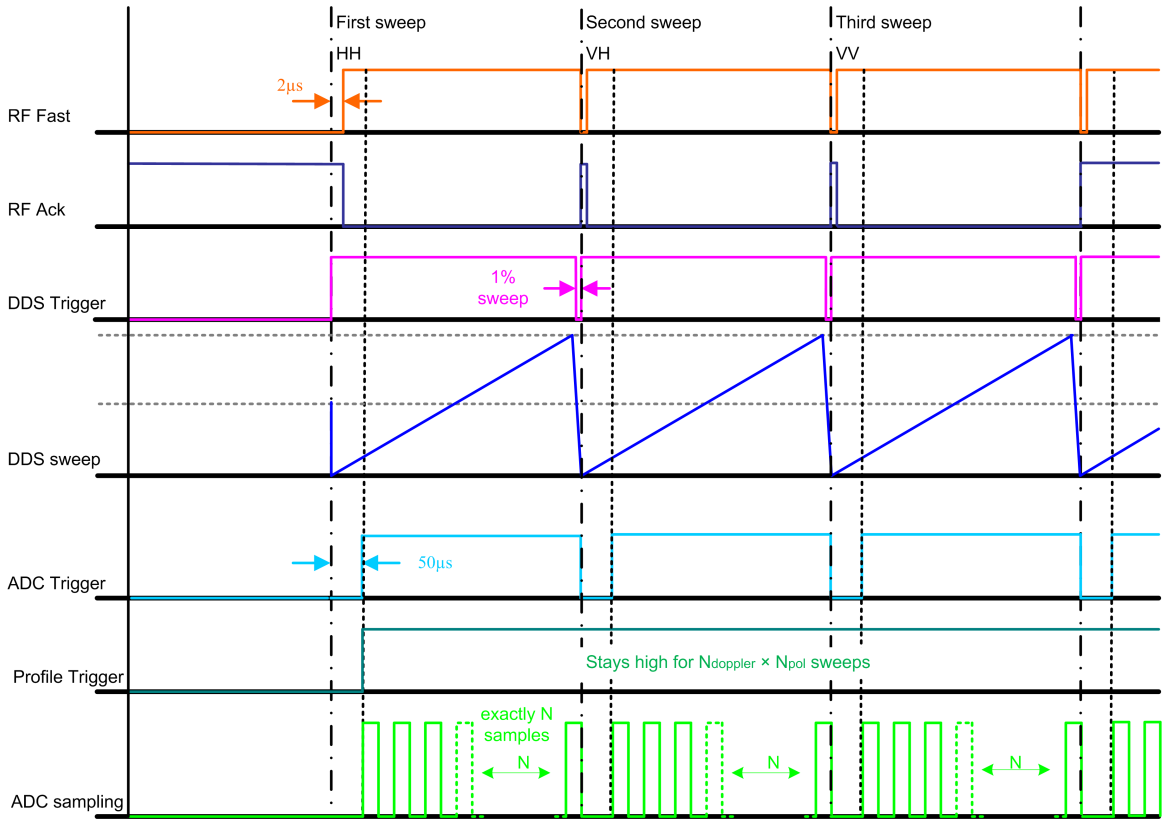


Figure 6.9: The three measurement modes

For a proper sequence of events, the Timing module produces four trigger signals and receives one:

- RF_fast (produced):
  This signal controls transmission. When this signal logic low there is no transmission, when it's logic high transmission is on at the condition that the RF_enable is also high.
- RF_ack (received):
  This signal is received from TARA, when this signal is logic high it means that there is no transmission in progress. At this stage a new polarization or antenna beam can be set. The antenna beam and polarization must only be set when there is no transmission in order to prevent damage to the switches.
- DDS trigger (produced):
  This signal triggers the DDS to start the frequency excursion. It is high for the first 99% of the sweep time. When the signal gets high the sweep starts at the bottom frequency up to the top frequency. When the signal gets low the sweep ramp down very fast.
- ADC trigger (produced):
  On the high flank of this signal the digitizer starts sampling Nrange samples from the receiver. When the signal equals a logic low, the ADC does not sample.
- ADC profile trigger (produced):
  This signal is needed in order to start sampling for each profile with the same polarization/antenna

beam. When this signal and the ADC trigger are both logic high, sampling is allowed. This has been done because after each profile is created it takes some time to start making a new profile which causes the ADC to start sampling in wrong sweep.

Transmission State Machine

The transmission state machine is used to switch between the different settings needed for all possible transmission and reception configurations of the TARA. It is shown in figure 6.10 The End-Transmission state represents the state in which the state machine resides if it is not used. When the Timing-Module is started, the State machine goes to the Set-Transmission-State. This is the state in the middle in which there is no transmission, allowing the configuration of the TARA to be changed without any risk of damaging the switches in the radar. Damage occurs when switching occurs while power flowing through them. As was told in the Hardware section, it takes less than 1 μs to change the setting of the TARA, therefore the state machine takes 1 μs plus 1 μs margin to change to one of the 6 transmission states. These transmissions states differ from each other in either the receiver/transmitter setting or the feed that is being used to transmit the signal. When the frequency sweep is completed the RF_fast switches to low and thereby ending transmission. This removal of power behind the Tx switch is detected by a power detection circuit which sets the RF_ack signal high. The rising flank of this signal is used as a trigger for the state machine to go to the Set-Transmission-State where the TARA setting can be safely set in another configuration because there is no power being transmitted. In the case of a Noise-Measurement there is no signal being transmitted during the time, data is being collected for the processed noise segment. Therefore the RF_ack signal can not be used during this process. But because RF_enable is off during this measurement which ensures that no power will be transmitted, the falling flank of the RF_fast signal can now be used as the trigger go in the Set-Transmission-State. The transmission state machine is loaded into the PXIe-6535 at the start of a measurement where it is active until the end of the measurement.

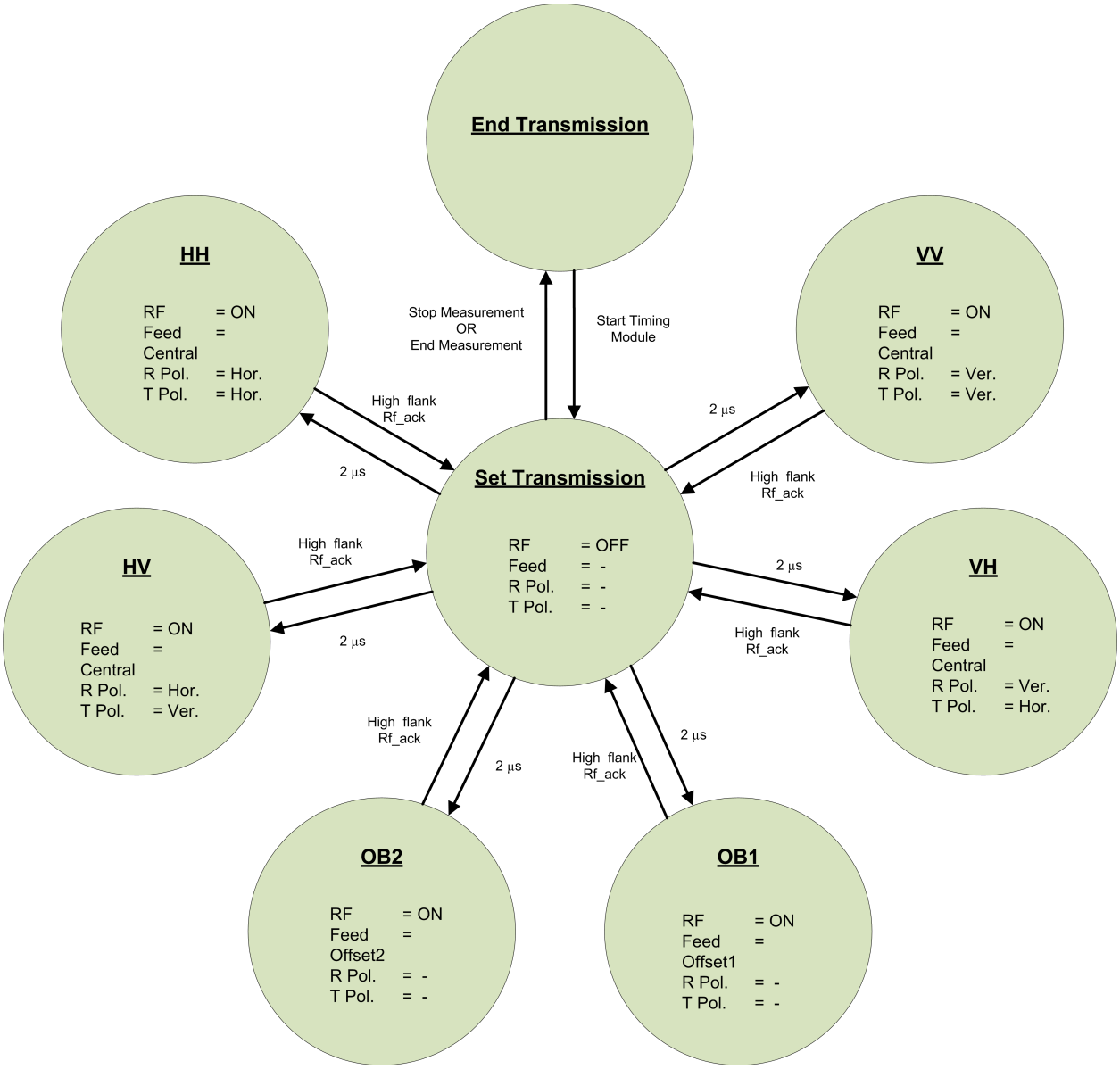


Figure 6.10: Transmission state machine

Calculating Needed Variables

The ADC, DDS and TM all have to be synchronized with each other in such a way as to optimize data collection during each sweep. In order to do this, variables have the be calculated at the beginning of every measurement which stay constant during the measurement. These variables then can be used to make triggers in the Timing Module, and setup the digitizer and DDS in the proper way. This process is needed because of three reasons. The first one being the fact that there is some time needed for the transmission signal to reach steady state after the transmission is switched on. Therefore sampling only can be done after 50 μs, this time includes quite a margin but this ensures that the transmission is in steady state when the sampling starts.

Then the ADC, DDS and TM all have to be configured in such a way that the samples are evenly spread

across the sweep and that no sampling occurs outside the sweep because then synchronization will be lost and the data from the measurement are unusable.

The last reason is that the ADC only has certain possible sample frequencies as shown in formula 6.4. So to be able to spread the samples and ensure that non are taken outside the sweep, the 50 μs needed for steady state is changed according to the need of the measurement.

$$f_{sample} = \frac{60\frac{MS}{s}}{n}, 4 < n < 1200 \tag{6.4}$$

Figure 6.11 will be used to explain in which way the variables needed are calculated and their physical explanation is depicted in the figure as well. The first part of the calculation, indicated with the number 1 in the figure, is to compute the optimum sample frequency. This can be done using the following formula:

$$f_{ADC\_clk\_op} = \frac{N_{Range}}{t_{sweep} - t_{setup\_op}} \tag{6.5}$$

The optimum setup time equals 50 μs, which means that now the exact number of samples can be taken from here on and it will also stop sampling within the sweep. However, this sample rate can not be used because it probably does not match the boundaries formula 6.4. In order to do get a sample frequency supported by the ADC, formula 6.6 must be used. This formula gives the sample rate closest to the optimum rate and always picks the one higher than the optimum rate so that sampling does not happen outside the sweep. This can be seen in the figure indicated with the number 2.

$$f_{ADC\_sample} = \frac{f_{ADC\_osc}}{floor(\frac{f_{ADC\_osc}}{f_{ADC\_sample\_clk\_op}})} \tag{6.6}$$

This sample rate calculated will now be send to the ADC to use during measurements. The next part of the figure (indicated by the number 3) shifts the sampling to the end of the sweep because then for every measurement the end of the sweep will be sampled. This can be done by adjusting the setup time so that the sampling starts later. This can be done using formula 6.7 in combination with formula 6.8.

$$t_{sampling} = \frac{N_{range}}{f_{ADC\_clk}} \tag{6.7}$$

$$t_{setup} = t_{sweep} - t_{sampling} \tag{6.8}$$

Formula 6.7 computes the actual time it take to sample, formula 6.8 then computes the setup time. Hence, all the samples are evenly spread during the sweep and the optimum has been reached. This setup time will be used by the Timing Module to make a trigger used by the ADC to start sampling.

The last part of the figure indicated by number 4, shows the sweep that is produced by the DDS when set correctly. Because the excursion frequency desired by the user is not the same anymore as the actual sampling starts after a while when the sweep is already started, an additional frequency for the excursion must be calculated to still sample the desired excursion. This is done by using formula 6.9.

$$f_{add} = f_{exc} \times \frac{t_{sweep}}{t_{sampling}} - f_{exc} \tag{6.9}$$

With these values calculated the DDS can be set. In order to do this some values need to be computed which are shown in the lowest part of the figure as well, namely the FTW, CTW1, RDW, RSRR, FSRR, FDW. The way this is done can be found in the Appendix LabVIEW under the section Calc Matlab. The Matlab code used in this section can be found in the Appendix Matlab code.

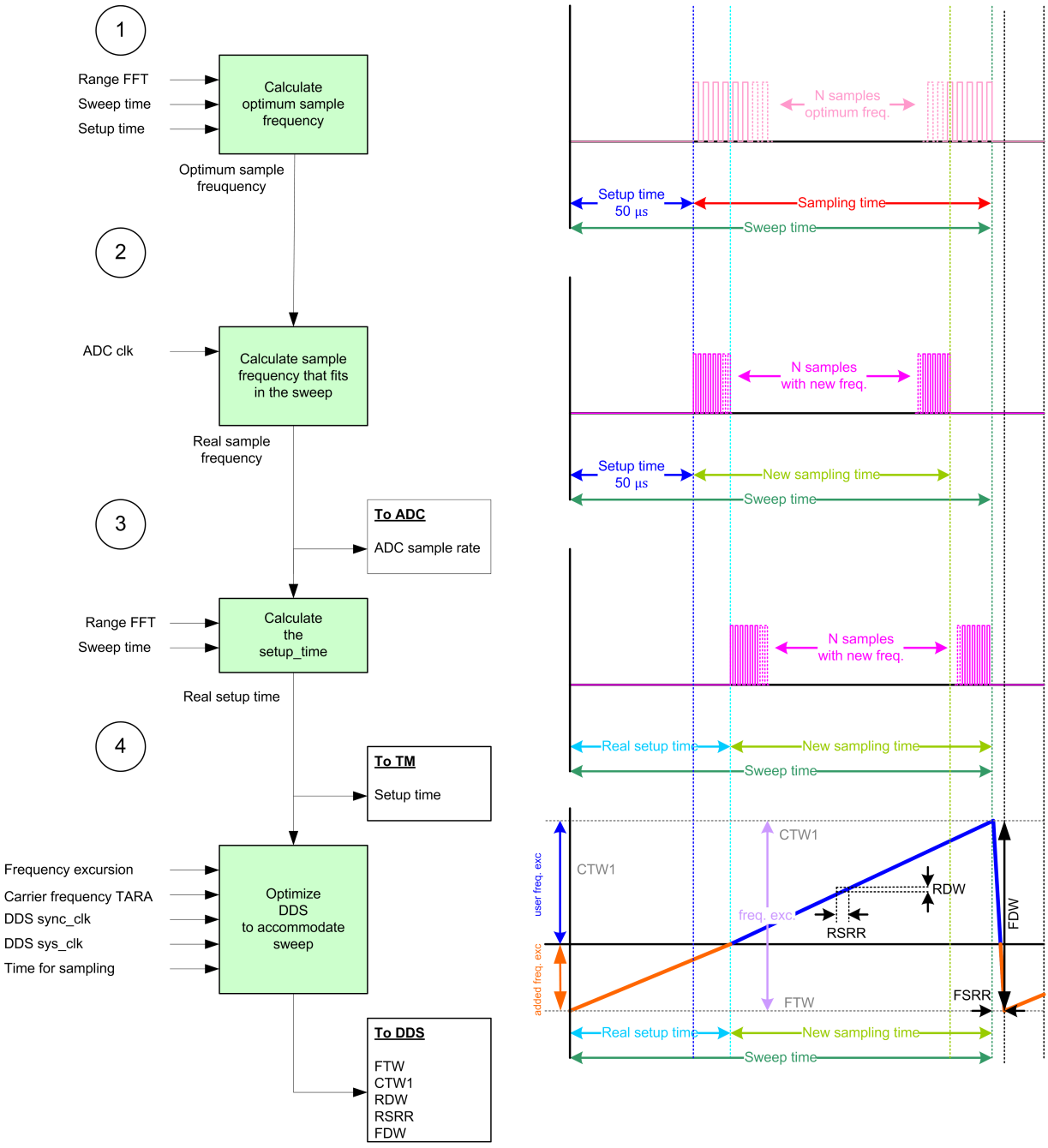


Figure 6.11: Explanation of Calc Matlab

Timing and Synchronization

All of the signals that are described in the section above are listed in the figure below (figure 6.12) to get a complete overview of the timing and synchronization for a noise and data measurement. At the left the signals are shown and in de top the current state of a measurement.

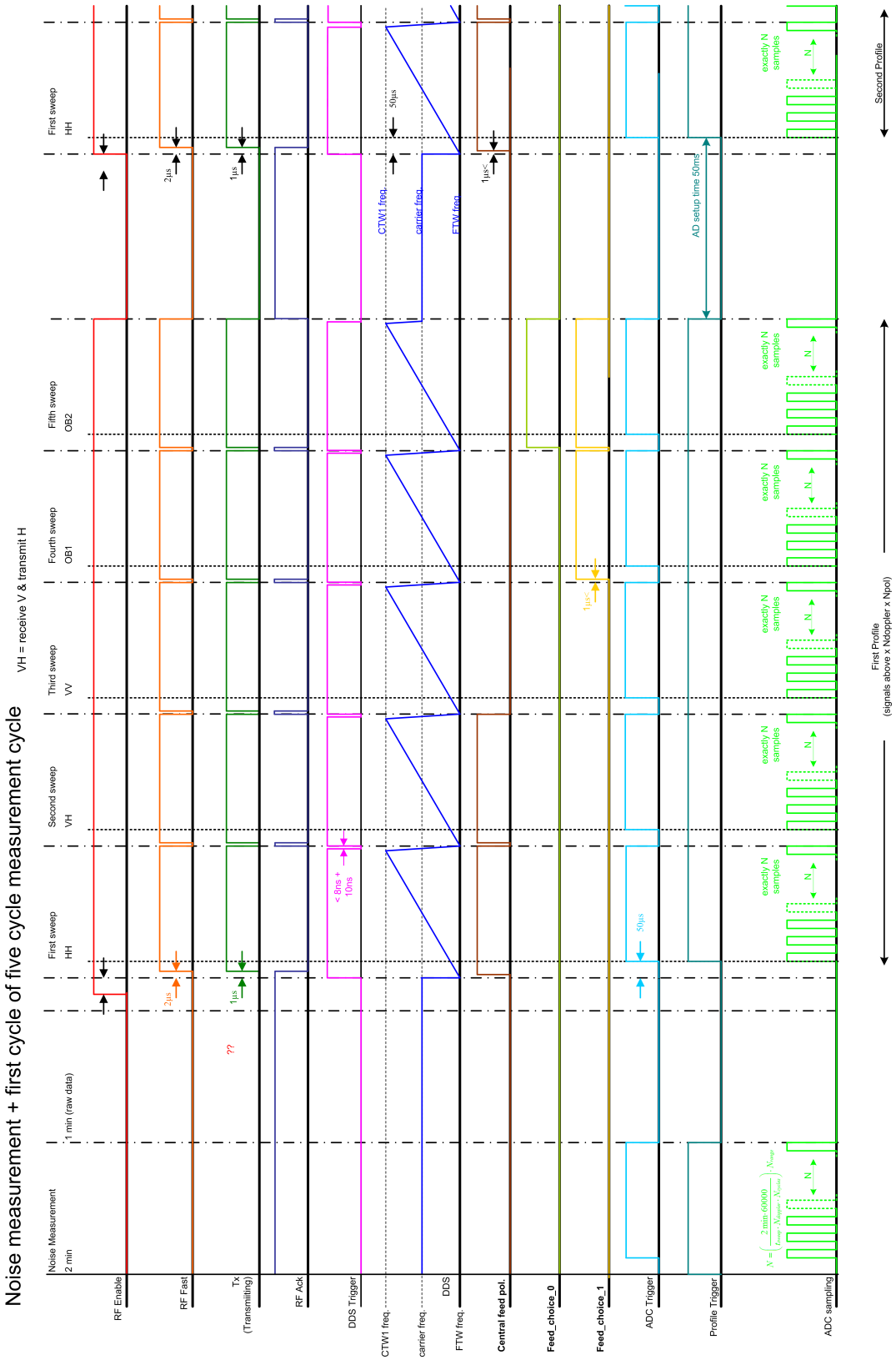


Figure 6.12: Signal Diagram

Chapter 7

System Evaluation

In this chapter the results of the project are evaluated by looking at some measurement results (obtained during the implementation of the new product within the TARA system at the CESAR observatory in Cabauw) and by reviewing the program of requirements. First, in section 7.1, the measurements, acquired during the product implementation day will be presented. After this overview, all the requirements made in chapter 3 are checked. In section 7.2 the product use is discussed. After that in 7.3, the environment of the system is evaluated. Then in 7.4 the requirements for system design are discussed and, finally, in 7.5 the production of the system is evaluated.

7.1 Measurement results

On Tuesday, June 6th 2011, the ATMOS group, consisting of three supervisors (Christine, Tobias and Yann) and two technicians (Fred and Paul) went to the CESAR site accompanied by all five students of the BAP group 6 (Enzo, Ester, Jeroen, Martijn and Satoshi). The main purpose of the visit was the final implementation on site of the first product designed and delivered by the group. Forecasts expected no rain for this day, which for this occasion was off course not ideal for doing and testing precipitation measurements. In the beginning there were some difficulties with connecting some of the cables between the computer and the TARA interface but these were solved quite fast. After some testing, the first measurements were performed (pictures can be found in Appendix measurement) using a three polarization cycle with the main beam. Although there were no rain clouds present, the interferences from cars passing by could be measured. This is depicted in figure 7.2 were the cars can be seen as the red lines in the green on the second display. These measurements were performed by applying the none filtering processing code in the radar processor block (see the main block diagram figure in Appendix General Block Diagram). This specific processing allowed the user to evaluate the new system with real targets which would not be possible with the standard processing (the cars would have been filtered out in the standard one). Actually, the same measurements were found during the measurements earlier in the morning, using the old TARA equipment, as can be seen on the photo in figure 7.1. The similarities between the old and the new system proved that the system is working correctly.

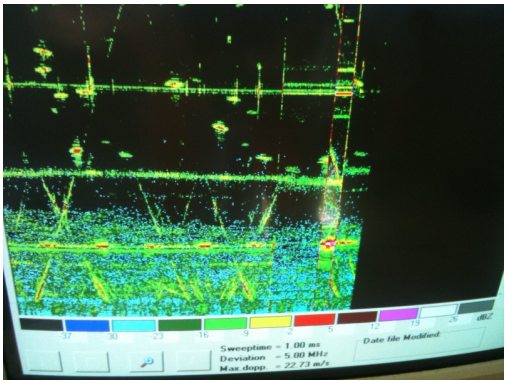


Figure 7.1: Measurement before installing the new system

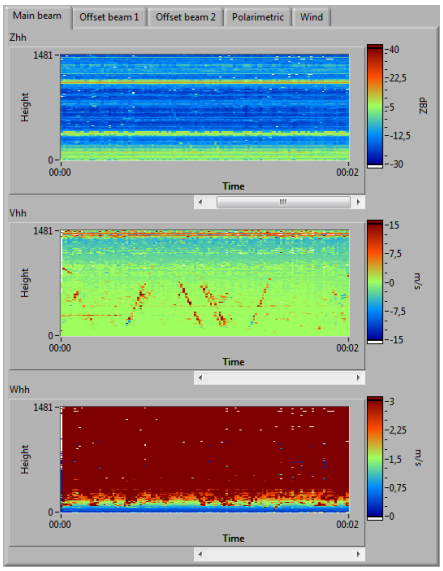


Figure 7.2: Measurement after installing the new system

7.2 Product use

The system performs weather measurements by the control of transmission and reception of EM-waves and it processes the acquired data with sufficient speed to display radar outputs in real-time. Users are able to select diverse measurement specifications depending on the type of observation to perform (i.e.: range resolution, doppler resolution, maximum detected range). The system is capable of supplying real-time data clusters (equal to a fixed amount of data) to the data processor. A timestamp is created in the UTC time format. First of all, the system carries out a noise measurement before starting the actual measurement. This noise measurement is required to correctly process the data in the next stage. The system makes a header according to the measurement specifications. The system is capable in acquiring raw data. The system allows for continuous measurements without user interaction. Furthermore, the system has the ability to save comments in the header. What is already made but still needs to be implemented is the System Control that is able to display and anticipate all the errors it experiences.

7.3 Environment

The software runs on a normal Windows PC. The used PC is a high end consumer PC running on the Windows 7 Professional 64-bit edition operating system. The product can easily communicate with the other parts of the system. During the testing, there was only one problem that one of the cables was not soldered correctly but that problem was quickly solved by soldering it in the right way. The system is required to run in a stable manner, but that is not completely the case. Every couple of times the software runs, the computer becomes unstable and the software program has to be restarted for an unknown reason. Supported by the feedback given by the ATMOS group, consisting of the future users, the software is easy to work with.

7.4 System design

Usage features

The design of the system makes use of an open approach, which allows for the easy implementation of new functions and simple debugging. Users are able to update the code easily and it is easy to implement future versions of the Matlab code used. The software is modular, but it could be structured a little bit better to allow for making adjustments with more ease. It is possible to use the block 'user input' to select the different transmission options using well-defined signals, although the settings for the attenuation and filter appear to work out wrong. The data processor outputs the measured radar parameters and measurement time. The block 'user input' is not yet able to choose which data it should store. This was not implemented, because of time constraints for the storage block. However, the already existing Storage block design provides an easy and quick opportunity to implement such a block in the near future. Finally, because the software is easy adjustable, the system requires very little maintenance.

Production, commissioning and liquidation

The software is modular and because the LabVIEW software can run on Windows, Linux and Mac, it is easily adjustable to other systems. It is possible for the user to update the code easily, due to the intuitive way LabVIEW works and because of the good documentation which has been provided during the project. Although it is worth to note that this documentation could be improved on some parts due to a lack of time. The code used for processing was given in Matlab and is easy to implement in LabVIEW. After installing the software, the system is ready to use immediately. For the largest part, the software was operational before the 7th of June 2011. The personnel available at the CESAR site and the people from the ATMOS group are easily able to install and un-install the software.

7.5 Production

For the production, all the requirements have been met. The software is built in LabVIEW, a standard software package for building computer based measurement systems. The PXI system is used for data collection and a DDS is used for generating the transmitted signal of the radar.

Chapter 8

Discussion

The project could be improved in several ways. This chapter will discuss the points that could be improved.

Time

This bachelor project requires a lot of time to complete successfully. This turned it into a challenging project. Because of this constraint it was not possible to make design choices or decisions that required a lot of time to implement , though it was possible to create a working system able to do measurements. If a few more weeks were added, other choices would also have been possible and more functionality would have been added. Functionality like the System Control or storage could also have been implemented and the attenuation control and low-pass filter select would have worked correctly.

LabVIEW

This language was chosen because many of the needed functionality was already supplied by National Instruments and because of it graphical nature would be easy to learn. Although it took some time to get used to the software, the fact that we managed to make a working radar and integrate all of the essential devices in seven weeks shows that our choice to work with LabVIEW was a good one.

Task Division

The way the work was originally planned to be divided was a bit off to what was actually needed. The control group (group 1) had to make a System Control, Radar Control and the overarching system. However, the Radar Control task turned out to be more difficult than expected which did not leave some time to make an overarching system. This task has been moved for a great part to processing group (group 2).

Chapter 9

Conclusion

The project aimed at creating a brand new design of the radar control unit of TARA, a 10 years old atmospheric radar. In order to design such a system, several blocks and state diagrams were embedded in the same common frame in order to create the overall structure of the system. First, the main block diagram was made with both project groups. This provided the basis to which the sub-blocks for the Radar Control part (tasks of sub-group 1) could be created. This first task required three full weeks. After that, the implementation in LabVIEW started. This took also a lot of time because every hardware component used in the system needed to be controlled via the VI's. After their creation, each sub-VI's have been tested separately. Finally, they have been connected to each other to get a complete working system. This was not as easy as expected, but eventually a full interconnected system was created.

For evaluation purposes, both groups went to the TARA site to implement the created system in TARA. After a few problems which were easy to solve, the first figures appeared on the computer screen. Most of the requirement of the system that was set at the beginning of the project has been met and could be evaluated on site. After some additional testing the result was clear: a working system.

As previously mentioned however, not all the requirements from the program of requirements were met. The reasons mainly arose from a lack of time and knowledge in LabVIEW for both the students and the supervisors. This resulted in loss of time and the fact that some of the functions that were hard to implement could not be realized on time.

Nonetheless, it is worth to notice that the quality of the work provided was considered high enough by the user so that they can use it to improve and finalize the radar system in the near future based on the design and what has been achieved so far.

Chapter 10

Recommendations

For future use and according to the fact not all the requirements have been met, a list of recommendations have been made.

First of all, the system could be made easier to use by implementing the attenuation control and the low pass filter select. This is not the case at the moment, which leads to controlling these two manually. Both systems already exist in LabVIEW, but the setting that needs to send to TARA isn't correct as this was not clear from the available TARA documentation.

Furthermore, some of the VI's should be restructured in order to adapt the program in the future easier. The way it is now is good to work with, but improvements such as a better documentation within the VI's, are desirable.

Thirdly the System Control is not implemented in the final system yet. This System Control VI does already exist and has been successfully tested as well, but due lack of time it is not in the system which has to be done in the near future to be able to react and report occurring errors correctly.

Then, some sensors at the TARA site need to be installed who can measure the temperature of the feeds and the power that is used for transmission. The implementation in LabVIEW will be not that difficult because the System Control VI is already prepared for the implementation of these two sensors.

Finally, some of the time margins can be optimized. In the current implementation they are set larger than necessary for safety reasons. When more time is available for testing at the TARA site the margins can be reduced because then the exact required lengths of the margins will be known.

Bibliography

[1] Thesis of the processing group.

[2] M. Muller, "Climate change - clouds remain the misty factor" *Delft Outlook* vol. 3, pp. 22-27, 2007.

[3] Ronald E. Rinehart, *Radar for meteorologists*, fifth edition. Nevada, Missouri: Rinehart Publications, 2010, pp. 1-14.

[4] Herman Russchenberg. (2011, May) TARA, the S-band Transportable Atmospheric Radar. Available: http://home.tudelft.nl/en/research/knowledge-centres/irctr/research/radar/research-facilities/tara/

[5] Transportable Atmospheric RAdar TARA: User Manual, TU Delft, The Netherlands, 2005, pp 1-7.

[6] S.H. Heijnen, L.P. Ligthart ,"TARA: Transportable atmospheric radar", Oct 1998.

[7] S.H. Heijnen, L.P. Ligthart, H.W.J. Russchenberg,"First measurements with TARA: An S-band Transportable Atmospheric Radar", July 2000.

[8] S.H. Heijnen, L.P. Ligthart and H.W.J. Russchenberg,"Improvement of the performance of FM-CW", July 2000.

[9] L. P. Ligthart and S. H. Heijnen, Transportable multi-beam Doppler polarimetric radar, in Proc. 14th Int. Conf. Microw., Radar Wireless Commun., (MIKON), May 2022, 2002, vol. 2, pp. 334340.

[10] S.H. Heijnen, J.S. Van Sinttruijen, W.F. Van der Zwan, L.P. Ligthart, IRCTR, Delft Univ. of Technol., "A dedicated computer system for FM-CW radar applications"

[11] Ventura, J. Figueras, H.W.J. Russchenberg, Delft University of Technology-International Research Centre for Telecommunications-transmission and Radar (IRCTR), Mekelweg, 4, 2628CD Delft, THE NETHERLANDS. Email: j.figueras@irctr.tudelft.nl, "Improvement of the Performance of FM-CW Radar Systems by using Direct Digital Synthesizers: Comparison with Voltage Controlled Oscillators"

[12] Heijnen S.H. et al; 3-D wind measurements with the S-band atmospheric profiler TARA; AMS conference Munich; 2001;

[13] D. Spinellis; Dept. Manage. Sci. & Technol., Athens Univ. of Econ. & Bus. ,"Choosing a Programming language."

[14] Cesar observatory (2011, June 10). [Online]. Available: http://www.cesar-observatory.nl/index.php?q=5

[15] Prodrive (2011, June 10). [Online]. Available: http://www.prodrive.nl

[16] Technische Universiteit Eindhoven (2011, June 10). [Online]. Available: http://www.tue.nl

[17] Dell T1500 (2011, June 10). [Online]. Available: http://www.dell.com

[18] VI wikipedia (2011, June 10). [Online]. Available: http://labviewwiki.org/VI

[19] Functional global variables LabVIEW (2011, June 10). [Online]. Available: http://labviewwiki.org/Functional_global_variable

[20] State machines LabVIEW (2011, June 10). [Online]. Available: http://labviewwiki.org/State_machine

[21] Improvement of the performance of FM-CW Radar Systems by using Direct Digital Synthesizers: Comparison with Voltage Controlled Oscillators

[22] National Instruments, "NI PXI-6682H Timing Module", NI PXI-6682 Series User Manual 2009, [Online]. Available: http://www.ni.com

[23] National Instruments, "NI PXI-5922 16 to 24-Bit Digitizer", NI PXI/PCI-5922 Specifications 2008, [Online]. Available: http://www.ni.com

[24] National Instruments, "NI PXIe-6535 10 MHz Digital I/O", NI PXIe/PCIe-6535/6536/6537, Specifications 2010, [Online]. Available: http://www.ni.com

[25] National Instruments, "NI PXIe-8360 MXI-Express", MXI-Express x1 Series User Manual and Specifications 2011, [Online]. Available: http://www.ni.com

[26] National Instruments, "NI PXIe-1082", NI PXIe-1082 User Manual and Specifications 2010, [Online]. Available: http://www.ni.com

[27] Analog Devices, "AD9958 2-Channel 500 MSPS DDS with 10-bit DACs", Data Sheet Rev A 2008, [Online]. Available: http://www.analog.com

Appendix A

Block Diagrams

Radar Control

Block Diagram

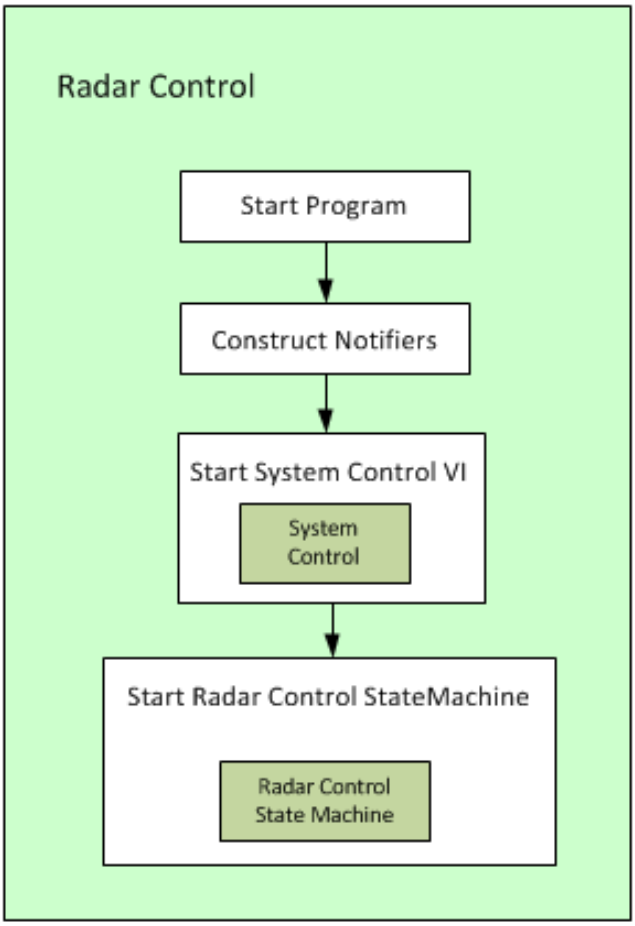


Figure A.1: Radar control

Function

The main program. This VI control the sequence and order of all other VIs in the program.

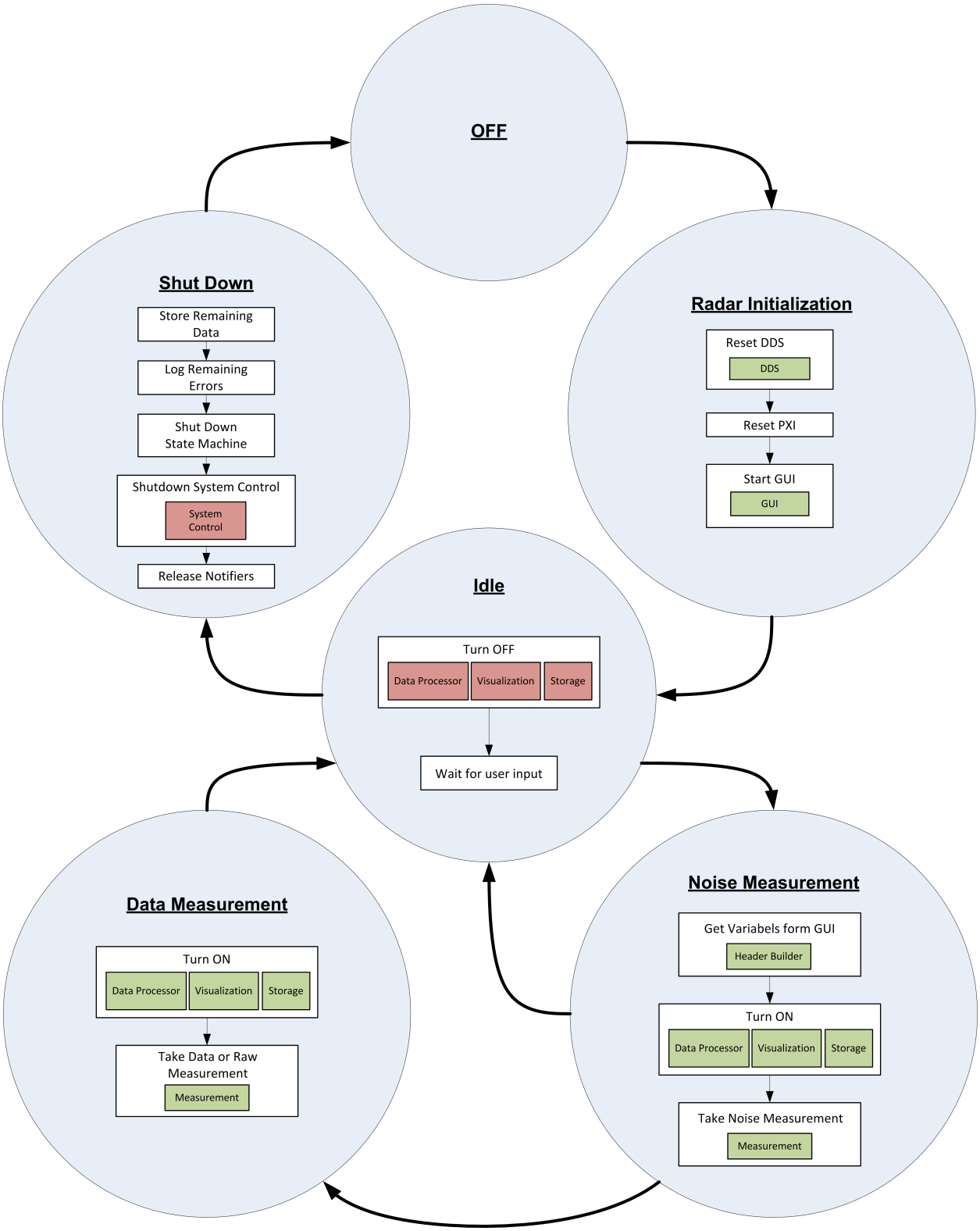


Figure A.2: Radar control

System Control

Block diagram

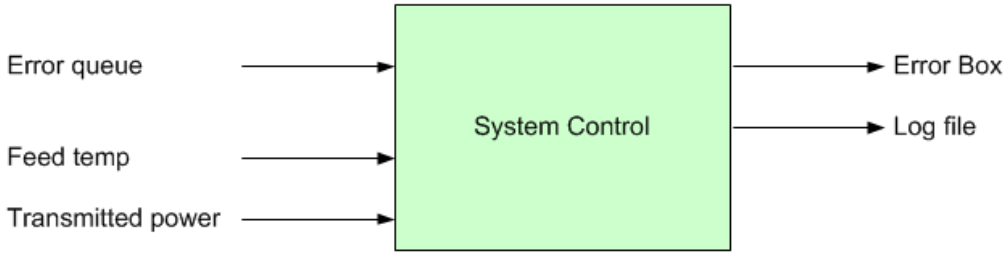


Figure A.3: Blockdiagram system control

Function

The System Control checks whether the system is running normal and checks for errors. It displays the latest 10 errors, creates a directory if necessary and writes all errors to a log file.

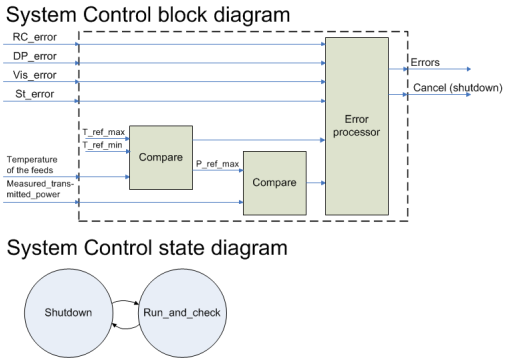


Figure A.4: Block- and state diagrams system control

Implementation

Figure A.4 shows the block diagram of System Control. The output consist of the error box and a log file. The input consists of a queue containing errors and two system parameters (Feed temperature and transmitted power). If an error enters the queue, it enters the system control block. Besides, if the feed temperature gets out of bounds or the transmitted power becomes too high, an error is generated internally in the VI named system_control.vi (see Appendix LabVIEW). According to the error code, a color is assigned to the error and temporarily saved with the latest 10 errors by the VI called colored_strings.vi. Those are

displayed in the error box which is displayed in the GUI. Besides this, it uses a VI called logger.vi to write a log file. This VI checks whether a directory with the current date exists and creates one in a subdirectory of the input base path if necessary. Then it writes to a log file with in the name the date and time when the measurement was started.

GUI

The Graphical User Interface where the user can select the wanted settings that are used in the other VIs.

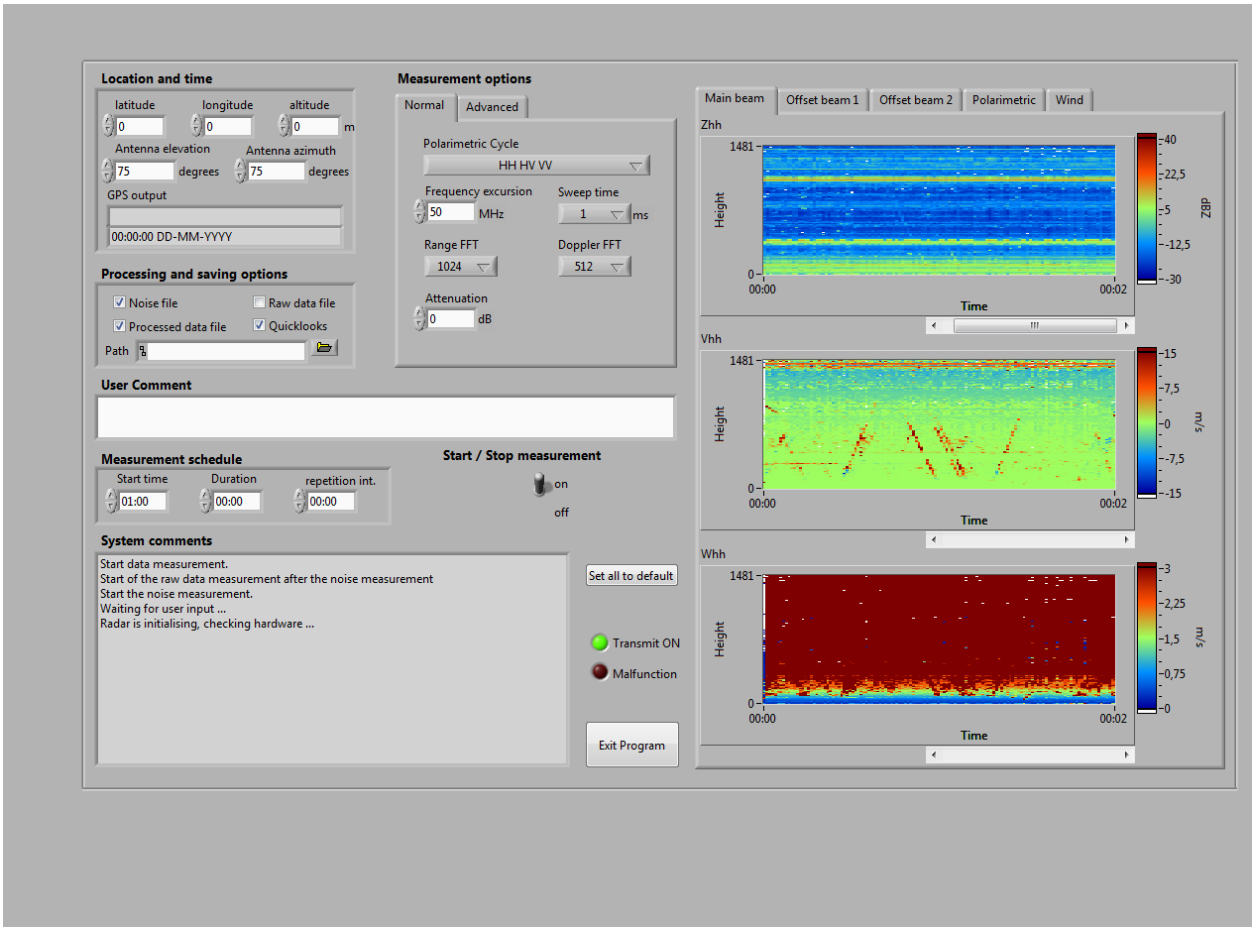


Figure A.5: Measurement with the GUI

Header Builder

Block diagram

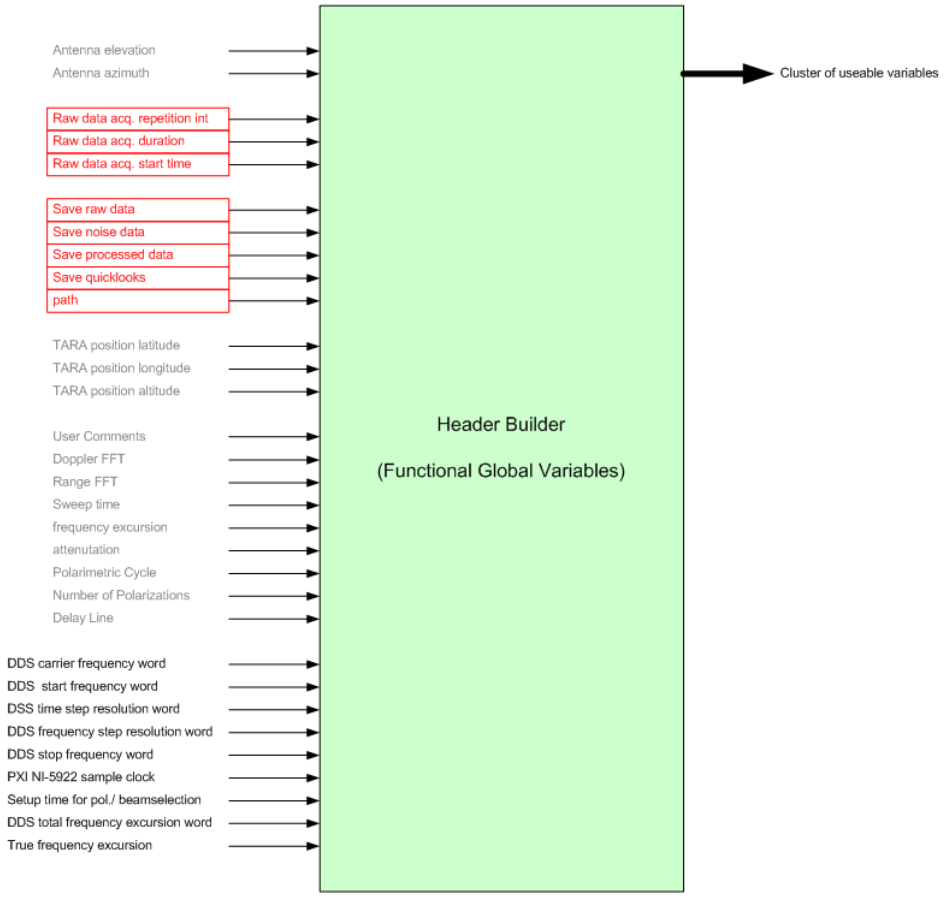


Figure A.6: Block diagram Header Builder

Function

The Header Builder VI is what they call in the LabVIEW world a Functional Global Variable VI. It gets all the measurement settings form GUI and the variables calculated by Calc-Matlab, bundles them together and stores them for use for all other VI's that need them. They remain static.

Calc matlab

Block diagram

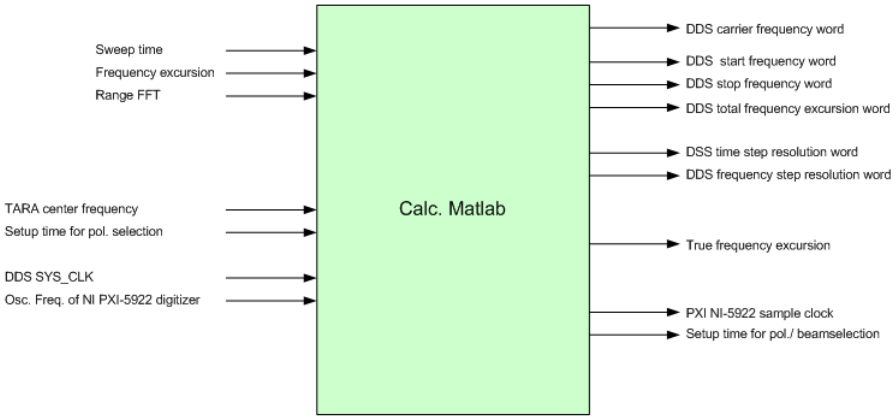


Figure A.7: Block diagram Calc matlab

Function

The function of this VI is to calculate the variables needed to configure the ADC, DDS and TM in such a way such that data collection is optimized during each sweep. These variables are calculated at the beginning of each measurement and stay constant during that measurement.

Implementation

This VI collects the needed variables from the Header Builder and starts to calculate the optimum sample frequency of the ADC. After that it calculates the corresponding inputs for the TM and the DDS. This process is needed because of three reasons. The first one being the fact that there is some time needed for the ADC to reinitialize the trigger and for the transmission signal to reach steady state after the transmission is switched on. Therefore sampling can only be done after 50 μs. These 50 μs include quite a margin but this ensures that the transmission is in steady state when sampling starts.

Then the ADC, DDS and TM all have to be configured in such a way that the samples are evenly spread across the sweep and that no sampling occurs outside the sweep because then synchronization will be lost and the data from the measurement will be unusable.

The last reason is that the ADC only has certain possible sample frequencies. So to be able to spread the samples and ensure that non are taken outside the sweep, the 50 μs is slightly adjusted as can be seen in figure A.8. This in order for the transmission to reach steady state and for the samples to fit exactly in the sweep. Because of the margin is taken as big as 50 μs no problems are expected.

After this adjustment the variables needed as inputs for the DDS and TM are calculated so that it accommodates the measurement in the proper way.

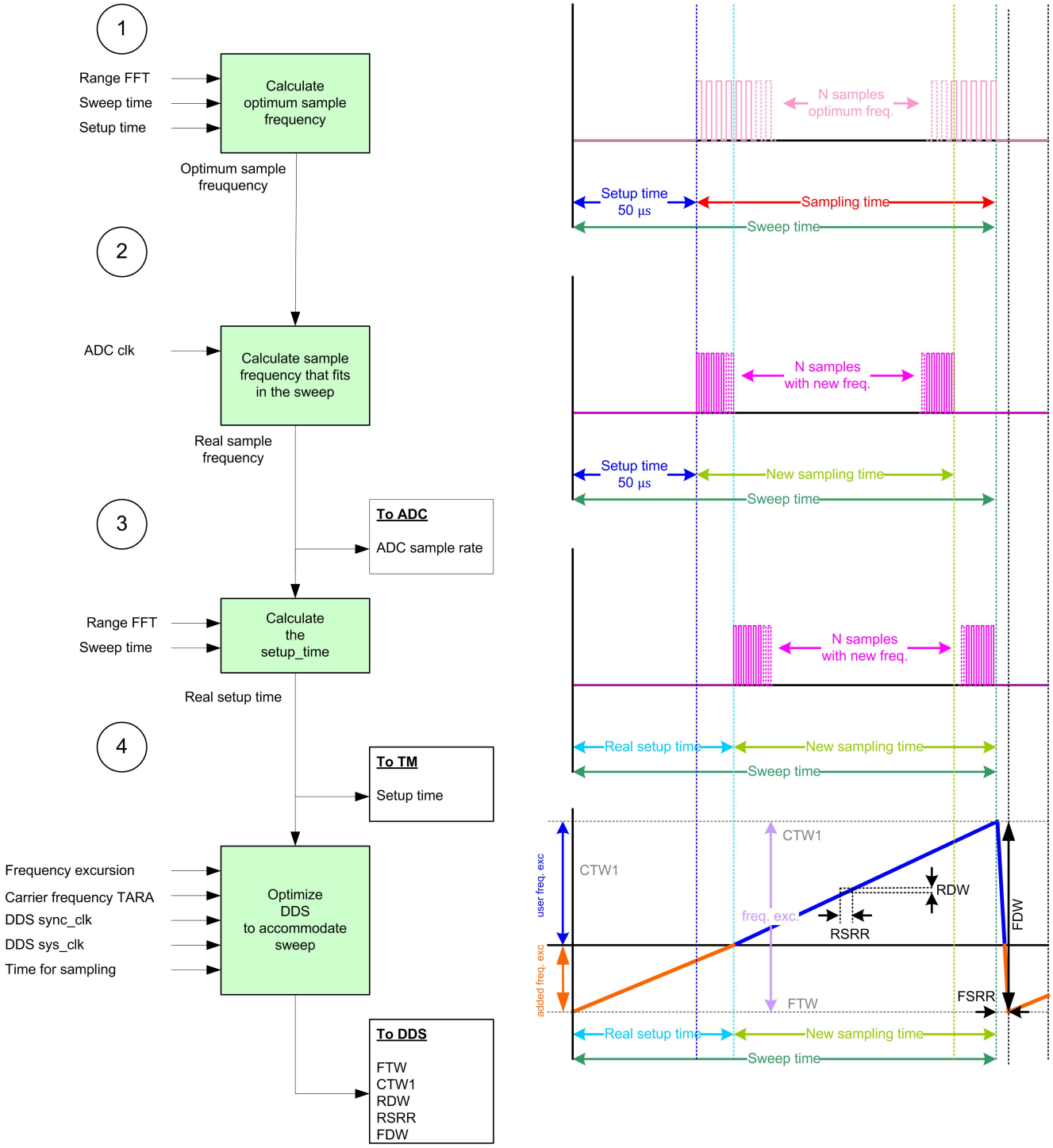


Figure A.8: Calculations for ADC,TM and DDS

Measurement

Block diagram

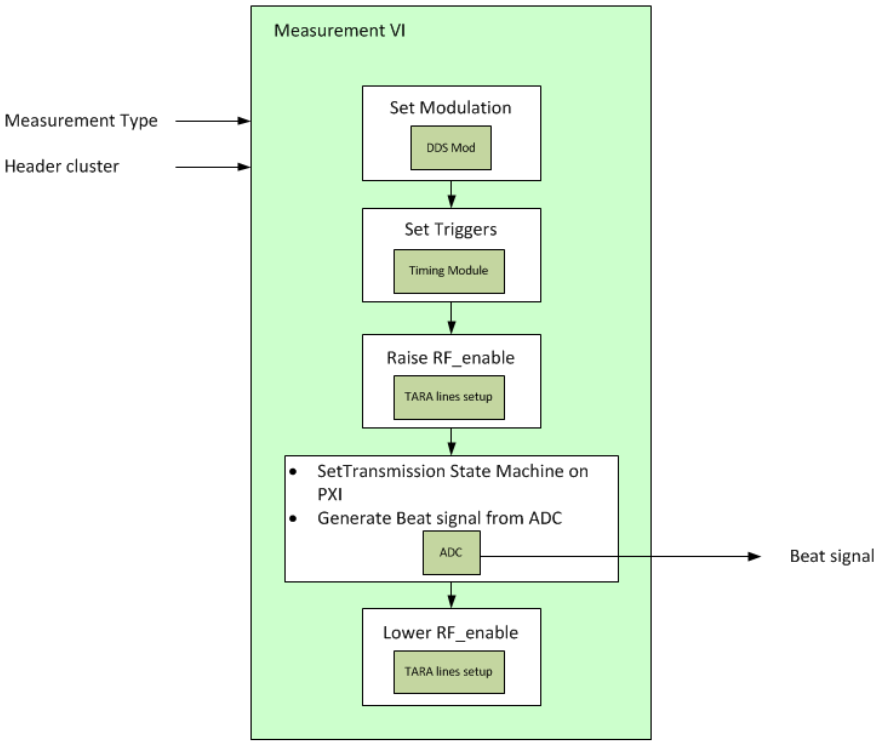


Figure A.9: Block diagram Measurement

Function

This is a VI that connects the DDS, TM, ADC and the TARA-Lines-Setup VIs together to get one measurement block which can be used to do the selected measurement:

- Noise
- Noise + Raw Data
- Processed Data

Implementation

Figure 7.8 shows the different settings possible according to the measurement types. As can be seen, the sub-VIs are set differently for every type of measurement.

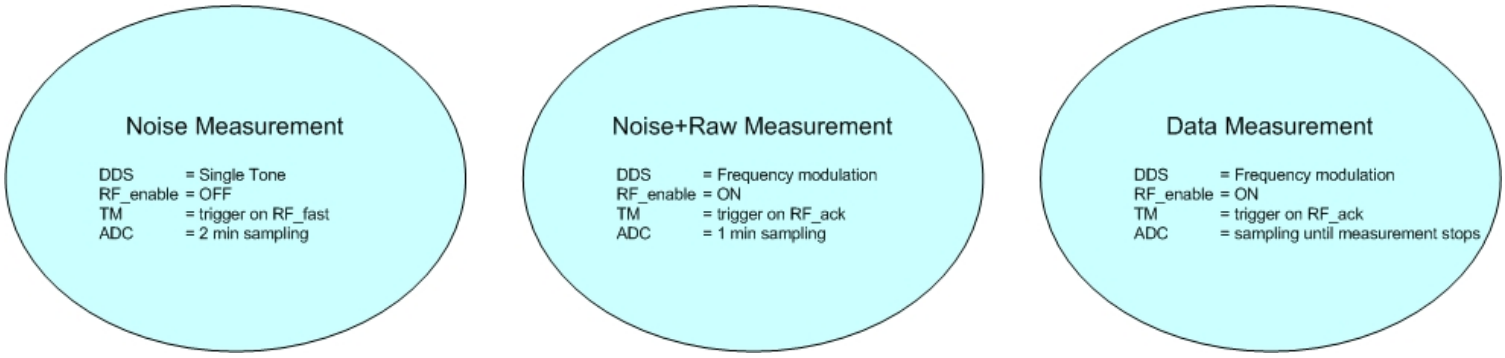


Figure A.10: Modes of measurement

DDS

Block diagram

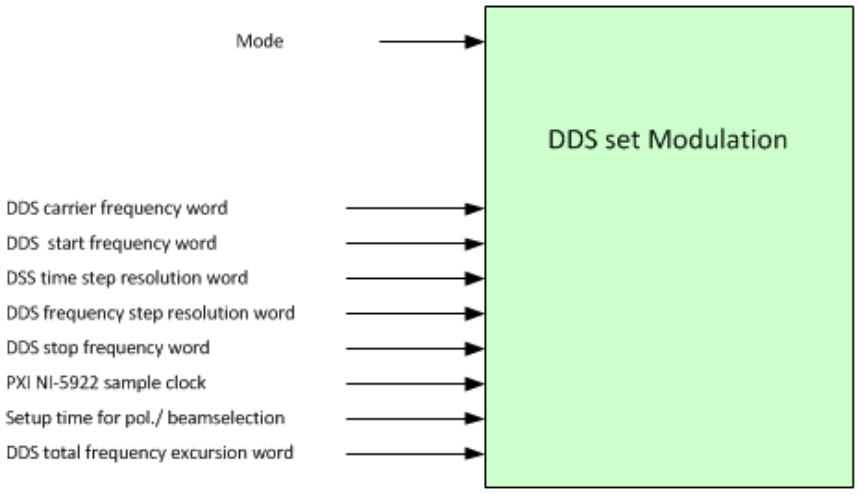


Figure A.11: Block diagram Measurement

Function

This VI programs the DDS in order to execute the desired measurement.

Implementation

This VI is mainly dependent of the measurement type. When a noise measurement is required it sets up a single tone at the carrier frequency of 160 Mhz. Otherwise, when a noise measurement with raw data or a data measurement is required, the DDS will be set to generate a frequency modulated sweep. The Calc matlab program is used to calculate the right values to be set in the registers.

Timing module

Block diagram

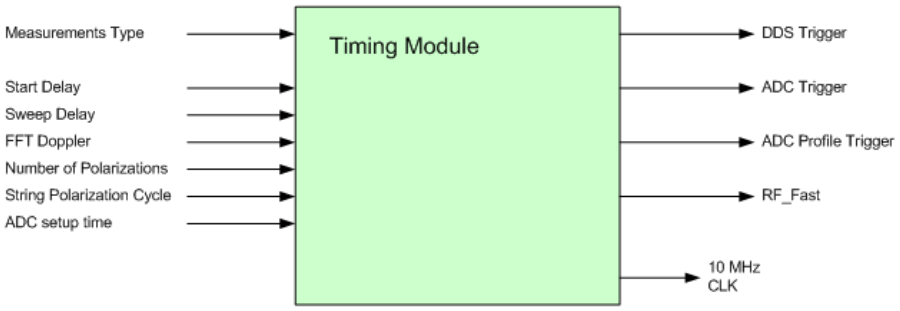


Figure A.12: Block diagram timing module

Function

This program controls the NI-PXI-6682H Timing Module card. It has three functions:

- Supplying the Master clock to which all other clocks in the system lock to ensure a good synchronized system; e.g. the oscillator, the internal clock of TARA and the ADC. This clock of 10 MHz is chosen for this task because it is the most stable clock available.
- Generating four triggers that are used to control the timing of the DDS, ADC and TARA to ensure the synchronization of the system.
- Controlling the setting of polarization to ensure that switching polarization is only done when transmission is off to prevent damage to the switches of TARA. This can be done by monitoring the RF_ack signal provided by the TARA.

Implementation

The Master clock is provided by the internal oscillator that is hardwired on the PXI-6682H to the clock out pin as can be seen in the datasheet [22]. This clock signal is then split to provide the clock to the three devices.

The four trigger signals are created in the VI with the specific settings according to the header and the Calc Matlab program. The DDS and RF_ack trigger are wired to the PFI pins on the front panel and the ADC triggers are both connected to the backplane of the TM.

The RF_ack signal from TARA is connected to one of the PFI pins on the front panel. This signal will be used as a trigger signal in order to set the next polarization while being sure that there is not any transmission in progress.

ADC

Block diagram

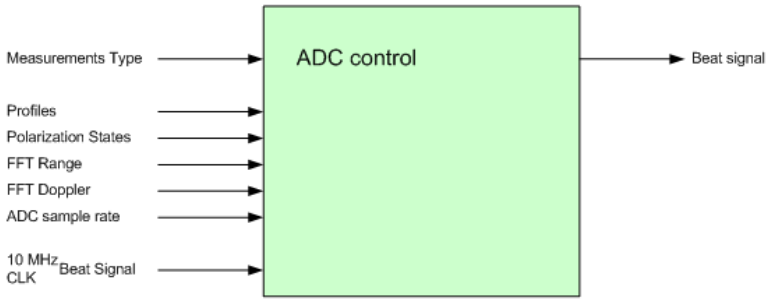


Figure A.13: Block diagram ADC

Function

This VI controls the NI PXI 5922 Digitizer for the sampling of the received data from TARA . It makes the beat signal and sends it to the data processor.

Implementation

The PXI is set to make a record of N_{range} samples at the given sample frequency. This record is created at every rising clock flank of the ADC_trigger and will be saved in the on-board memory of the PXI [reference]. After such a record is created it takes some time for the trigger to reinitialize, which is called the rearm time. Hereafter, the digitizer is ready to receive the next trigger. The memory of the PXI will be send to the computer when a total profile has been made by the digitizer. Such a profile consists of a number of records, calculated by formula A.1:

$$N_{records\_in\_profile} = N_{pol} \times N_{doppler} \tag{A.1}$$

The incoming data arrives in the form of a 2D array. The VI then acquires a number of profiles asked for by the user and puts the profiles into the data type named beat signal, together with the timestamp and the measurement type. The timestamp is created by the TM at every beginning of a new profile. When the beat signal is created it will be send to the data processing block. This process can be seen in figure A.14 where the number of profiles asked for by the user equals 2.

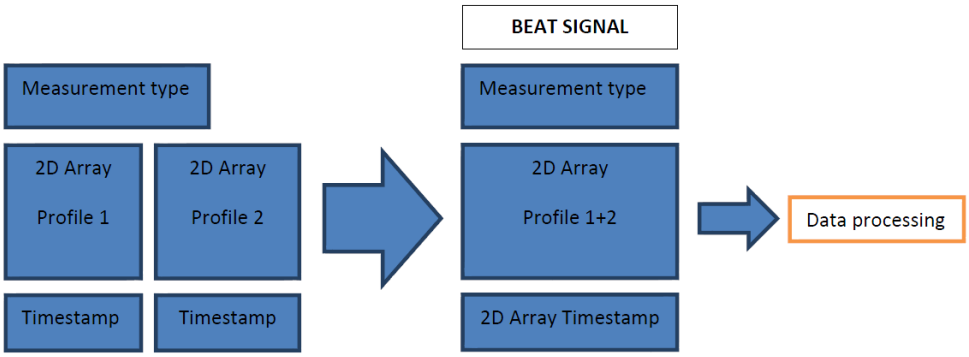


Figure A.14: Beat signal

A delay arises each time the profile from the memory of the digitizer is send to the computer. That is because LabVIEW has to put the data into an array and has to restart the ADC for the next profile. This takes some time which causes the ADC to miss some triggers. Because of this, it is not sure anymore whether the ADC starts sampling again at the beginning of a new polarization cycle. To prevent this from happening the ADC has to wait an additional time in order to start the record acquisition at the beginning of a new polarization cycle. Tests showed that it takes about 20 ms to fetch the data from the ADC memory and restart the ADC. Therfore, for safety this time has been set to around the 50 ms and is implemented into the system using another trigger named the adc_profile_trigger. When both this trigger and the ADC_trigger are high the ADC is starting to sample a new profile.

TARA lines setup

Block diagram

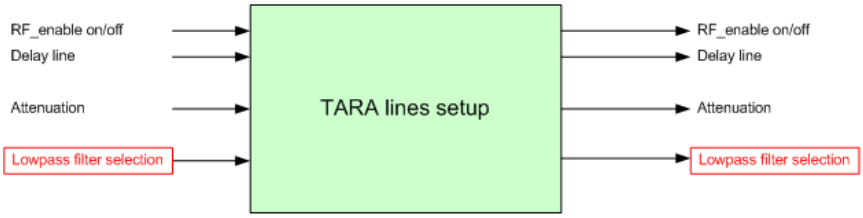


Figure A.15: TARA lines setup

Function

This VI is used to set the output value of RF_enable, attenuation, low pass filter and delay-line.

Implementation

The inputs will be send over the lines to TARA via the PXIe-6535 Digital I/O. It is a block controlled in the measurement VI because the RF_enable changes upon the different measurement types. TARA I/O (in Appendix F) shows how the Digital I/O is connected to TARA.

Portmap

Block diagram

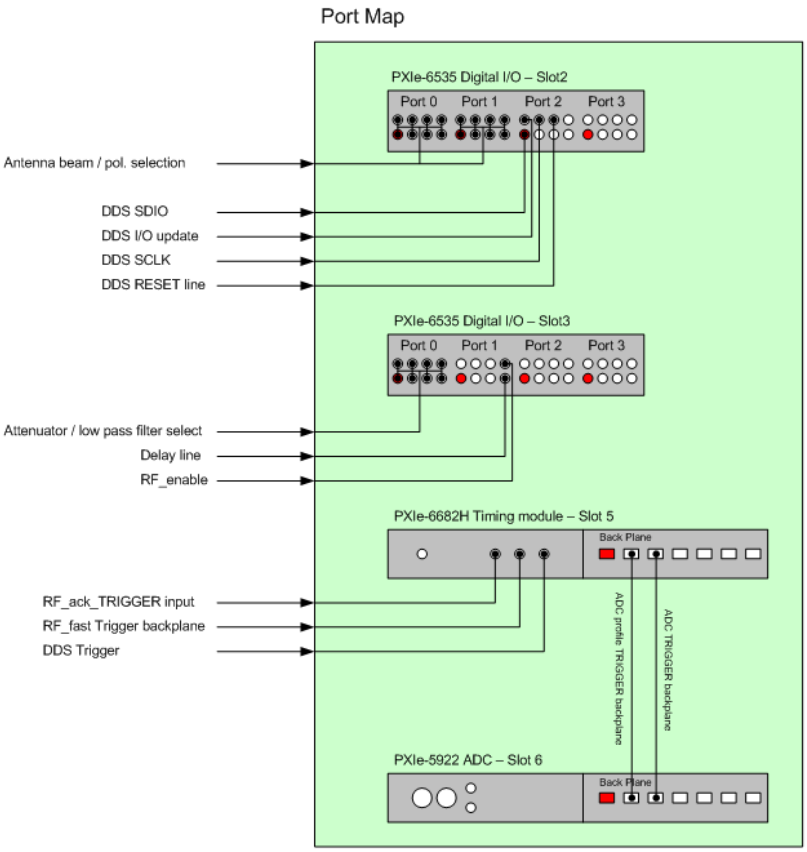


Figure A.16: Block diagram port map

Function

Is a VI in which all the trigger lines, data lines and used slots are defined. This is used to connect lines from the outer world to LabVIEW world.

Implementation

Consists of a number of constants in which the possible lines and slots of the different PXIs can be selected from a drop down menu. All these constants are bundled to a cluster and connected to the single output of the VI. Other VIs can import this portmap and extract their needed lines or slots to read or write data to it. This is an easy way to make changes in the line structure possible without having to change every VI separately.

Appendix B

LabVIEW Figures

B.1 System control

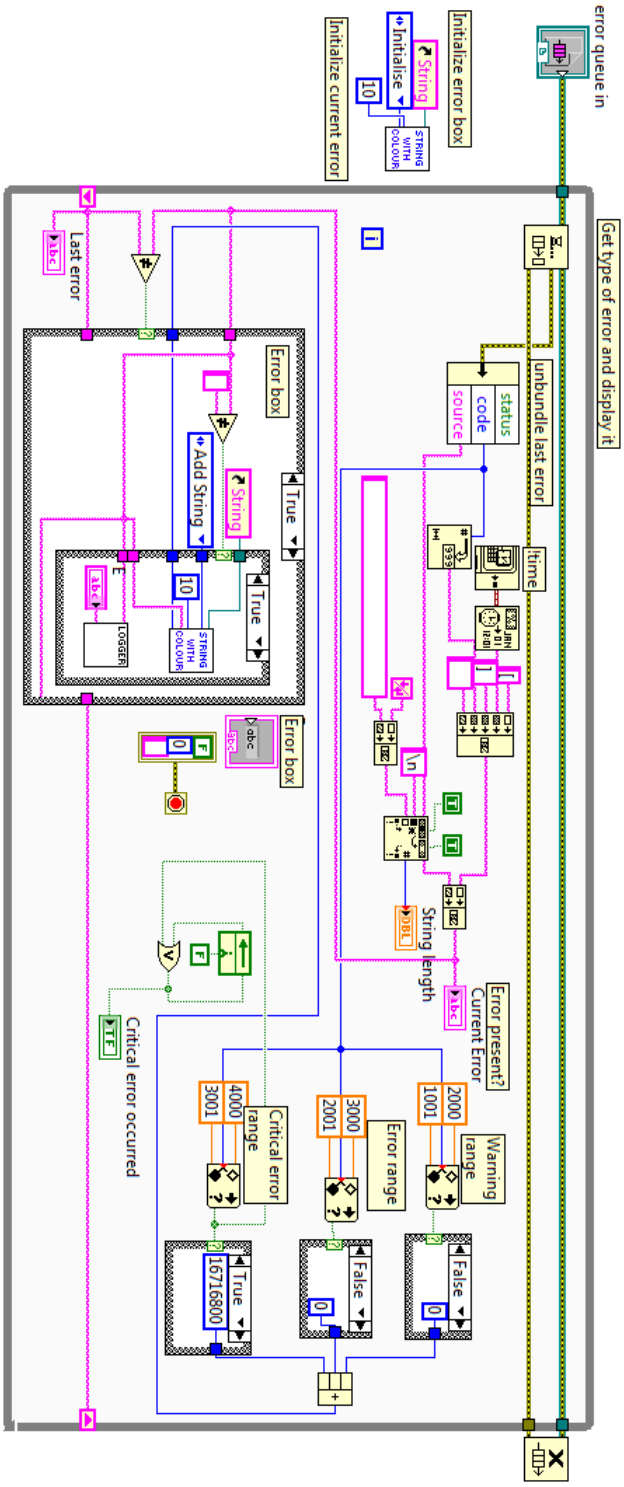


Figure B.1: System control VI - “Error Present” state

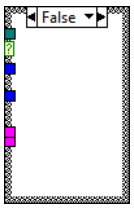


Figure B.2: System control VI - “No error Present” state

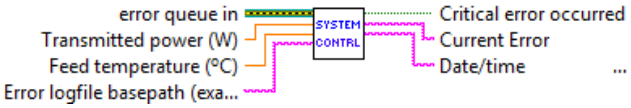


Figure B.3: System control icon

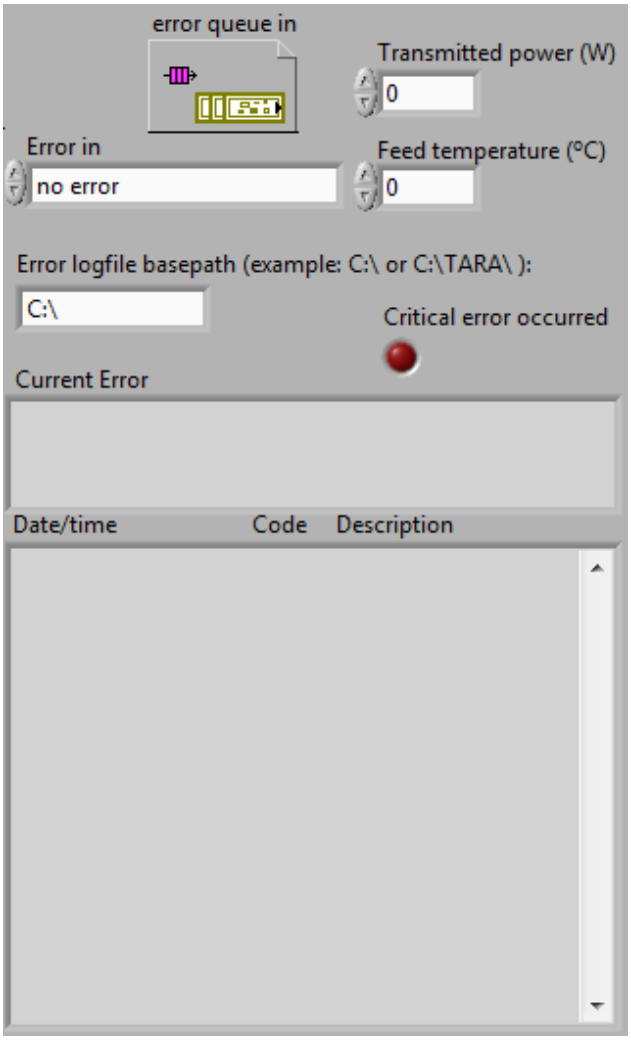


Figure B.4: System control panel

B.2 System control - Logger

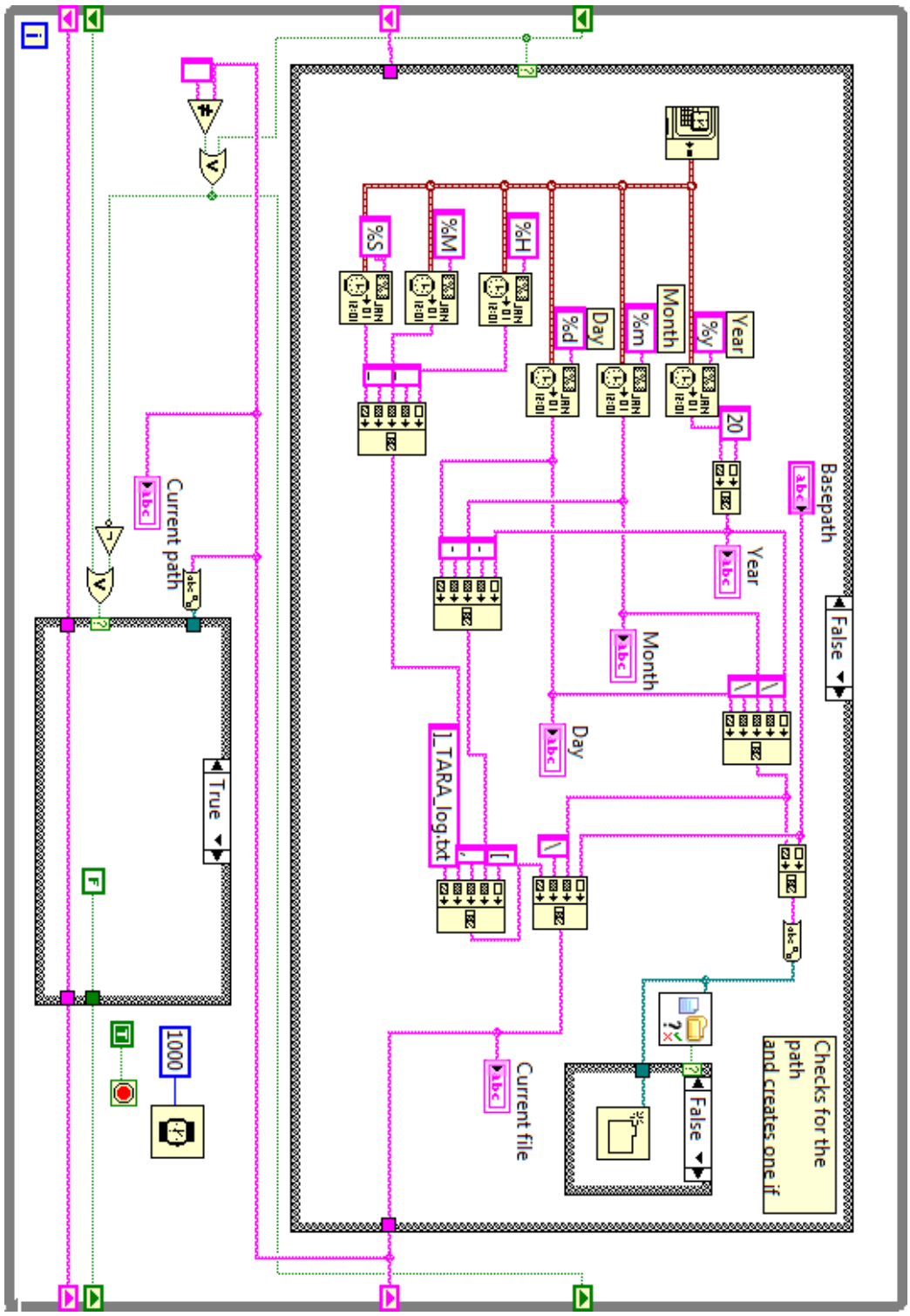


Figure B.5: System control - logger VI - "Directory Present" state

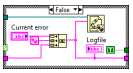


Figure B.6: System control - logger VI - “Create Directory” state

Figure B.7: Logger icon

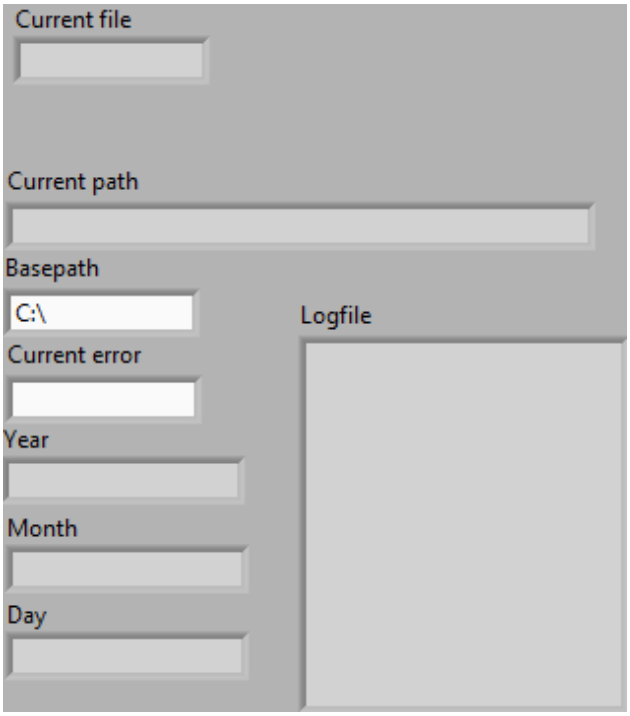


Figure B.8: Logger panel

B.3 Calc Matlab

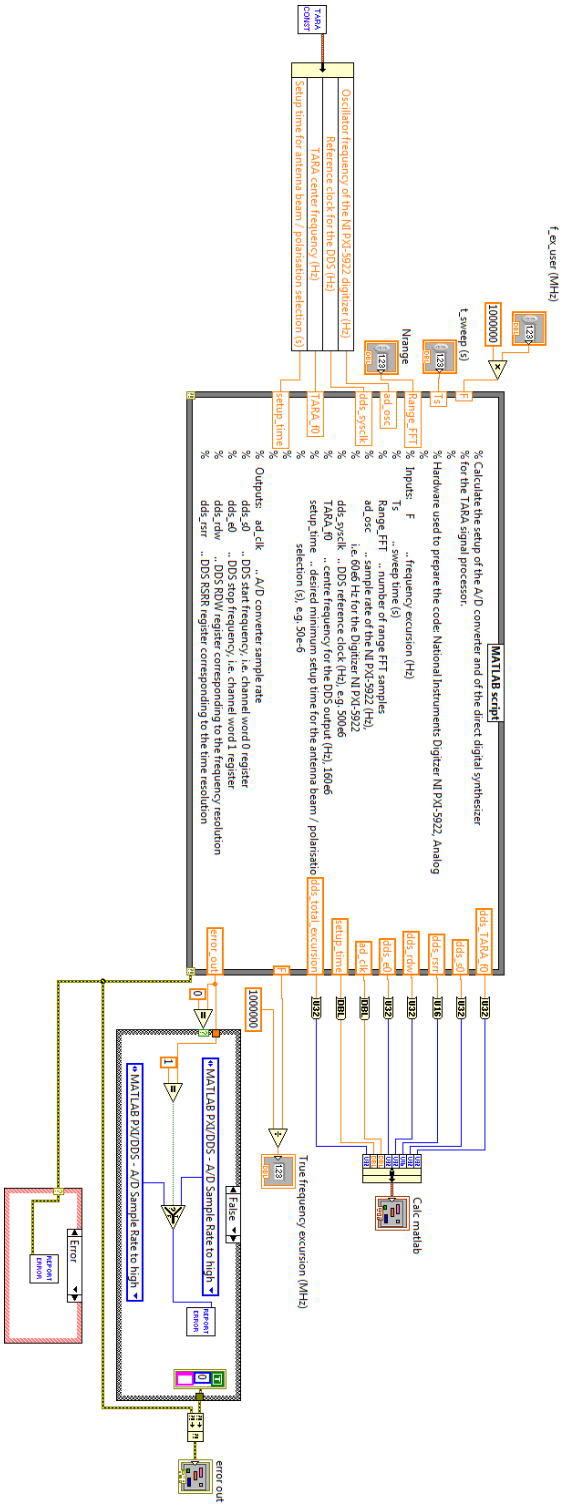


Figure B.9: Calc Matlab VI

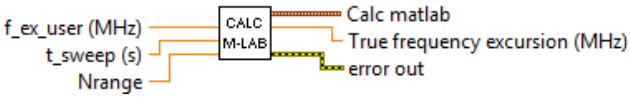


Figure B.10: Calc Matlab icon

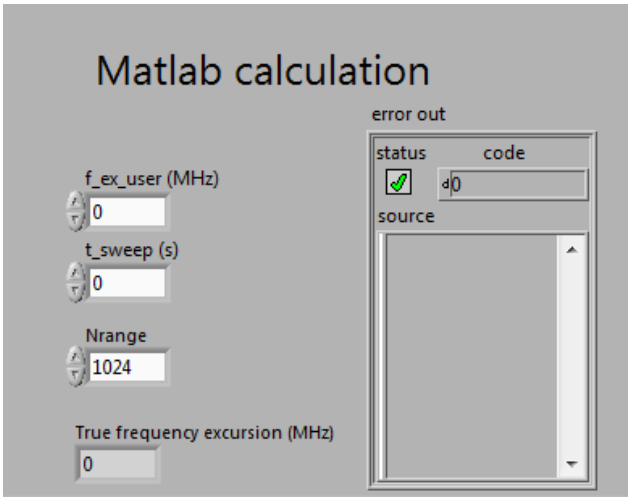


Figure B.11: Calc Matlab panel - the used Matlab can be found in Appendix Matlab

B.4 Header builder

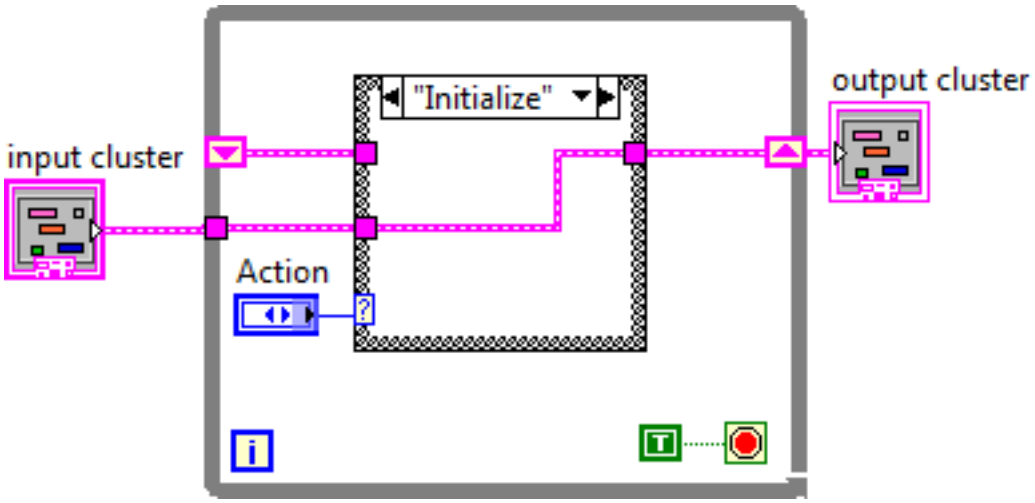


Figure B.12: Header Builder VI - “Initialize” state

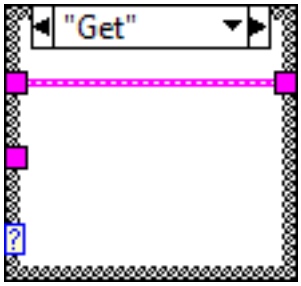


Figure B.13: Header Builder VI - “Get” state

Figure B.14: Header Builder icon

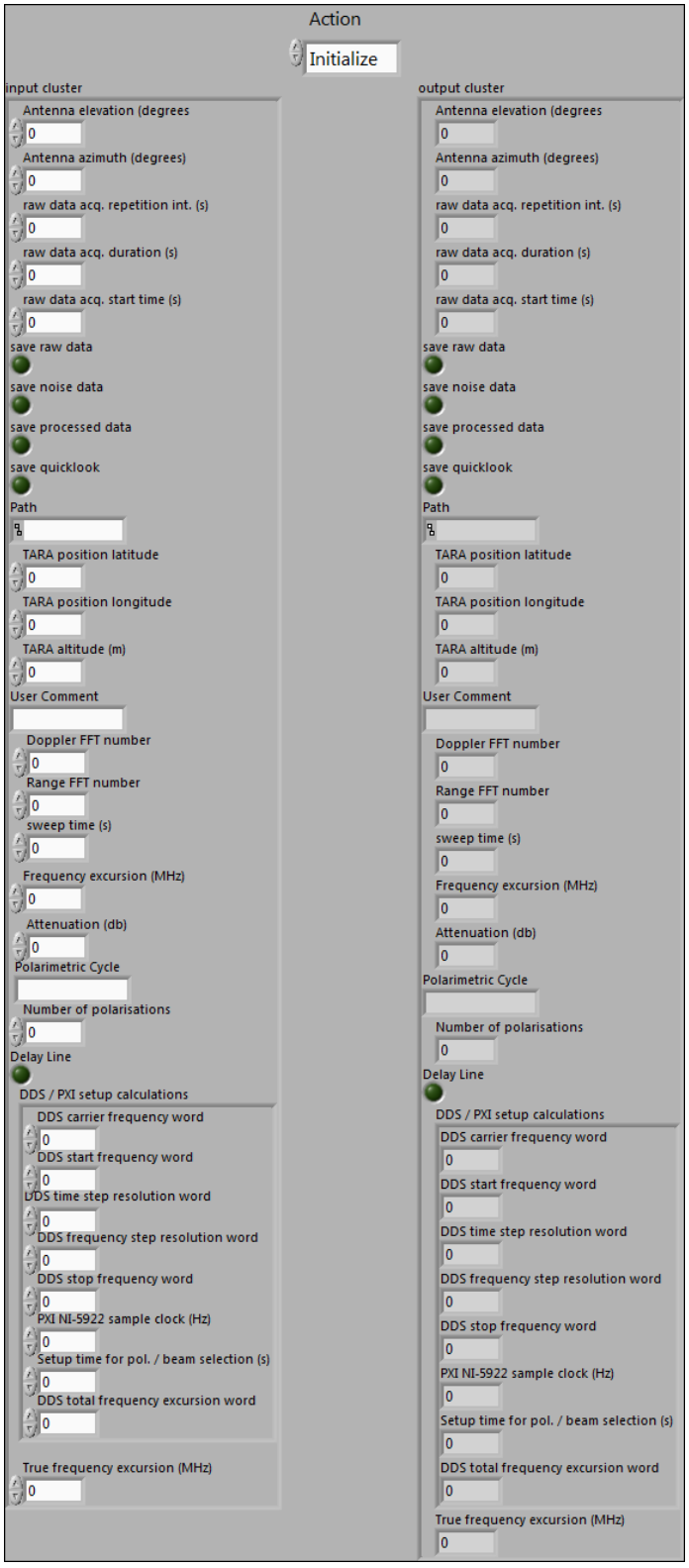


Figure B.15: Header Builder panel

B.5 Measurement

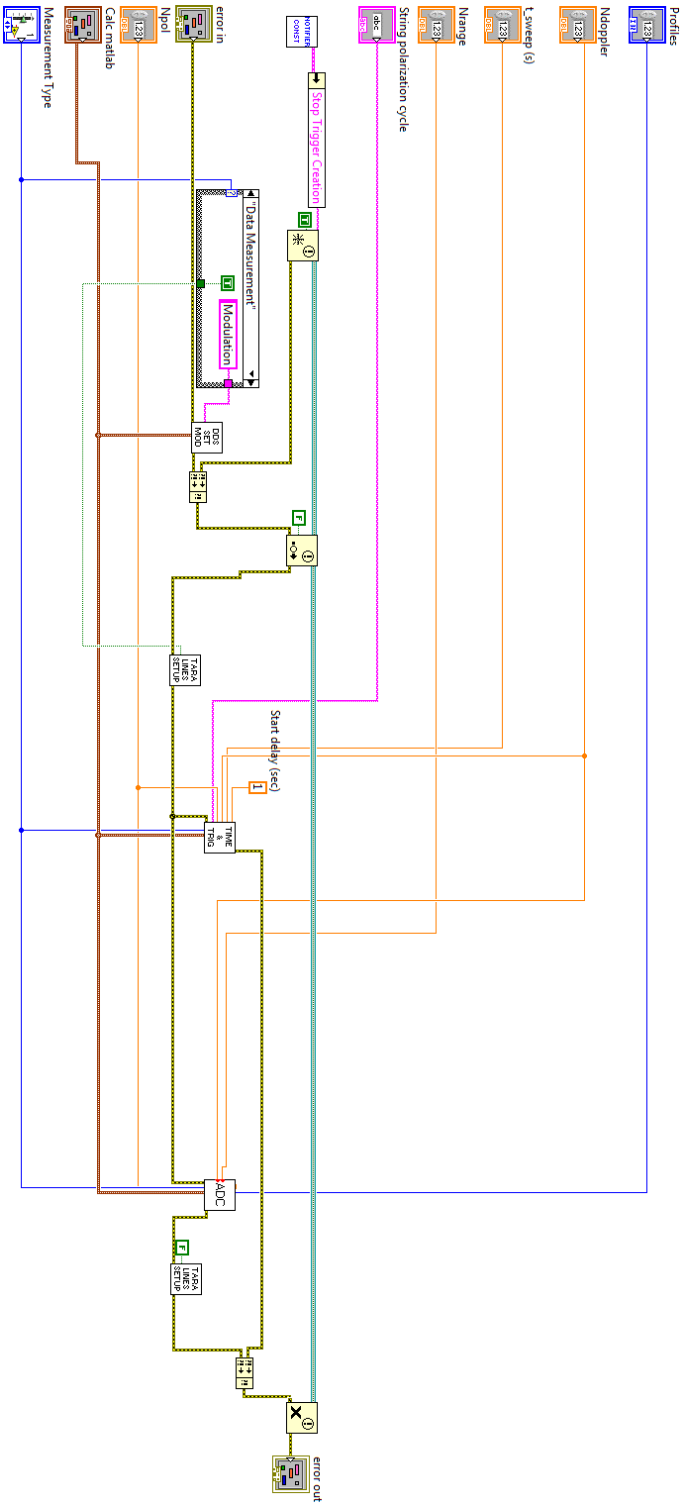


Figure B.16: Measurement VI - “Data Measurement” state

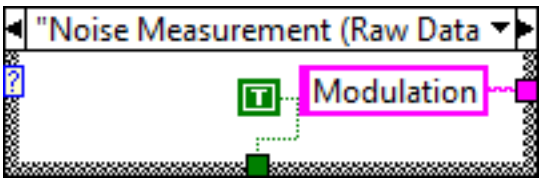


Figure B.17: Measurement VI - “Noise Measurement + Raw Data” state

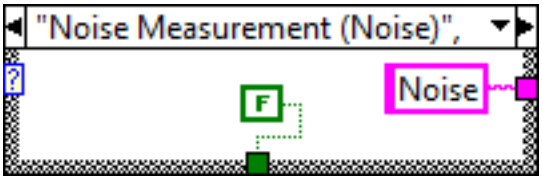


Figure B.18: Measurement VI - “Noise Measurement” state

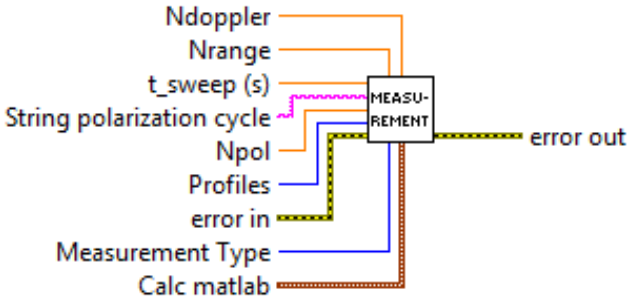


Figure B.19: Measurement icon

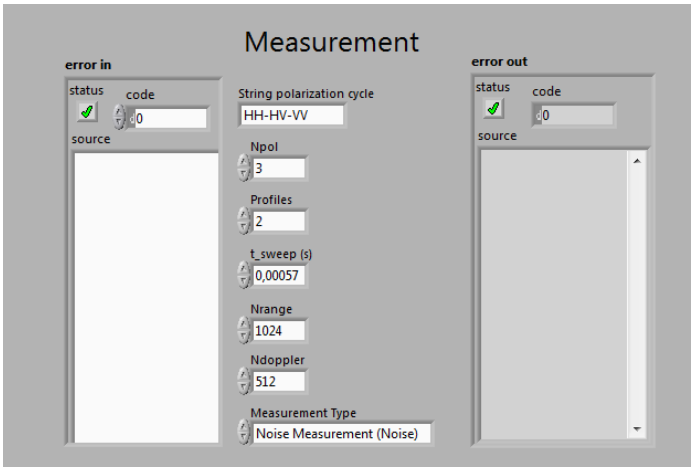


Figure B.20: Measurement panel

B.6 DDS Modulation

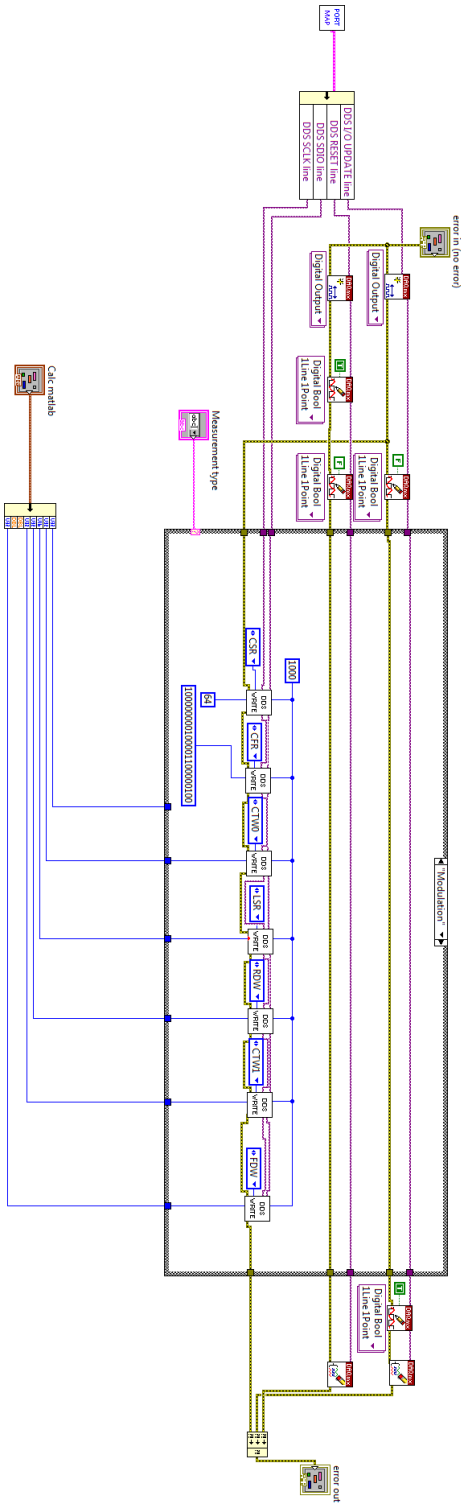


Figure B.21: DDS Modulation VI - “Modulation” state

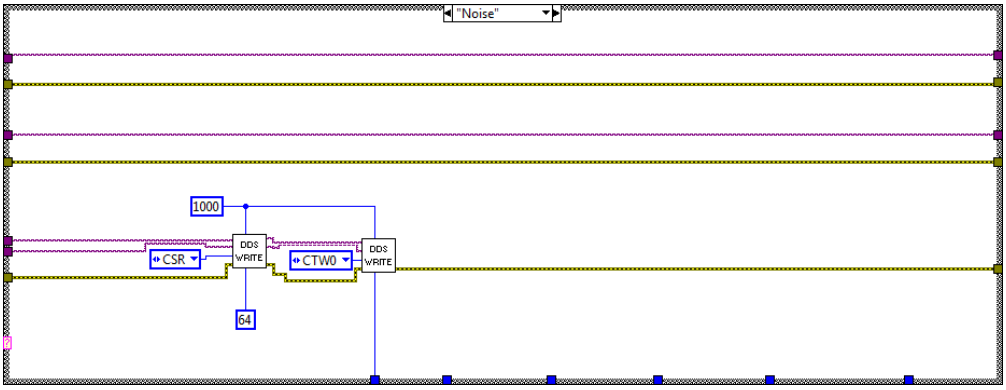


Figure B.22: DDS Modulation VI - “Noise” state

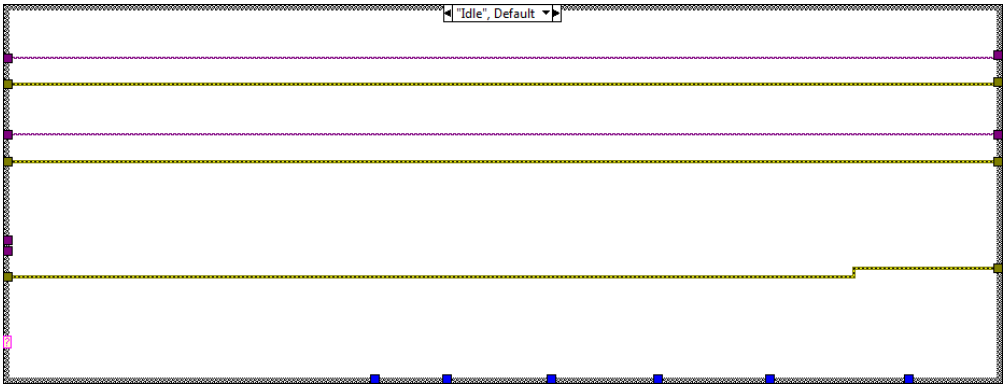


Figure B.23: DDS Modulation VI - “Idle” state

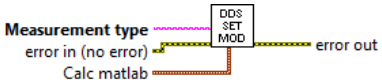


Figure B.24: DDS Modulation icon

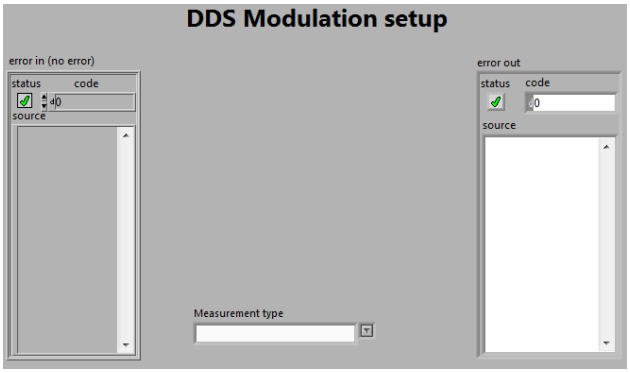


Figure B.25: DDS Modulation panel

B.7 DDS Write byte

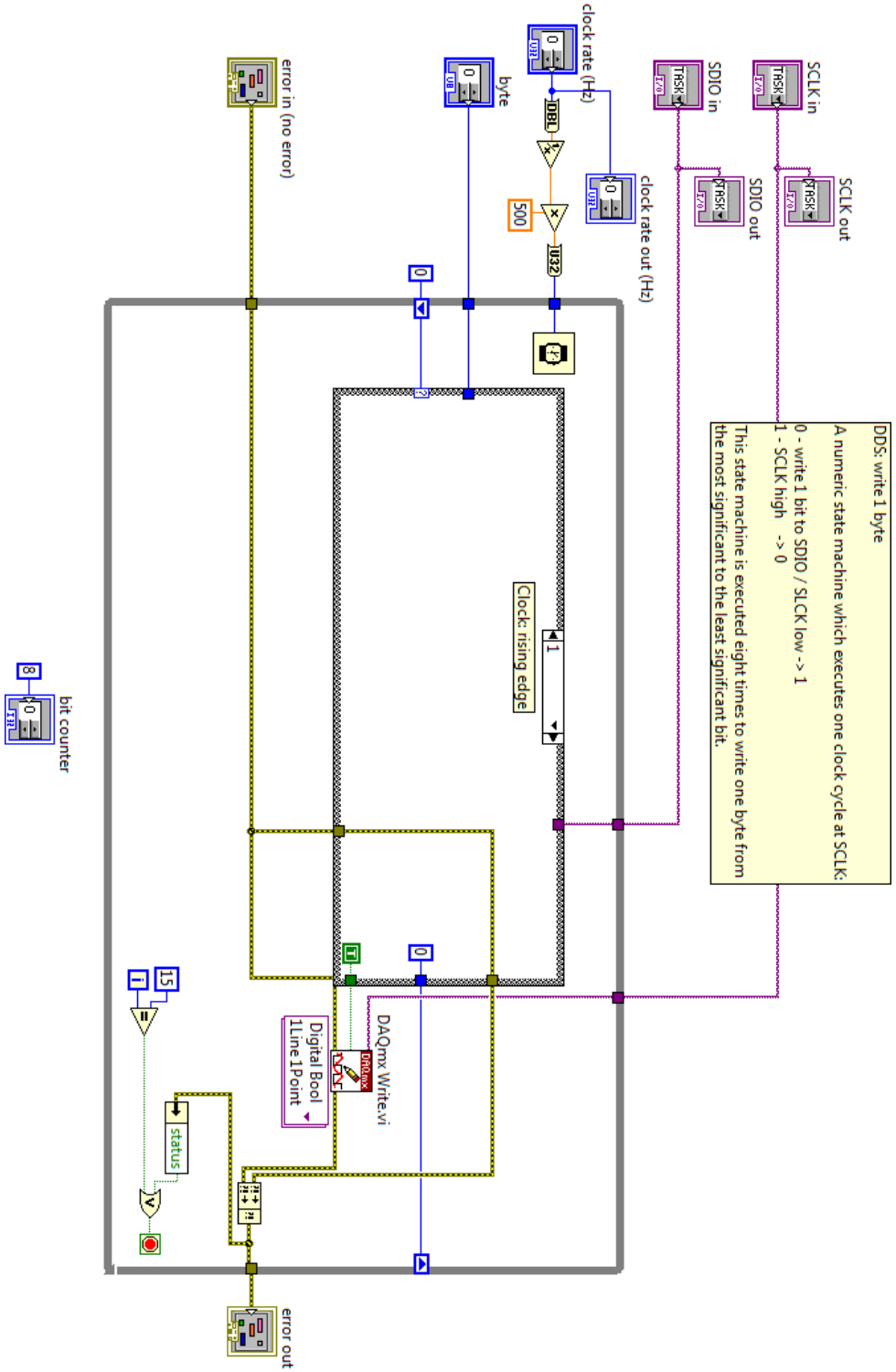


Figure B.26: DDS Write byte VI - “Clock rising edge” state

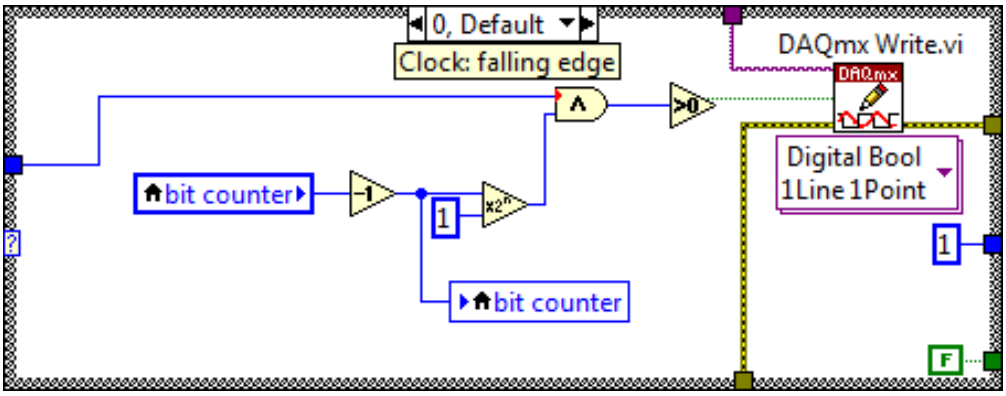


Figure B.27: DDS Write byte VI - “Clock falling edge” state

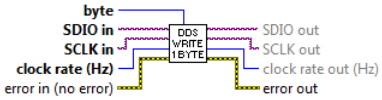


Figure B.28: DDS Write byte icon

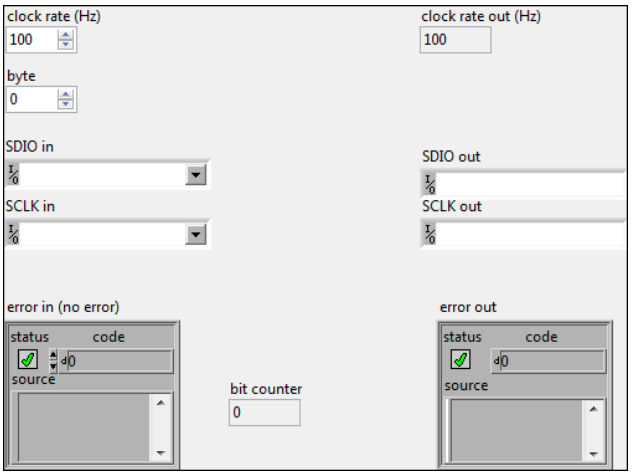


Figure B.29: DDS Write byte panel

B.8 DDS Write register

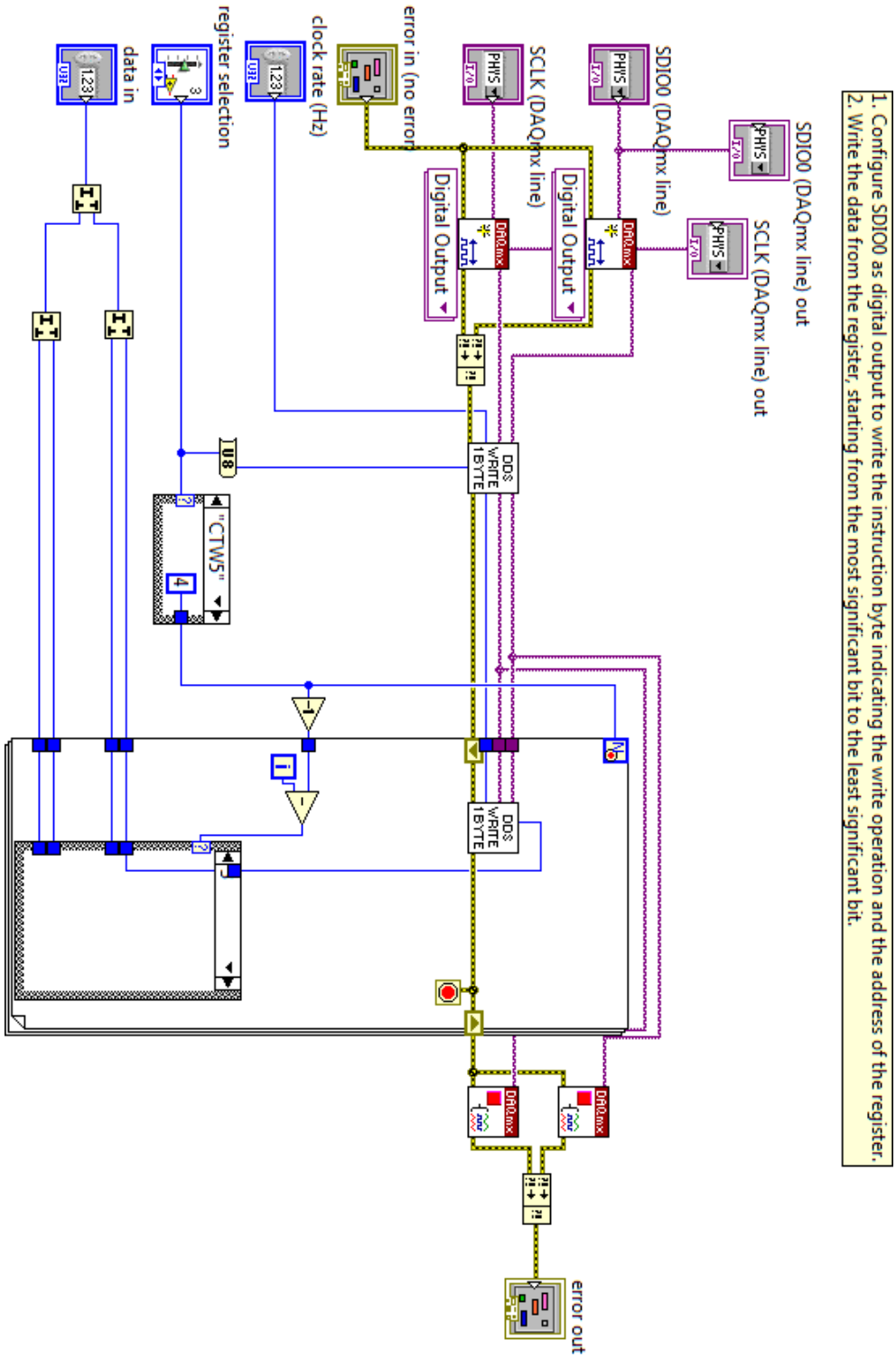


Figure B.30: DDS Write register VI

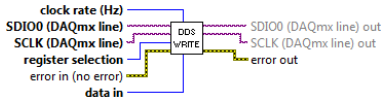


Figure B.31: DDS Write register icon

Figure B.32: DDS Write register panel

B.9 Timing module

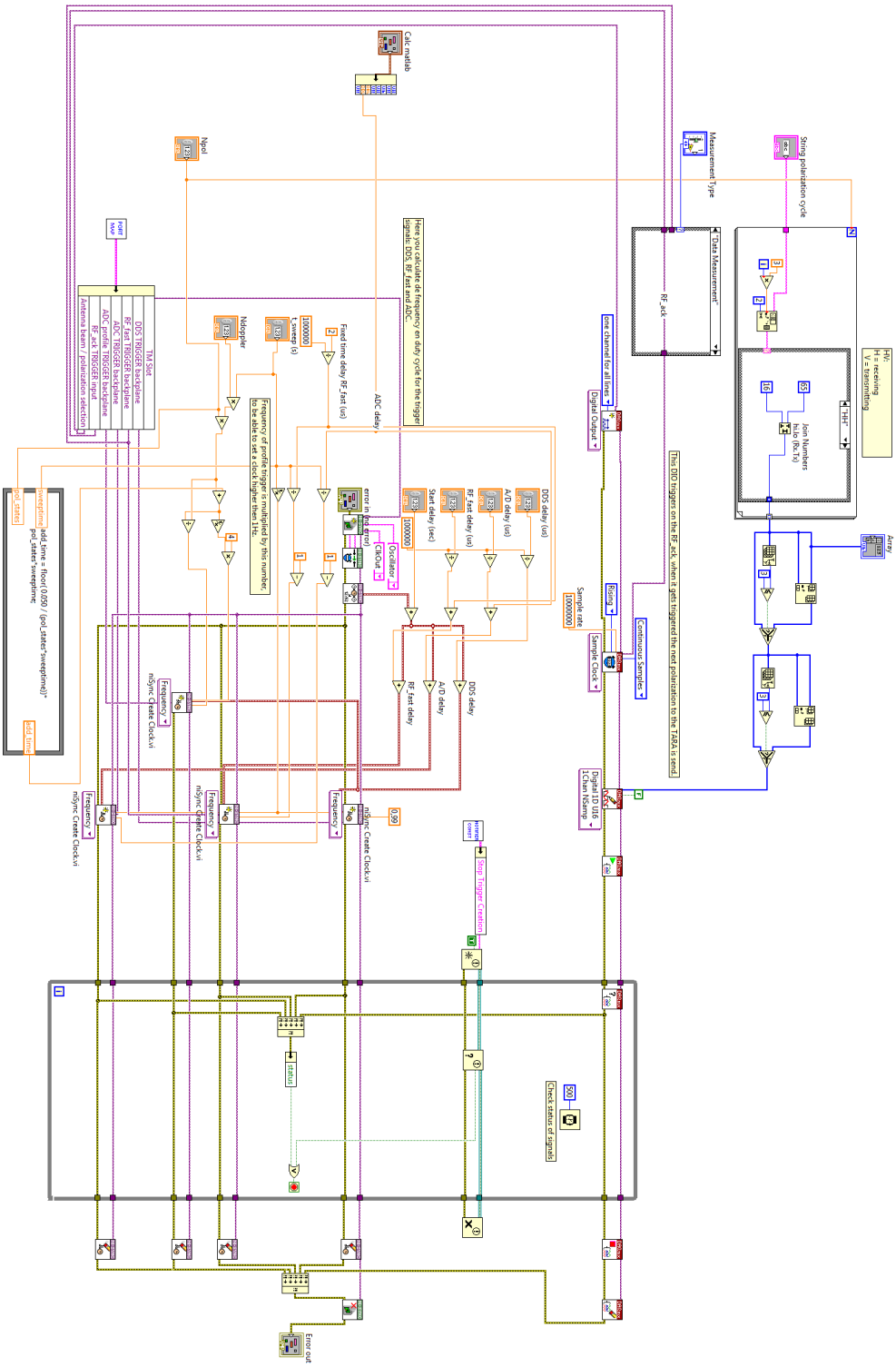


Figure B.33: Timing Module VI - “Data measurement” state

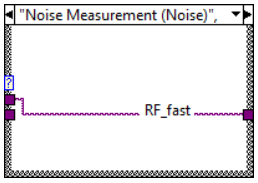


Figure B.34: Timing Module VI - “Noise measurement” state

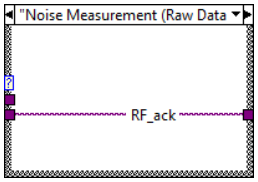


Figure B.35: Timing Module VI - “Noise Measurement + Raw Data” state

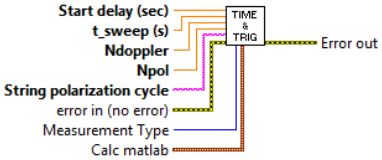


Figure B.36: Timing Module icon

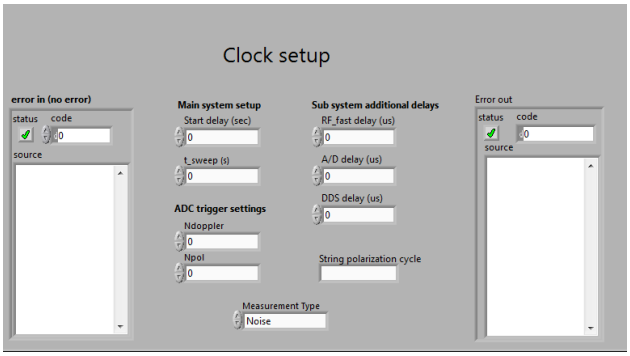


Figure B.37: Timing Module panel

B.10 Tara lines setup

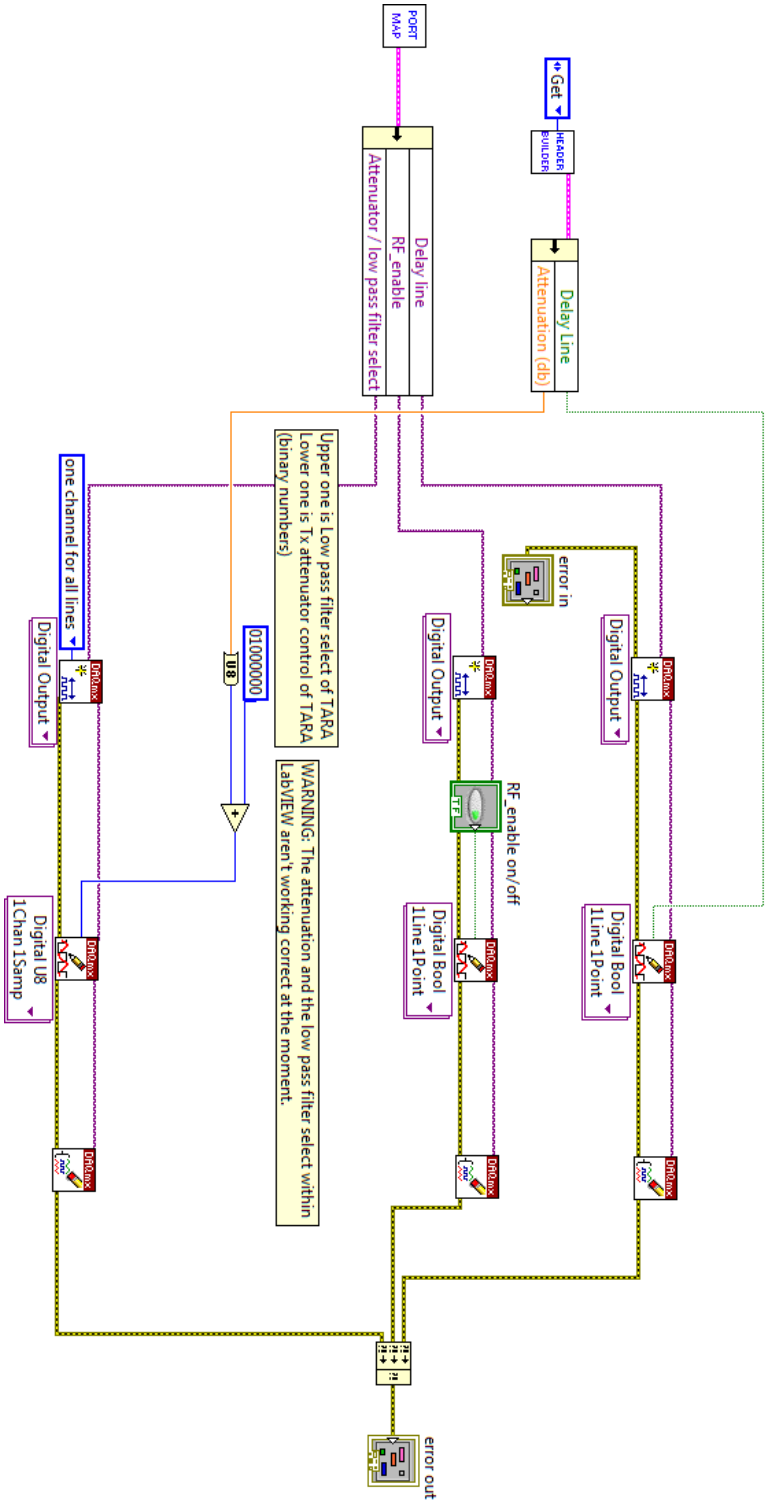


Figure B.38: Tara lines setup VI

Figure B.39: Tara lines setup icon

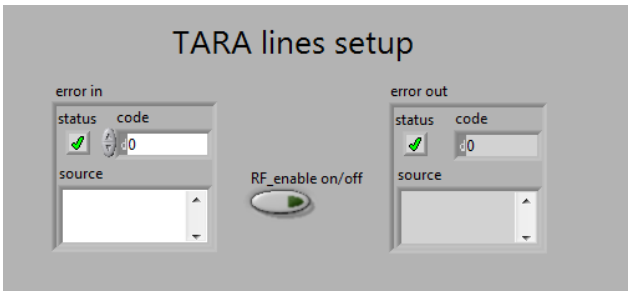


Figure B.40: Tara lines setup panel

B.11 ADC

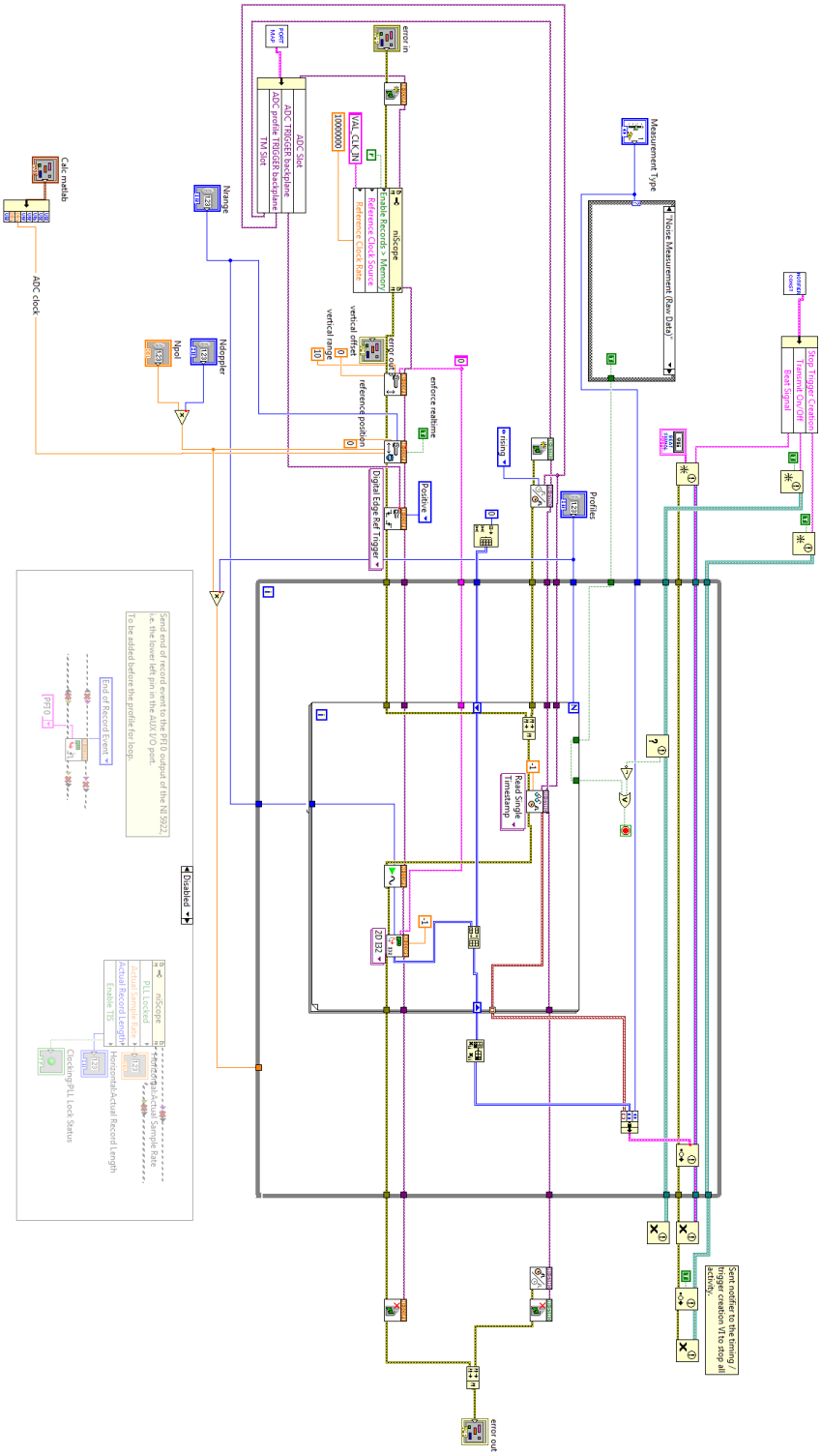


Figure B.41: ADC VI

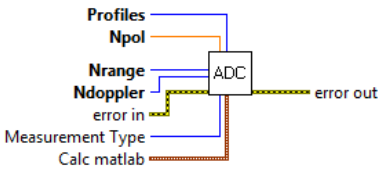


Figure B.42: ADC icon

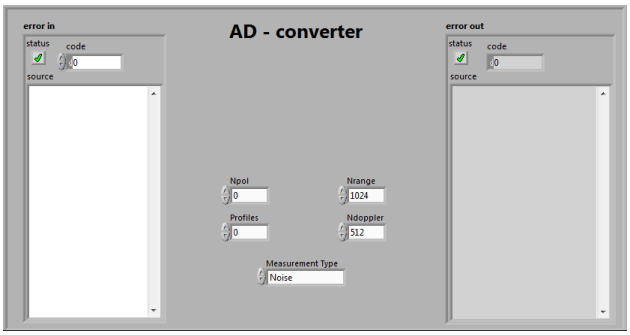


Figure B.43: ADC panel

B.12 Portmap

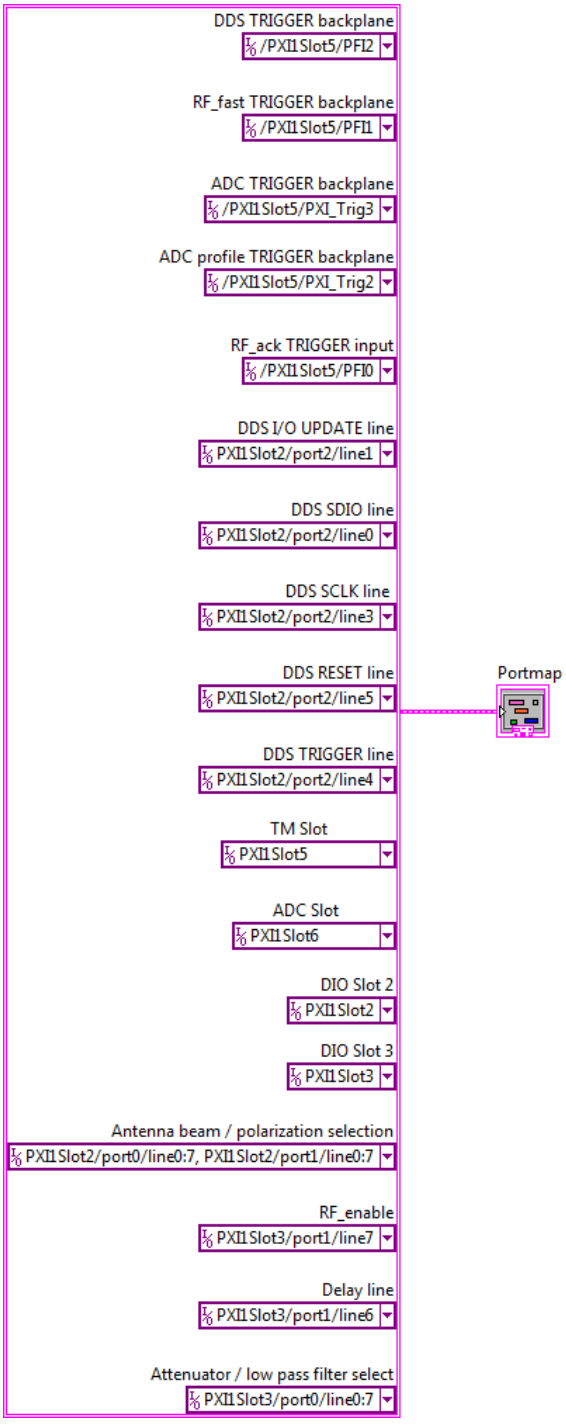


Figure B.44: Portmap VI

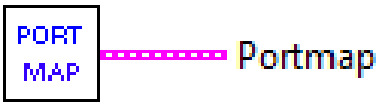


Figure B.45: Portmap icon

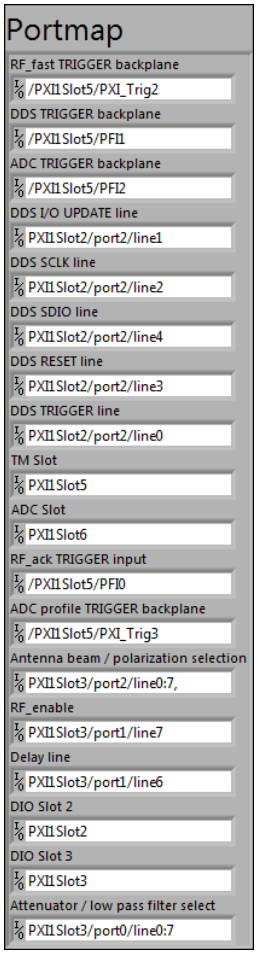


Figure B.46: Portmap panel

Appendix C

General Block Diagram

In order to obtain an overview of all the sub systems the radar interface needs to work correctly, a block diagram was created for the entire system. It is shown in figure C.1. This diagram shows the signals that each block within the system will send out to other blocks and which signals they need to receive.

It has gone through a number of iterations and with the requirements of the design becoming better specified, it developed into the diagram that can be found in Appendix A. It consists of the following seven blocks:

GUI

The Graphical User Interface is made to allow the user to define the desired type of measurement via an intuitive and easy to use interface (the front panel of the user interface is displayed in Appendix D). The GUI contains:

- A switch to start or stop the measurement
- Input boxes for all the variables the user needs to set for the measurement
- Tabs to switch between normal and advanced measurement options
- Check boxes to choose which data the user wishes to save
- Input boxes to set a measurement schedule
- A text box to input user comments
- 15 graphs displaying the processed data parameters as a function of height and time
- 5 tabs grouping the 15 graphs by parameters for the main beam, the 2 offset beams, the polarimetric parameters and the wind parameters
- A text box displaying the system status/errors
- The GPS location and time

All of the measurement settings are sent to the Header Builder and the signal from the on/off switch and the measurement schedule are sent to the Radar Control block.

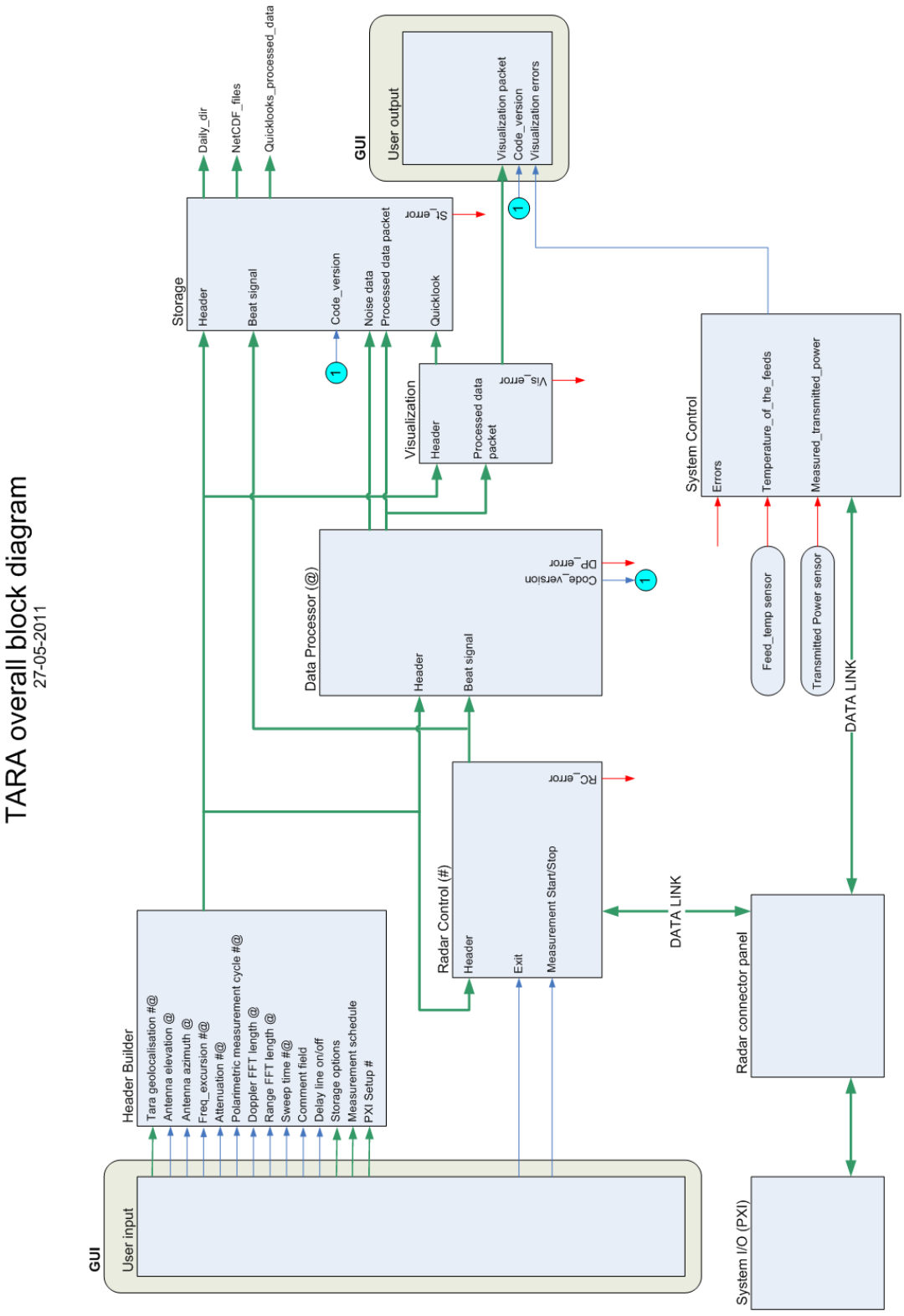


Figure C.1: Overall block diagram

Header Builder

The Header Builder takes all of the measurement settings that are set in the GUI and bundles them together so they can be easily transmitted. All variables present in the Header Builder block are fixed and set only once before the measurement. Therefore, they can easily be transferred to the other blocks.

The Radar Control needs the measurement settings to tell the PXI and consequentially the radar what type of measurement must be done. The Data Processor needs them in order to process the beat signal correctly, the visualization needs to know what exactly to display (the processed 3 or 5 cycle measurement data) and the Storage block needs to store the measurement settings together with the data that is to be saved.

Radar Control

The Radar Control block is designed to control and receive signals from the radar itself and start each of the other blocks when they are needed. Also, through the connection with the radar connector panel, it is able to send signals to the PXI which then sends the signals to the radar. By doing this, the Radar Control block can instruct the PXI to create the type of beat signal that is required for the noise measurement or the 3, 5 or customized cycle measurement.

Data Processor

The task of the Data Processor is to process the beat signal in order to extract atmospheric echoes that are received by the radar, so they can be stored and displayed. The processing is only performed in case of specific 3 cycle and 5 cycle measurement configurations. If the beat signal is of a custom measurement configuration, it will not be processed and only raw data will be stored. The detailed block diagram of the Data Processor can be found in Appendix A.

At the start of every measurement, a beat signal containing only noise is sent to the Data Processor. This noise signal will be processed and deliver 2 sets of variables of which the first will be used to filter out the hardware noise in the upcoming measurements and the second one supplies a calibration constant. These 2 variables will be sent to the Storage block as Noise Data.

After the noise data has been processed, the Data Processor will receive a beat signal for either a 3 or a 5 cycle measurement and then needs to process them with either the "Pola_rt block" for 3 cycle measurements or the "Polawind_rt block" for 5 cycle measurements. The processed data that is created will then be sent to the output so it can be sent to Storage to save the data and to Visualization to display it in graphs.

The sub-blocks of the Data Processor will also send a string with the code version along with the data. This allows the user to easily check which Matlab code was used the processing of the data that is being viewed.

Visualization

The Visualization block has to create packets of visualized data which the GUI can interpret and display in graphs. It also needs to create the quicklooks that need to be saved. The detailed block diagram of the Visualization can be found in Appendix A.

Up to 15 processed radar spectral polarimetric parameters coming out of Data Processor block are sent out as a visualization packet to the GUI. Each of the processed parameters is range and time dependent (represented as 2D matrices). They are therefore visualized on a 2D graph (time and range) in the GUI.

Storage

The Storage block needs to store the data so it can easily be found back and extracted. The detailed block diagram of the Storage can be found in Appendix A. The noise data, processed data, beat signal and the quicklooks can all be sent to this block along with the header information depending on what the user desires.

For the sake of clarity and in order to easily trace back the measurement specification, the header data are always combined together with the different data stored.

There should be 4 types of data packages that can be stored:

- A noise package, which consists of the header, processed noise data from a measurement of 2 minutes and 1 minute of raw data.
- A transmission package, which consists of the header and the processed data.
- A raw data package, which consists of the header and raw data.
- A quicklook package

To be able to different packages at the same time, several buffers are considered to be used in parallel after the merging of the required data has been done. When the buffers are full, the data should be written in NetCDF format. Finally, the data should be stored into several package dedicated folders present on the hard disk. These folder should also be placed within daily classified directories. This folder classification tree will be handled and created by the Storage block itself.

The Storage block also has its own control block. This block will be notified if the day is at an end, so that new file for a new day of measurements can be created. It will also receive a signal which indicates what packages the user wants to save as well as a signal that tells the Storage block that the measurement has been stopped and it needs to save the data.

System Control

The System Control block is there to receive and process errors. When one of the sub-blocks of the system runs into problems it will output an error to the System Control block. This block will then output the error to the user through the GUI and in the worst case it will send a signal to the Radar Connector Panel to shut down the radar.

Appendix D

Measurement

In order to prove that the final system is working, there had to be checked whether the trigger and timing signals were working in the proper way. In order to this the triggers were connected to the oscillator from which figures can be taken, or additionally data files which can be plotted in Matlab later on.

Figure D.1 shows the DDS sweep and the DDS trigger signal plotted in Matlab. The red line represents the sweep, in order to get this graph plotted together with the trigger (black line) the y-axis of the sweep was chosen in such a way that it fits in the figure. The y-axis shown corresponds to only the black line. Another thing that has changed to get such a figure are the settings of the sweep. Normally the frequency excursion equals 5 MHz and the center frequency equals 160 MHz, however, the used oscillator was not able to show the sweep correctly because of this high frequency. That's the reason a frequency excursion has been chosen equal to 0.5 MHz and a center frequency of 1 MHz.

What can be seen in the figure is that when the trigger signal has a flank up, the signal is starting to sweep from the lower frequency. Just before the trigger signal becomes low the frequency is at its maximum frequency, after the flank downwards the frequency falls rapidly back to the starting frequency waiting for the next trigger. This works in the correct way.

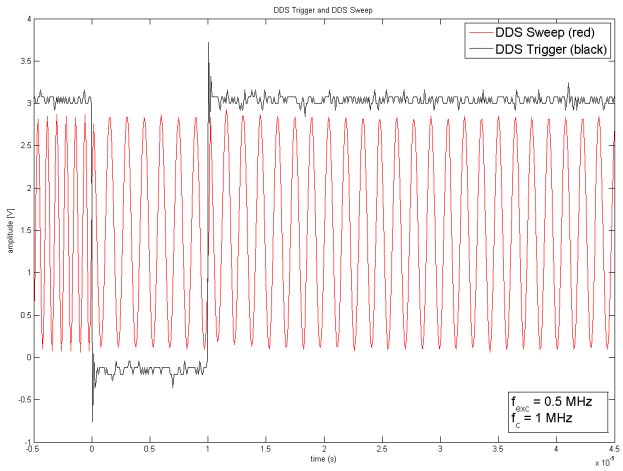


Figure D.1: DDS trigger sweep

At the TARA site measurements have been taken with the oscillator as well. These are depicted in figure D.2, where also a zoomed version is displayed. The green line represents the RF_ack signal coming from TARA and the black line the RF_fast trigger generated by the Timing Module. What can be seen is that

when the black trigger line becomes low (and the transmission switchs off) the green line jumps to high after a small delay. This is correct because when the transmission is off the RF_ack should be high as indication that there is no power on the lines anymore.

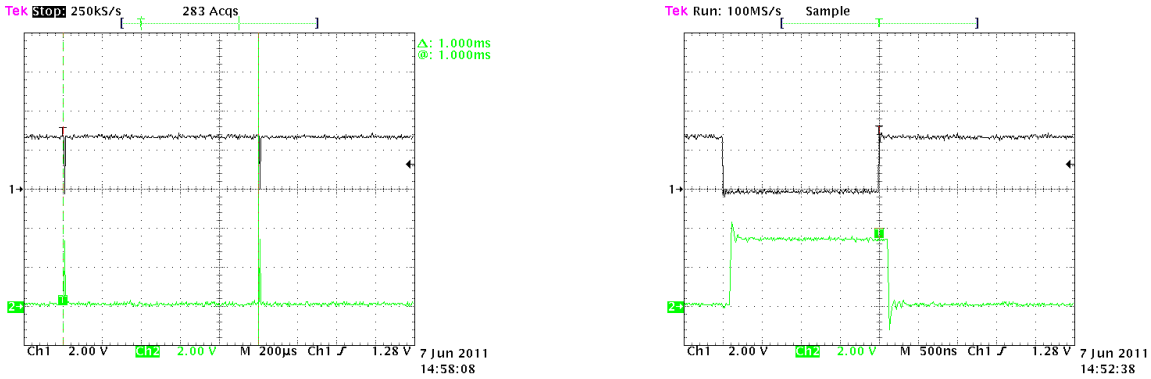


Figure D.2: RF_fast trigger & RF_fast trigger with zoom

Figure D.3 shows two triggers, the top one is the ADC trigger and the middle one is the RF_fast. The lowest one is both signals overlapped. What can be seen is that both start low at the same time (in this case at t = 0 and one sweep time later, t = 1 ms), after about 2 μs RF_fast gets high which means transmission starts. After about 120 μs the ADC starts triggering because this signal gets high as well. This 120 μs is the setup time which is calculated using the Calc Matlab VI. This pattern of the signals is correct.

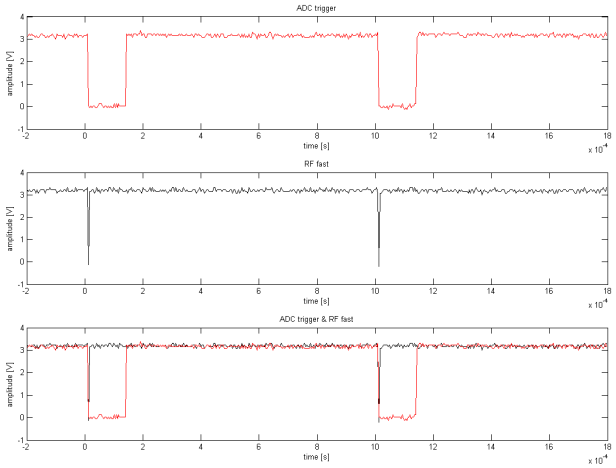


Figure D.3: RF_fast and ADC triggers

Appendix E

History of the TARA

In this chapter the history of the TARA system will be given. First a description of the original system will be provided as it came into use in April 2000. Then, changes performed in 2004 during a maintenace period on part of the hardware configuration, and the reasons behind them, will be presented. Finally this chapter will finish with an overview of the planned hardware setup for the upgrade, together with an explanation why this setup is different.

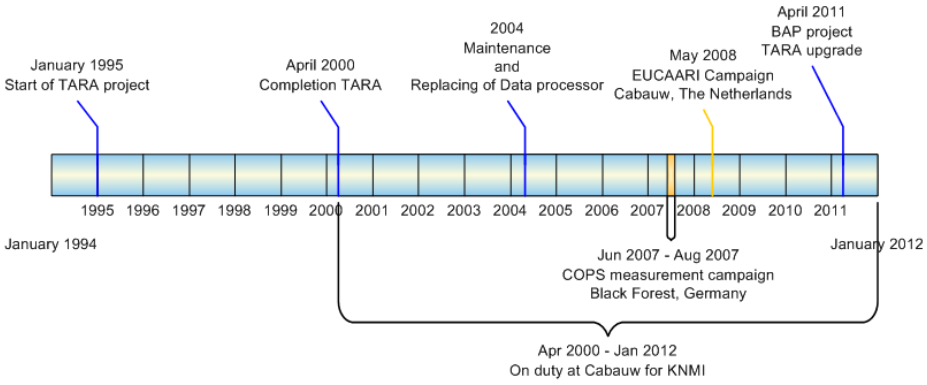


Figure E.1: TARA time line

E.1 Transportable Atmospheric Radar

The project for designing and building the TARA system started early 1995 at the International Research Centre for Telecommunications-transmission and Radar, IRCTR. The radar became operational in April 2000. These first five years project was financed by the Dutch Scientific Society (STW).[6] The final system was to be a transportable atmospheric profiler that could be used for the study of the static and dynamic behavior of the atmospheric phenomena at different geographical locations [21].

System Description
Strict requirements were considered for the construction of the TARA radar. As the acronym TARA implies, the system was designed to be transportable. For this reason the radar was entirely mounted in a modified 12 meter container so that it can be transported within international standards of road transport. Installing

the system into its measurement mode or dismantling the system for transport should both take less than one week. A great effort was also put into realizing the measurement diversity provided by this radar. The central frequency of the device was chosen at 3,3 GHz in order to detect reflections from ice cloud particles, precipitation regimes as well as from clear-air turbulence. Central beams of the transmitter and receiver antenna are multi-polarized enabling the measurement of the full scattering matrix. The polarization of both antennas can be switched independent of each other on a sweep to sweep base such that co and cross-polar measurements can be done.

A triple feed system was also implemented to enable electronic beam switching over an angle of 15 degrees in two orthogonal planes. A Doppler measurement at three different angles combined with the assumption of sufficiently long correlation lengths for the horizontal wind velocity makes three dimensional wind speed determination possible. The antennas are based upon parabolic reflectors. They have a three meter diameter. This was a trade-off between high gain and transportability. The characteristic large shields around the antennas are used to reduce side lobes in the 90 degrees direction and are clearly visible.

For maximum flexibility the system is based on the FM-CW (Frequency Modulated Continuous Wave) principle, meaning that the carrier frequency is modulated from a start to stop frequency. Processing of the echo's recieved form this tranmitted signal provide range and doppler information obtained from two consecutive FFT's performed on the data. The fact that the the Radar uses FM-CW allows the RF-amplifiers to be solid-state because of the much lower peak-power transmitted. This makes increases ease of maintanence and reliabilty of the radar.

System parameters such as the sweep time, frequency excursion, transmitted power and signal polarization can be changed independently from each other and are controlled by the computer. For the measurements, the antennas can be directed into any elevation between the horizon, 0 degrees and vertical 90 degrees. Real-time processing was originally done on a dedicated DSP-based computer system. This processing included filtering, noise correction, clutter suppression and statistical quantification of the Doppler spectrum. Display of the reflectivity measurement and storage of the data are both done in real-time. Most of the information from this section was acquired from [21],[9] and [6].

Table 1
Specifications of the TARA system

Parameter	Value	Additional remarks
Type	FM-CW	
Central frequency	3.3 GHz	
Transmitted power	100 W	can be attenuated in 10 dB steps
Dynamic range	80 – 90 dB	
Noise figure	~1 dB	
Sweep bandwidth	2 – 50 MHz	computer controlled
Sweep shape	triangular, saw tooth	can be arbitrarily
Sweep time	> 1 ms	can be staggered
Sampling	< 2 MHz	16-bits ADC
Samples per sweep*)	1024	
Sweeps per Doppler spectrum*)	512	
Polarisation (linear)	HH, HV, VH, VV	antennas controlled independently; central beam only; offset beams are single linearly polarised
Resolution aspects		
Max. range	38 km	@ 2 MHz bandwidth
Min. range resolution	3 m	@ 50 MHz bandwidth
Unambiguous Doppler speed	~22.7 m/s	max. @ 1 ms sweep time
Doppler resolution	~8.9 cm/s	@ 1 ms sweep time
Sensitivity[*)]	@ 5 km	@ 1 km
Reflectivity	$\leq 2.3 \cdot 10^{-14}$ m^{-1}	$\leq 0.9 \cdot 10^{-15}$ m^{-1}
Radar cross-section	$\leq 1.8 \cdot 10^{-8}$ m^2	$\leq 2.8 \cdot 10^{-11}$ m^2
*) SNR = 0 dB, resolution = 40 m		

Figure E.2: TARA system specifications

E.1.1 Initial Control & processing Units

When TARA was originally designed, the hardware that would control the radar and do the processing was a dedicated computer system [10]. This system was custom designed and made by the company Prodrive, located in Eindhoven, the Netherlands [15].

A picture of the original system is given below. From top to bottom, the first rack consists of a stable power supply for the dedicated computer below it. The computer will be discussed in the next section. The following two racks are filled with the RF/IF and LF/IF circuitry supplemented with status output and delay-line. Next in line is a cable to compensate for the difference in length of the cables between the transmitter and receiver. The last one consists of another power supply which is used to power the three racks above and the transmitter and receiver feeds on the outside of the TARA.

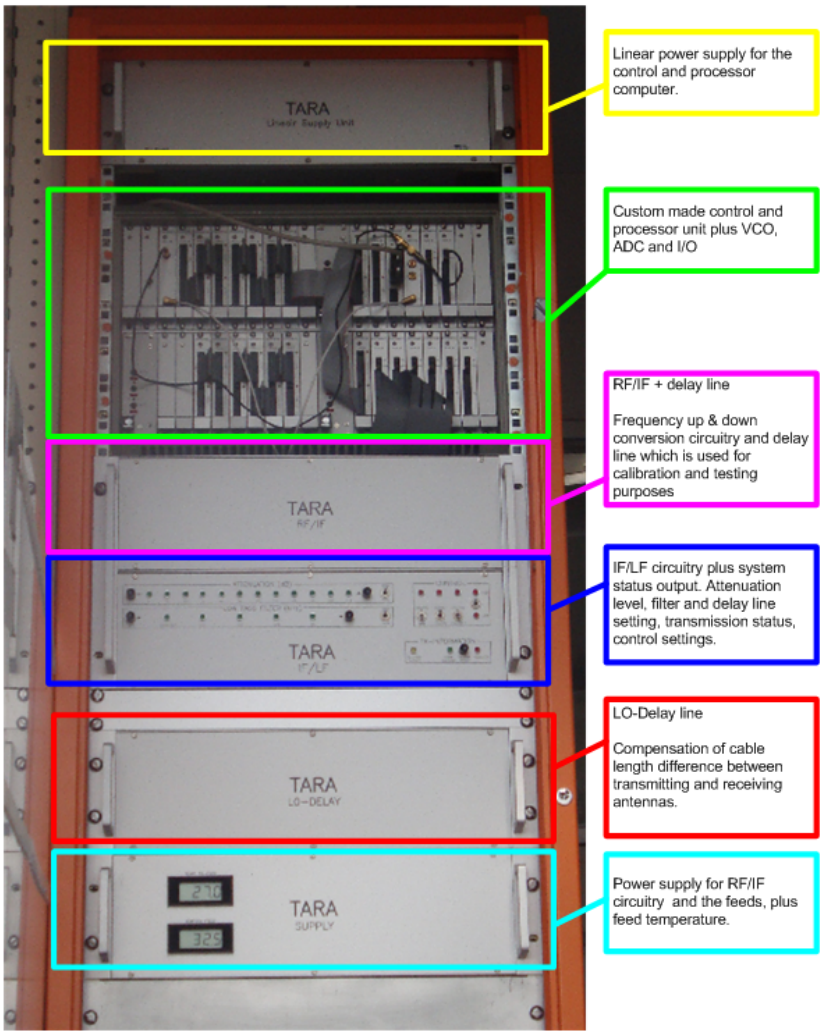


Figure E.3: Prodrive stack

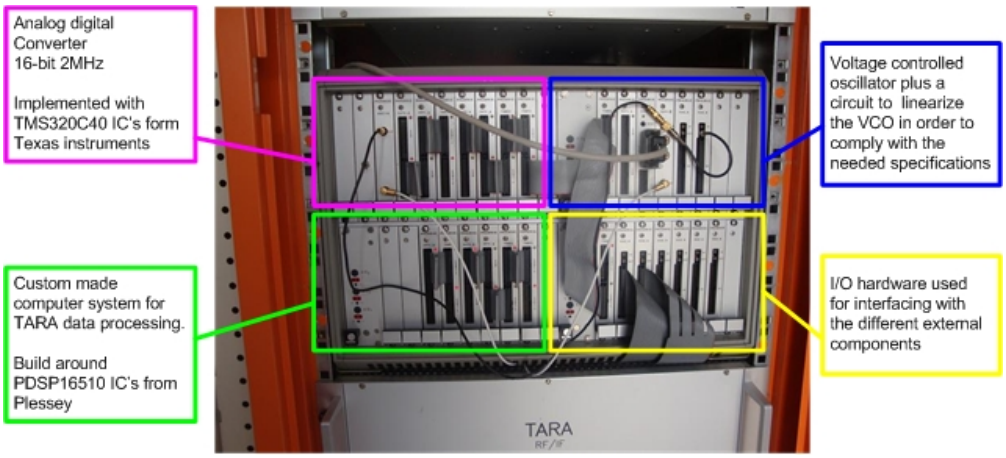


Figure E.4: Prodrive frontpanel

The Dedicated Computer, VCO, ADC and I/O The dedicated computer system was designed to perform both the required processing and the radar-system control. Control tasks of the radar hardware include, linearization of frequency sweep, polarization and beam switching, bandwidth and frequency agility, timing and synchronization issues. Polarization and beam switching, bandwidth and frequency agility can be changed by the user on a sweep to sweep base.

Real-time processing was done on a dedicated DSP-based computer system and included filtering, noise correction, clutter suppression and statistical quantification of the Doppler spectrum. Data was digitized with a 16-bit 2 MHz ADC and stored into a local memory before being handled by the processing in the computer mentioned above. The input output interface (I/O) supplies the needed connectivity to subsystems outside the Prodrive cabinet. Such as the Transmitter, Receiver, Storage and Monitor.

In FM-CW radars, non linearity of the frequency sweep limit the achievable resolution, because the range resolution is both determined by the bandwidth and the frequency sweep linearity.

In general, a VCO does not comply with the needed specifications and therefore the frequency curve has to be linearized. In Tara the linearization is done in a feed-forward way. This means that the voltage depending frequency is measured. This measurement is done in a stationary state. From the measured frequency characteristic, a correction table is calculated. Before starting a measurement, a control signal for the VCO is calculated from the table and is stored into a RAM-memory. During each sweep this memory is played back, generating a linear frequency sweep.

E.1.2 Current Hardware Configuration

In 2004 the performance of the Plessey ICs in the Data processing had deteriorated to such an extend that the data processing needed to be replaced. Besides, there was a need for some maintenance to keep the system running. Because at the time, FFT algorithms where commonly available in C and because of the processing power of common PCs, the need for custom made data processing was no longer present. So the choice was made to implement the data processing on a PC.

Because the computer for the data processing did not have a slot for a ISA card, another computer was needed for the user input and radar control. That is the reason why the TARA is currently controlled by two PCs. Such a non fully integrated system suffered from several drawbacks which decreased the possibilities for the real-time processing, the storage of the data and the remote controlled of the radar system.

In order to provide the connectivity from original ADC to the PC a new I/O device was needed. For this functionality a PCI6534 was ordered from National instruments. This change in configuration was more of a patch-up to keep the system operational until a proper upgrade could be realized.

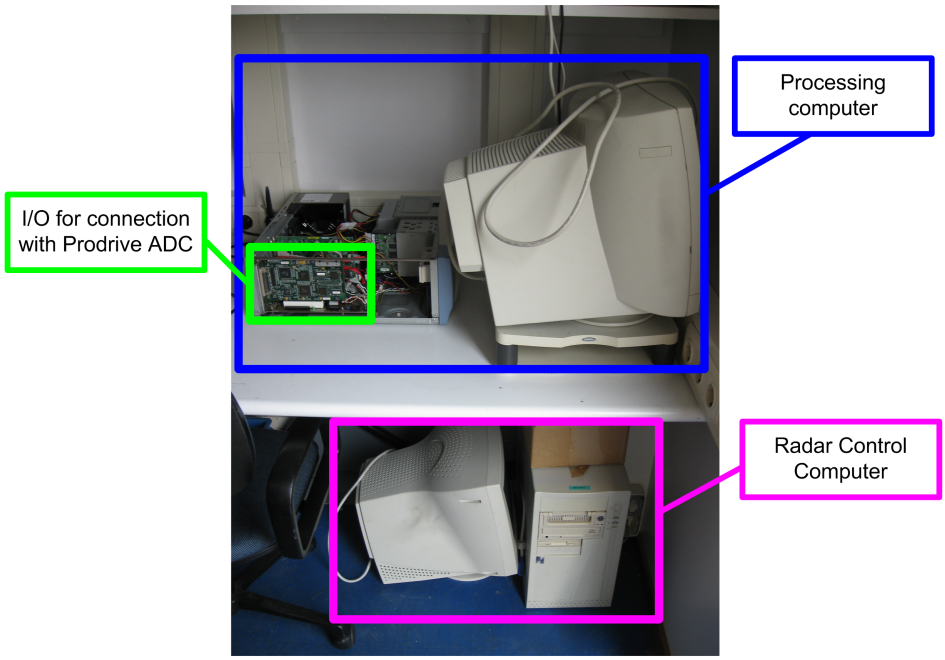


Figure E.5: Current Hardware Configuration

E.1.3 The Planned Setup

The planned setup should improve the overall system in a number of ways.

- Increased system performance
- Flexible software and hardware (use of Matlab within a LabView environment for example)
- Processing and control on one computer
- One integrated system
- Compact hardware setup
- Easy adaptability for future research

The used computer

Figure E.6: DELL Precision T1500 [17]

The computer made available for the TARA is a Dell Precision T1500. Figure E.6 shows the PC. Inside this computer there is an Intel core i7 870 CPU with four cores with hyperthreading, all four can run at a clock frequency of 2,93 GHz. Futher more this system has 8.00 GB of memory and it is driven by the Microsoft Windows 7 64 bit-edition. This computer will replace both the computers that are currently used for the control and processing.

The PXI-system The PXI-system is build by National instruments. This company specializes into developing products that simplify development and thus allowing for more rapid proto-typing. This is achieved by offering a graphical programming software and modular hardware that can be bought off the shelf and is plug and play to install.

If in the future there might be a need for further alterations on the radar, the PXI-system allows the users to modify or improve the hardware by simply removing, adding or changing the cards in the different slots. This makes the hardware very flexible for future alterations.

All hardware components are also supplied with software elements which can be easily used within the graphical programming software named LabVIEW. This software comes supplied with useable documentation and although it is a programming language that runs within its own enviroment that is run on a operating system, it is generally reliable. These features make it relatively easy to design software that adequately controls and configures the hardware components from National Instruments.

To summarize, the system provides the ability to rapidly construct a working proof-of-concept without many of the interface and start-up problems other solutions have. Also taking into account the good

experiences with the system by the RSE group in the past and the service provided by National Instruments, the PXI-system can be considered as the hardware of choice for this project.

Figure E.7: PXI

Our system consisted of a NI PXIe-1082 chassis with the following cards.

Direct Digital Synthesizer

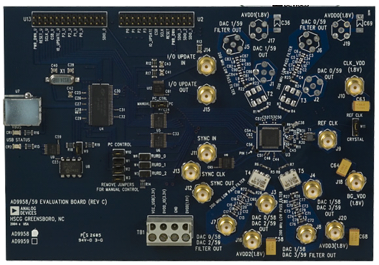


Figure E.8: DDS

After the upgrade, the signal generation will no longer be done by the VCO. This component will be replaced by a Direct Digital Synthesizer (DDS). Due to the digital nature of this device, many of the limitations of a VCO are overcome [11]. shows that a DDS provides increased performance in FM-CW radars. Further advantages listed are increased flexibility due to ease of programming, long term stability due to its digital nature of the device and with advent of digital communications, its reduced price. The DDS has already been implemented in the FM-CW radar, the IDRA radar with satisfactory performance increase and is now the standard way signals are generated in IRCTR.

Oscillator

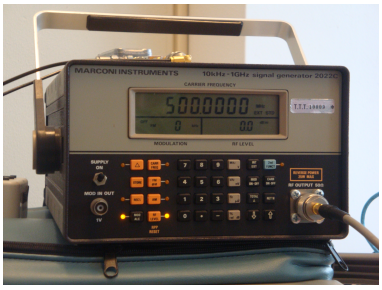


Figure E.9: Oscillator

The DDS is controlled by a stable 500 MHz reference clock generated by the oscillator, this clock signal will replace the internal clock of the device. This oscillator can be phase locked to another external clock provided by the PXI-system, which enables the DDS to be in synchronization with the hardware of the PXI-system.

Appendix F

TARA I/O

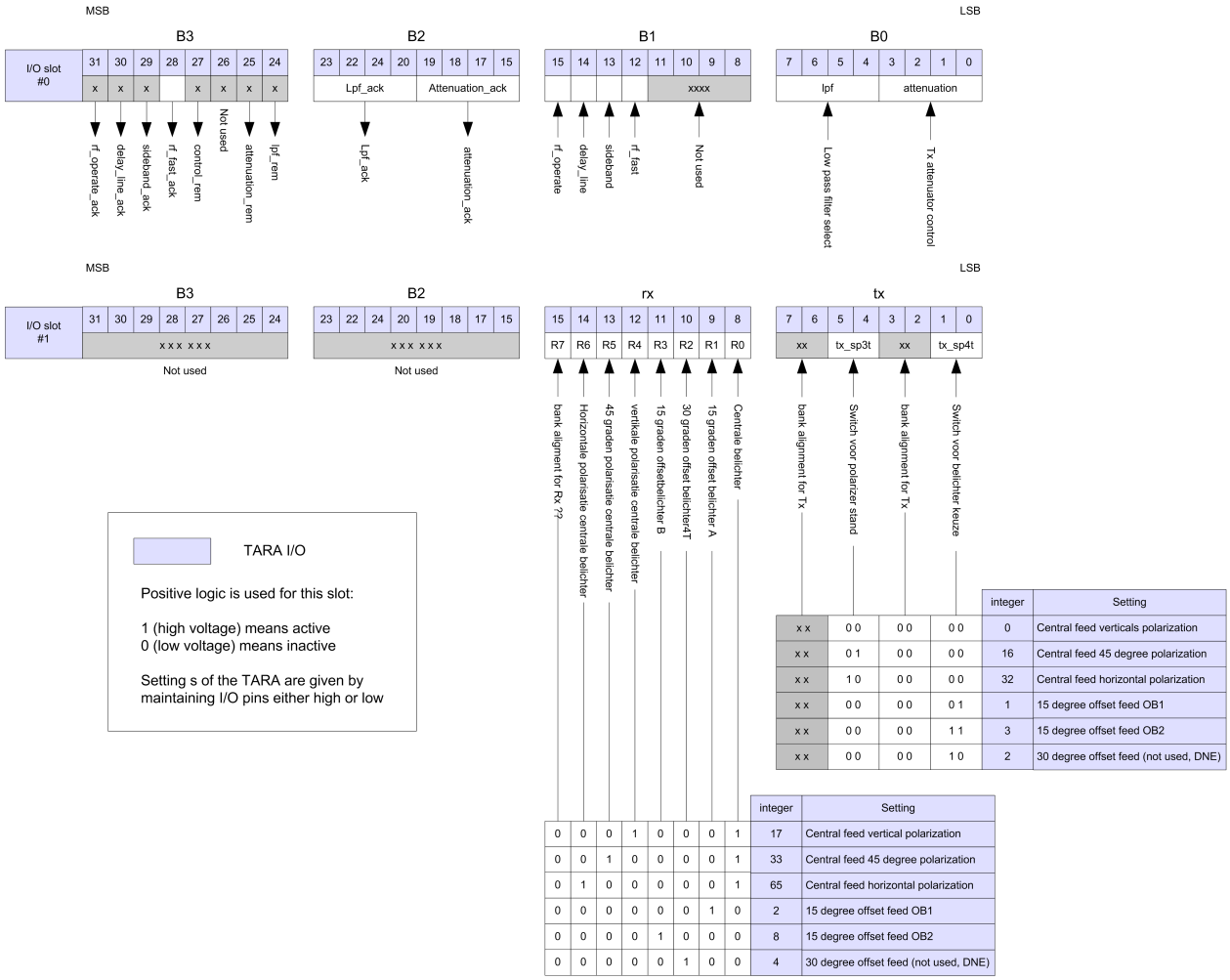


Figure F.1: TARA I/O

List of Symbols

$N_{doppler}$	Length doppler FFT
N_{pol}	Number of polarizations
N_{range}	Length range FFT
t_{sweep}	Sweep time
t_{added}	Time between two profiles to fetch the acquired data from the ADC into the computer memory
t_{ADC_setup}	ADC setup time
f_{ADC_clk}	Clock signal of the Anolog to digital convertor
$f_{DDS\_SYNC\_clk}$	Synchronization clock of the DDS
$f_{DDS\_SYS\_clk}$	System clock of the DDS
f_{ex_user}	Frequency excursion defined by the user
f_{ex_real}	Real frequency excursion
$f_{0,TARA}$	Main frequency of TARA
$f_{ADCsampler}$	Frequency of the analog to digital sampler
$f_{TARA,min}$	Start frequency of the frequency excursion
$f_{TARA,max}$	Maximum sweep frequency of the frequency excursion, used for calculating the CTW1
Δf_{DDS}	Difference between the minimum and maximum frequencies of the DDS
Δt_{DDS}	Length of one sweep
FTW	DDS-setup value containing a calculated value from the start frequency of the frequency excursion
CTW1	DDS-setup value containing a calculated value from the maximum frequency of the frequency excursion
RDW	DDS-setup value containing a calculated value from the total frequency excursion
RSRR	DDS-setup value containing a calculated value from the length of one sweep

List of Formulas

Matlab formulas

$$\Delta R = \frac{c \times \tau}{2} \tag{F.1}$$

$$f_{sample} = \frac{60\frac{MS}{s}}{n}, 4 < n < 1200 \tag{F.2}$$

$$t_{rearm} = 144 \times T_{sample\_clk} \tag{F.3}$$

$$N_{profiles} = \frac{t_{noise\_desired}}{N_{dopp} \times N_{pol} \times t_{sweep}} \tag{F.4}$$

$$f_{ADC\_clk\_op} = \frac{N_{Range}}{t_{sweep} - t_{setup\_op}} \tag{F.5}$$

$$f_{ADC\_sample} = \frac{f_{ADC\_osc}}{floor(\frac{f_{ADC\_osc}}{f_{ADC\_sample\_clk\_op}})} \tag{F.6}$$

$$t_{sampling} = \frac{N_{range}}{f_{ADC\_clk}} \tag{F.7}$$

$$t_{setup} = t_{sweep} - t_{sampling} \tag{F.8}$$

$$f_{add} = f_{exc} \times \frac{t_{sweep}}{t_{sampling}} - f_{exc} \tag{F.9}$$

Appendix G

Matlab code


```
function [ad_clk, dds_s0, dds_e0, dds_rdw, dds_rsrr, setup_time, dds_TARA_f0,↙
dds_total_excursion, ...
          F, range_res, error_out] = ...
                TARA_setup_calculation(F, Ts, Range_FFT, ad_osc, dds_sysclk, TARA_f0,↙
setup_time)
% Calculate the setup of the A/D converter and of the direct digital synthesizer
% for the TARA signal processor.
%
% Hardware used to prepare the code: National Instruments Digitzer NI PXI-5922, Analog↙
Devices DDS AD9958.
%
%   Inputs:     F            .. frequency excursion (Hz)
%               Ts           .. sweep time (s)
%               Range_FFT    .. number of range FFT samples
%               ad_osc       .. sample rate of the NI PXI-5922 (Hz),
%                               i.e. 60e6 Hz for the Digitizer NI PXI-5922
%               dds_sysclk   .. DDS reference clock (Hz), e.g. 500e6
%               TARA_f0      .. centre frequency for the DDS output (Hz), 160e6
%               setup_time   .. desired minimum setup time for the antenna beam /↙
polarisation
%                               selection (s), e.g. 50e-6
%
%
%   Outputs:    ad_clk       .. A/D converter sample rate
%               dds_s0       .. DDS start frequency, i.e. channel word 0 register
%               dds_e0       .. DDS stop frequency, i.e. channel word 1 register
%               dds_rdw      .. DDS RDW register corresponding to the frequency↙
resolution
%               dds_rsrr     .. DDS RSRR register corresponding to the time resolution
%               setup_time   .. real setup time for the antenna beam and polarisation↙
selection
%               F            .. true frequency excursion (Hz)
%               range_res    .. true range resolution (m)
%               error        .. 0 - successful execution
%                               1 - A/D sample rate too high, increase sweep time or↙
reduce range FFT samples
%                               2 - A/D sample rate too low, reduce sweep time or↙
increase range FFT samples
%
%   References: [1] AD9958 specifications document, Revision A.
%               [2] NI PXI/PCI-5922 Specifications.
```



```
% ----- setup of values that are specific fo the NI PXI-5922 and the AD9958 -----
dds_sync_clk = dds_sysclk/4;    % DDS synchronisation clock (Hz), internally derived↙
from the system clock, p. 10 of [1]
```

```
% ----- A/D converter setup calculations -----
optimal_setup_time = Ts / (1+Range_FFT/ad_rearm_time);  % (s), derived from the two equations that the minimum
                                                        %      required setup_time needs be larger than
                                                        %      (rearm_time / ad_sample_clk), and that the
                                                        %      optimal ad_sample_clk = Range_FFT / (Ts - setup_time)

ad_optimum_sample_clk = Range_FFT / (Ts - optimal_setup_time);          % (Hz)
ad_real_sample_clk    = ad_osc / floor(ad_osc / ad_optimum_sample_clk); % (Hz), always choose the next possible higher A/D sample rate

time_for_sampling     = Range_FFT * (1/ad_real_sample_clk);             % (s)
real_setup_time       = Ts - time_for_sampling;                         % (s)

% check whether the computed setup time that fullfils the requirements for the trigger rearm time of the
% A/D converter is larger than the setup time required for the antenna beam / polarisation selection
if (real_setup_time >= setup_time)
    setup_time = real_setup_time;
else
    ad_optimum_sample_clk = Range_FFT / (Ts - setup_time);                  % (Hz)
    ad_real_sample_clk    = ad_osc / floor(ad_osc / ad_optimum_sample_clk); % (Hz), always choose the next possible higher A/D sample rate

    time_for_sampling     = Range_FFT * (1/ad_real_sample_clk);             % (s)
    setup_time            = Ts - time_for_sampling;                         % (s)
end

% check whether the A/D converter clock within the specifications
if floor(ad_osc / ad_optimum_sample_clk) < ad_min_divider, error_out = 1; end
if floor(ad_osc / ad_optimum_sample_clk) > ad_max_divider, error_out = 2; end



% ---- DDS setup calculations -----

% optimise the choice of the DDS time resultion with respect to the calculeted setup time
minimum  = 10;
rsrr_opt = -1;
for i=1:255
    tmp = abs((setup_time / (i/dds_sync_clk)) - round(setup_time / (i/dds_sync_clk)));
    if (tmp < minimum),
        rsrr_opt = i;
        minimum  = tmp;
    end
end
dds_delta_t   = rsrr_opt / dds_sync_clk;                % (s),  DDS time resolution
```

```
f_additional = F * (Ts/time_for_sampling) - F;          % (Hz), frequency excursion
that needs to be
                                                        %       added for the setup
time
dds_f_start = round(TARA_f0 - (F/2 + f_additional));    % (Hz), start frequency for the
sweep
dds_f_stop  = round(TARA_f0 + F/2);                     % (Hz), stop frequency for the
sweep

dds_no_of_steps = Ts / dds_delta_t;
dds_delta_f     = (dds_f_stop - dds_f_start) / dds_no_of_steps; % (Hz), DDS frequency
resolution



% ----- set the output variables -----

dds_rdw         = ceil(dds_delta_f * 2^32 / dds_sysclk);   % DDS RDW register
(frequency step resolution)
dds_rsrr        = rsrr_opt + 256;                          % DDS RSRR register (time
resolution) setup for
                                                           % ramp up + finest time
resolution for ramp down
dds_s0          = dds_f_start * 2^32 / dds_sysclk;         % DDS Channel Frequency
Tuning Word register
dds_e0          = dds_f_stop * 2^32 / dds_sysclk;          % DDS Channel Word 1
register
ad_clk          = ad_real_sample_clk;                      % A/D converter sample clock
dds_TARA_f0     = TARA_f0 * 2^32 / dds_sysclk;             % TARA centre frequency used
for single-tone DDS output
                                                           % total frequency excursion
used to ramp down in one step
dds_total_excursion = (dds_f_stop - dds_f_start) * 2^32 / dds_sysclk;



% ----- compute the true frequency excursion and range resolution -----
dds_real_delta_f = (dds_rdw / 2^32) * dds_sysclk;
f_start = (setup_time / dds_delta_t) * dds_real_delta_f + dds_f_start; % (Hz), DDS
frequency when A/D sampling starts
f_end   = (Ts / dds_delta_t) * dds_real_delta_f + dds_f_start;         % (Hz), DDS
frequency when A/D sampling stops

F         = f_end - f_start;            % (Hz), true frequency excursion
range_res = 299792458  / (2*F);         % (m),  true radar range resolution

end
```