UNIVERSITAT POLITÈCNICA DE CATALUNYA
BARCELONATECH

ROBUST TRACKING SYSTEM IN FRONT OF RADIO-FREQUENCY INTERFERENCES. PART II: TRANSCEIVERS AND ERROR ESTIMATION

A Degree Thesis
Submitted to the Faculty of the
Escola Tècnica d'Enginyeria de Telecomunicació de Barcelona
Universitat Politècnica de Catalunya
by
Patrick James Cara Reyes

In partial fulfilment
of the requirements for the degree in
TELECOMMUNICATIONS SYSTEMS ENGINEERING

Advisors: Adriano José Camps Carmona
Jorge Querol Borràs

Barcelona, May 2016

Abstract

This is a project developed in the Theory of the Signal and Communication (TSC) Department in the Universitat Politècnica de Catalunya (UPC). The objective of this project is to track a transmitter position offering a high resistance to Radio Frequency Inference (RFI)

In order to implement the system, GPS data, the study of stability with Allan Variance and the delay estimation are used to check the precision of the Robust Tracking System (RTS). Also the transmitter is implemented to send a positioning signal to update the position every second.

The results obtained of the Allan Variance are in order of tenths of meters of accuracy. Also the communication between the transceivers has been achieved satisfactory.

Resum

Es tracta d'un projecte desenvolupat en el Departament de Teoria del Senyal i Comunicacions (TSC) de la Universitat Politècnica de Catalunya (UPC). L'objectiu d'aquest projecte és realitzar un seguiment de la posició del transmissor que ofereix una alta resistència a la Inferència de Ràdio freqüència (RFI)

Per tal d'implementar el sistema, les dades del GPS, l'estudi de l'estabilitat amb la Variància d'Allan i l'estimació del retard s'utilitzen per comprovar la precisió del Sistema Robust de Seguiment (RTS). També el transmissor s'implementa per enviar un senyal de posicionament per actualitzar la posició cada segon.

Els resultats obtinguts en la Variància d’Allan són de l'ordre de dècimes de metres d'exactitud. També la comunicació entre els transceptors s'ha aconseguit satisfactòriament.

Resumen

Se trata de un proyecto desarrollado en el Departamento de Teoría de la Señal y Comunicaciones (TSC de la Universidad Politécnica de Cataluña (UPC). El objetivo de este proyecto es realizar un seguimiento de la posición del transmisor que ofrece una alta resistencia a la Radio Inferencia de frecuencia (RFI)

Con el fin de implementar el sistema, los datos del GPS, el estudio de la estabilidad con la Varianza de Allan y la estimación de retardo se utilizan para comprobar la precisión del Sistema Robusto de Seguimiento (RTS). También el transmisor se implementa para enviar una señal de posicionamiento para actualizar la posición cada segundo.

Los resultados obtenidos en la Varianza Allan son del orden de décimas de metros de exactitud. También la comunicación entre los transceptores se ha logrado satisfactoriamente.

Dedication

To my family and friends,

for trusting me and always supporting me.

Acknowledgements

I wish to express my sincere thanks to Dr. Adriano Camps, Professor of the Signal Theory and Communications Department at the Universitat Politècnica de Catalunya, for giving me the chance to work with him and trust in me during this last years.

Also, my sincere thanks to Jorge Querol, PhD candidate of the Signal Theory and Communications Department at Universitat Politècnica de Catalunya, for his implication, guidance, help and patience with us every day to accomplish our goal.

I am also grateful to Dayal Kewalramani, project partner, friend and student, at the Universitat Politècnica de Catalunya, for sharing with me all these years, and being a great support in this project.

Special thanks to, Joaquim Giner, Ruben Tardío and Albert Marton, the laboratory managers at the Signal Theory and Communications Department of the Universitat Politècnica de Catalunya, to help us every day to make our work at the laboratory easier.

Finally for my parents, girlfriend, relatives and friends because without their support during this years I would never arrive here.

Revision history and approval record

Revision	Date	Purpose
0	14/04/2016	Document creation
1	11/05/2016	Document revision
2	11/05/2016	Document modification
3	12/05/2016	Document revision

DOCUMENT DISTRIBUTION LIST

Name	e-mail
Patrick James Cara Reyes	pjames_c_r@hotmail.com
Jorge Querol	jorge.querol@tsc.upc.edu
Adriano Camps	camps@tsc.upc.edu

Written by:		Reviewed and approved by:	
Date	14/04/2016	Date	dd/mm/yyyy
Name	Patrick James Cara Reyes	Name	Adriano Camps Jorge Querol
Position	Project Author	Position	Project Coadvisor Project Coadvisor

Table of contents

List of Figures

List of Tables:

1. Introduction

This project is related to the Radio Frequency Inference (RFI) threat which is becoming a serious problem in devices that use low power signals such as GNSS. The main goal of this project is to design, implement and test a Robust Tracking System (RTS) that offers a high resistance to RFI signals avoiding GNSS (GPS, Galileo, etc.) vulnerabilities. RTS might be used in applications that require high reliability such as transport of dangerous goods or vehicle tracking. Is needed one moving tracked emitter and, at least, three receivers strategically placed with known position as shown in Figure 1.

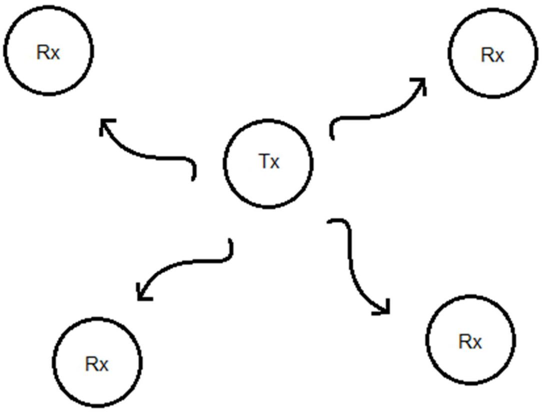


Figure 1 General overview

The project is carried out at the Signal Theory and Communications (TSC) department of UPC-BarcelonaTech, and in particular, in the Remote Sensing Laboratory (RSLab) research group.

This part of project consists in the determination of clock's stability, the implementation of the transceivers and the delay estimation between the transmitter and the receivers. Figure 2 and figure 3 shows a detailed block diagram of the transmitter and receiver devices. The principal elements of the systems are the Adafruit GPS, which receives the GPS signal, the 3DR transceivers, the microBlaze, which process the measures of frequency, the delay estimation and the latitude and longitude, and the RTS server, which process all the information to position the transmitter. The difference between the devices is that in the transmitter, a code is generated and sent to the receivers, and the receivers calculate the differential delay between them.

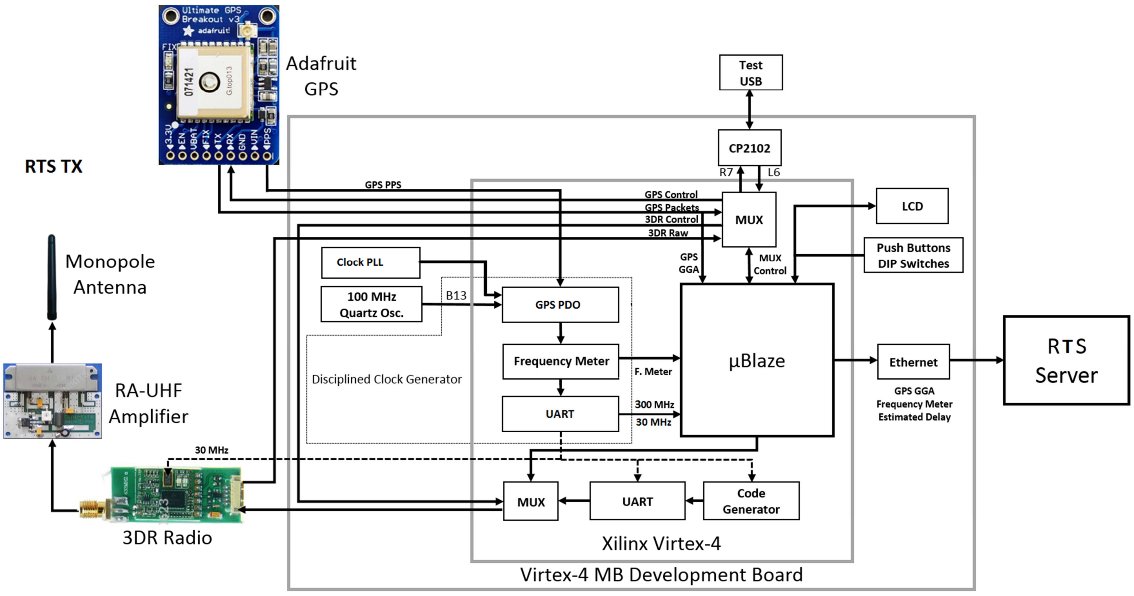


Figure 2 Transmitter block diagram overview

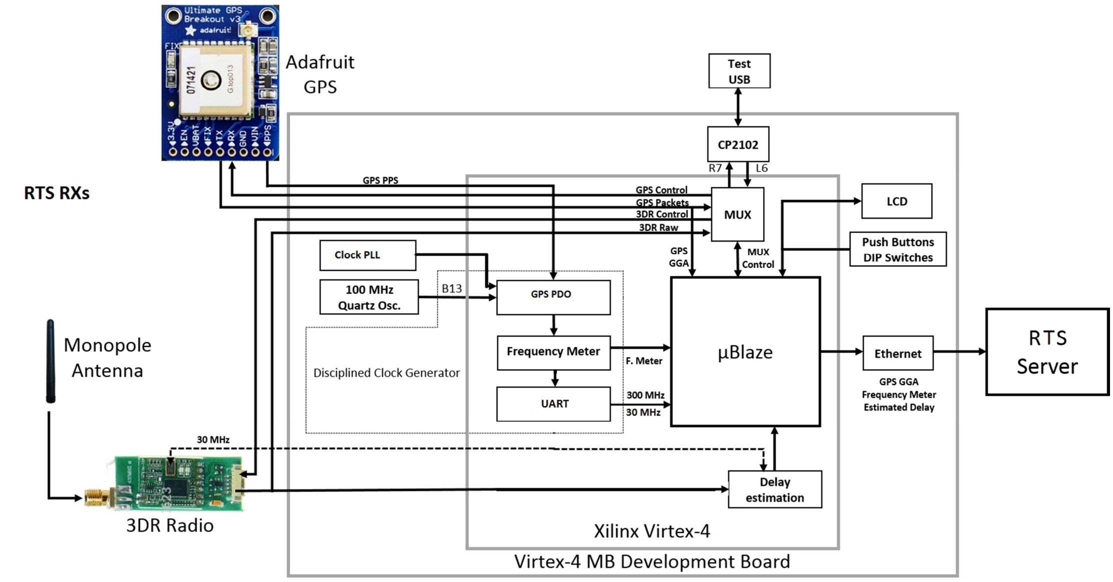


Figure 3 Receiver block diagram overview

1.1. Statement of purpose (objectives).

The project main goals are:

- Understand the scope of the project and how to achieve the results.
- Study the GPS pseudo-disciplined oscillator (GPSPDO) and the internal clocks of the Xilinx evaluation board stability.
- Implement the transmitter and the communication between the transceivers.
- Estimate the delay and the error between the transceivers.

1.2. Requirements and specifications.

Project requirements:

- Test and improve the performance of a GPSPDO, and determine its frequency and stability by means of a custom-made frequency meter. These measurements will be used to calculate the accuracy in the estimation of the position. This accuracy is expected to be of the order of tenths of meters, similar to other classical terrestrial positioning systems such as LORAN-C.
- Build up, at least, 1 moving transmitter and, at least, 3 receivers based each one on a Xilinx evaluation board, where the GPSPDO is implemented, together with a 3DR transceiver working at 433 MHz.
- Transmit and receive a pulse between the nodes every second, which will be used to estimate the delay and calculate the distance between them.
- Obtain the estimation of this differential delay using the previously characterized clock from the GPSPDO, and send it to the processing server.

Project specifications:

- Minimum receiver sensitivity: -117dBm. 3DR specifications[1].
- Open UHF band (433 MHz) is used to reduce propagation losses as compared to S-band (2.4 GHz) for example.
- The power gain at the transmitter will be set up to achieve a working range of the order of 100 km.

- Use of a low-cost Adafruit GPS to obtain the Pulse Per Second (PPS) signal necessary for proper working of the GPSPDO.
- Distance error estimation should be around 100 meters.
- Duty cycle is 1 second but without synchronism between nodes.
- Bit rate is 115.2 Kbps

1.3. Methods and procedures.

The project will be developed in the framework of one of the research lines carried out by Professor Adriano Camps at the RSLab, which is the detection and mitigation of RFI signals. In particular, the concept of RTS is part of the Ph.D. thesis of this TFG codirector Jorge Querol, and its first studies about RFI signals and their effects on GNSS devices will be taken into account during the development of the project. Moreover, a prototype of the GPSPDO has been implemented previously to the beginning of the project. However, its performance will be characterized and improved during the development of the project.

This project is divided in two parts. This is the second part “Transceivers and error estimation”, and the first part, “Communications and hyperbolic navigation”, was perfeormed by Dayal Kewalramani. The two parts are connected and we will develop them together, especially the final test.

1.4. Work plan with tasks, milestones and a Gantt diagram.

Work Breakdown Structure

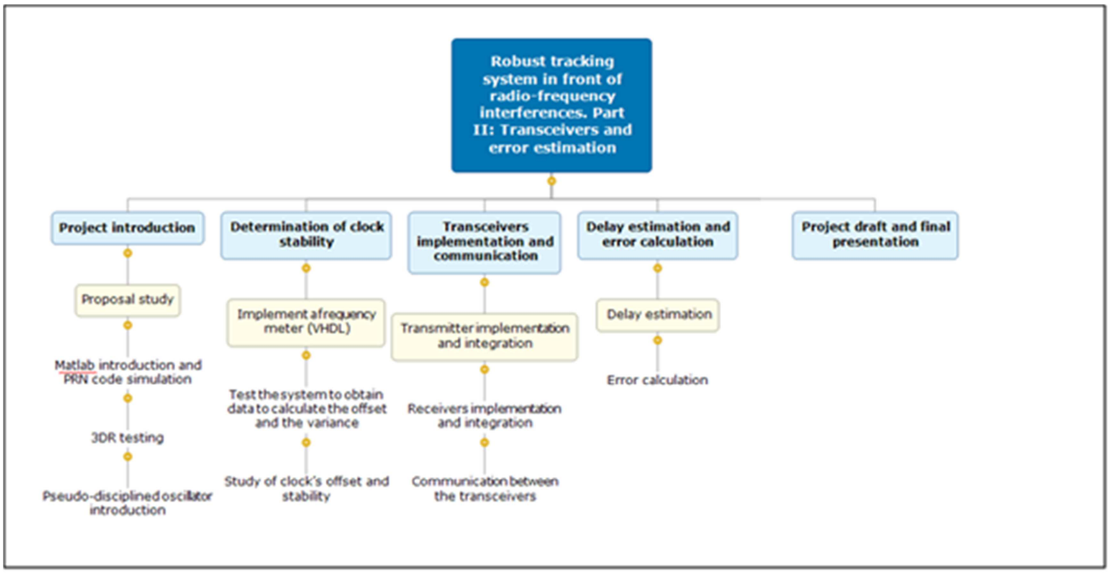

Robust tracking system in front of radio-frequency interferences. Part II: Transceivers and error estimation
Project introduction
Proposal study
Matlab introduction and PRN code simulation
3DR testing
Pseudo-disciplined oscillator introduction
Determination of clock stability
Implement a frequency meter (VHDL)
Test the system to obtain data to calculate the offset and the variance
Study of clock's offset and stability
Transceivers implementation and communication
Transmitter implementation and integration
Receivers implementation and integration
Communication between the transceivers
Delay estimation and error calculation
Delay estimation
Error calculation
Project draft and final presentation

Figure 4 Work breakdown structure

Work Packages

Project: Project introduction	WP ref: 1	
Major constituent: Preliminary work	Sheet 5 of 36	
Short description: Understand and learn all the project variables, requirements and specifications, the following steps to take and the previous simulations.	Planned start date: 30/01/15 Planned end date: 29/05/15	
	Start event: Meetings End event: Matlab simulations and 3DR test.	
Internal task T1: Proposal study. Internal task T2: Matlab introduction and PRN code simulation. Internal task T3: 3DR testing. Internal task T4: Pseudo-disciplined oscillator introduction.	Deliverables: Results	Dates: 29/05/15

Project: Determination of clock stability	WP ref: 2
Major constituent: Simulation	Sheet 5 of 36
Short description: Implement a frequency meter and a pseudo-disciplined clock (100MHz) in a Xilinx evaluation board. Using a PPS signal (Adafruit GPS) check if the frequency meter always counts 100MHz. The data obtained will let to calculate the frequency offset and the stability (Allan's variance). Also use a real frequency meter to calibrate the system and see if the results are good enough.	Planned start date: 16/09/15 Planned end date: 20/11/16
	Start event: Learn VHDL language End event: Frequency offset and variance results

Internal task T1: Implement a frequency meter (VHDL). Internal task T2: Test the system to obtain data to calculate the offset and the variance. Internal task T3: Study of clock's offset and stability	Deliverables: Frequency offset and variance results	Dates: 20/11/16

Project: Transceivers implementation and communication	WP ref: 3	
Major constituent: Integration	Sheet 6 of 36	
Short description: Implement the transmitter to send every second a bit to the receivers. Add an amplifier to increase the range. Think how the transmitter board will be used outside the laboratory. Integrate and implement the receivers (3DR) in Xilinx evaluation board to read the bit received.	Planned start date: 20/11/16 Planned end date: 30/03/16	
	Start event: Transmitter implementation End event: Communication between the transceivers	
Internal task T1: Transmitter implementation and integration Internal task T2: Receivers implementation and integration Internal task T3: Communication between the transceivers	Deliverables: Communication between the transceivers	Dates: 30/03/16

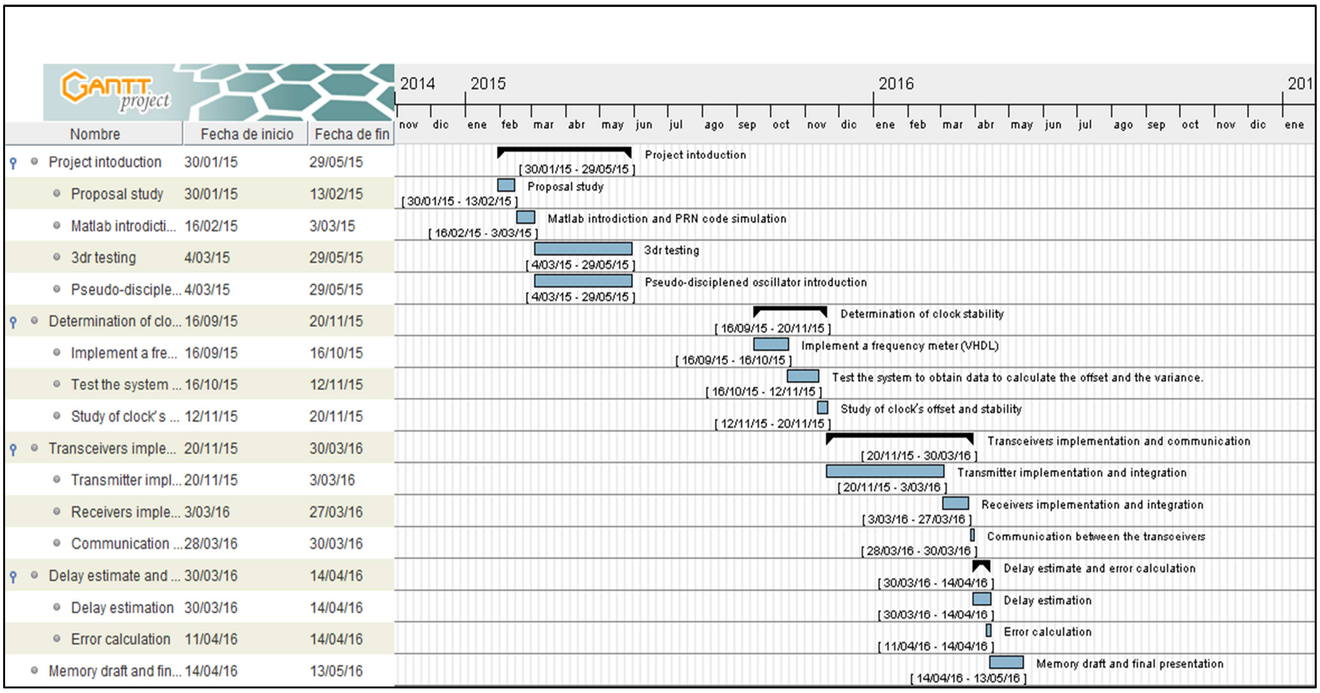
Project: Delay estimation and error calculation	WP ref: 4	
Major constituent: Estimation	Sheet 7 of 36	
Short description: Estimate the differential delay between receivers comparing their edges and see how many milliseconds of difference are in each one from the GPSPDO. Also calculate the error that will generate the clock and how affects it in the error position. Compare the practical results with the theory. These measurements are used to calculate the position of the transmitter.	Planned start date: 30/03/16 Planned end date: 14/04/16	
	Start event: Delay estimation End event: Results	
Internal task T1: Delay estimation Internal task T2: Error calculation	Deliverables: Results	Dates: 14/04/16

Project: Project draft and final presentation	WP ref: 5	
Major constituent: Writing	Sheet 7 of 36	
Short description: Write in the final document the entire project and prepare the presentation.	Planned start date: 14/04/16 Planned end date: 13/05/16	
	Start event: Project writing End event: Final project	
	Deliverables: Project memory	Dates: 13/05/16

Milestones

WP	Task	Short title	Milestone / deliverable	Date (week)
1	1	Proposal study		13/02/15
1	2	Matlab introduction and PRN code simulation	PRN code simulation	03/03/15
1	3	3DR testing	SNR values	29/05/15
1	4	Pseudo-disciplined oscillator introduction		29/05/15
2	1	Implement a frequency meter (VHDL).	Frequency meter in VHDL	16/10/15
2	2	Test the system to obtain data to calculate the offset and the variance	Obtain data of the system stability	12/11/15
2	3	Study of clock's offset and stability	Process the results	20/11/15
3	1	Transmitter implementation and integration	Implement the transmitter	03/03/16
3	2	Receivers implementation and integration	Implement the receivers	27/03/16
3	3	Communication between the transceivers	Communicate the transceivers	30/03/16
4	1	Delay estimation	Estimate the delay	14/04/16
4	2	Error calculation	Calculate the error	14/04/16
6	1	Project memory	Project memory	13/05/16

Gantt Diagram



Nombre	Fecha de inicio	Fecha de fin
Project intoduction	30/01/15	29/05/15
Proposal study	30/01/15	13/02/15
Matlab introdicti...	16/02/15	3/03/15
3dr testing	4/03/15	29/05/15
Pseudo-disciple...	4/03/15	29/05/15
Determination of clo...	16/09/15	20/11/15
Implement a fre...	16/09/15	16/10/15
Test the system ...	16/10/15	12/11/15
Study of clock's ...	12/11/15	20/11/15
Transceivers imple...	20/11/15	30/03/16
Transmitter impl...	20/11/15	3/03/16
Receivers imple...	3/03/16	27/03/16
Communication ...	28/03/16	30/03/16
Delay estimate and ...	30/03/16	14/04/16
Delay estimation	30/03/16	14/04/16
Error calculation	11/04/16	14/04/16
Memory draft and fin...	14/04/16	13/05/16

GANTT project
2014
2015
2016
201
nov dic ene feb mar abr may jun jul ago sep oct nov dic ene feb mar abr may jun jul ago sep oct nov dic ene
Project intoduction
[30/01/15 - 29/05/15]
Proposal study
[30/01/15 - 13/02/15]
Matlab introdiction and PRN code simulation
[16/02/15 - 3/03/15]
3dr testing
[4/03/15 - 29/05/15]
Pseudo-disciplened oscillator introduction
[4/03/15 - 29/05/15]
Determination of clock stability
[16/09/15 - 20/11/15]
Implement a frequency meter (VHDL)
[16/09/15 - 16/10/15]
Test the system to obtain data to calculate the offset and the variance.
[16/10/15 - 12/11/15]
Study of clock's offset and stability
[12/11/15 - 20/11/15]
Transceivers implementation and communication
[20/11/15 - 30/03/16]
Transmitter implementation and integration
[20/11/15 - 3/03/16]
Receivers implementation and integration
[3/03/16 - 27/03/16]
Communication between the transceivers
[28/03/16 - 30/03/16]
Delay estimate and error calculation
[30/03/16 - 14/04/16]
Delay estimation
[30/03/16 - 14/04/16]
Error calculation
[11/04/16 - 14/04/16]
Memory draft and final presentation
[14/04/16 - 13/05/16]

Figure 5 Gantt diagram

1.5. Description of the deviations from the initial plan and incidences that may have occurred.

The most important setback in the initial planning was the retrieval of the measurements of the implemented frequency meter. At the beginning, the plan was to use the frequency of the Xilinx evaluation board (oscillator 100 MHz). Then, converting the frequency with a GPS pseudo-disciplined oscillator (GPSPDO) to obtain the 10 MHz reference. Finally, convert this 10 MHz to 100 MHz with an external Phase-Locked Loop (PLL). After that, the next step is the obtaintion of these three frequency measures. The problem was that the PLL had some limitations. The 10 MHz given by the GPSPDO could not be converted well with the PLL because the generated clock is not exactly 10 MHz all the time. For this reason, the PLL did not generate the 100MHz clock properly.

The solution adopted was to divide all the system in two different oscillator sources. The first one is to convert the 100MHz of the Xilinx evaluation board to 300 MHz using a DCM (Digital Clock Manager), and then to use the GPSPDO in order to obtain a disciplined 30 MHz clock. The other way is to use the PLL to generate another 300 MHz clock, and discipline it with a second GPSPDO. The change from 100 MHz to 300 MHz has been chosen because the higher frequency at the GPSPDO input, the better the stability that can be obtained with the 30 MHz clock GPSPDO. However, frequency measurements are not synchronized because they are written and read with two different clocks, for this reason the measurements are not read correctly in the serial port. The solution was to use a register (FIFO) and modify the signal Pulse Per Second (PPS) detector. Finally, the objective is to obtain the Allan variance of each system and choose the best one. These setbacks have caused that the rest tasks have been delayed several weeks.

Another incidence that affects the rest of the tasks was the problem when the 3DR radio is connected and transmitting his identifier.

2. State of the art of the technology used or applied in this thesis

In this chapter, the state of the art of the technology used or applied in this project is presented considering the environment, technologies and the literature used to start the project.

GPS signal is used to obtain the PPS signal and the GGA information. The PPS signal is used to create the GPS PDO. The Allan Variance characterizes the stability of the clocks created by the GPS PDO and the internal clocks of Xilinx evaluation board. Finally, the distance between the transmitter and the receivers is calculated by the delay estimation

2.1. Introduction Global Navigation Satellite Systems (GNSS) and Global Position System (GPS)

The United States of America GPS consists of up to 32 Medium Earth Orbit satellites in six different orbital planes, with the exact number of satellites varying as older satellites are retired and replaced. Operational since 1978 and globally available since 1994, GPS is currently the world's most utilized satellite navigation system.

GPS only requires 4 satellites to determine 3D positioning and timing and guarantees 5 satellites in line of sight at any location.

Other Global Navigation Satellite Systems (GNSS) are already deployed as the russian Global'naya Navigatsionnaya Sputnikovaya Sistema (GLONASS) or the two systems in current development as Galileo by European Space Agency (ESA) or COMPASS from China also known as BeiDou-2.

The tracking algorithm designed at our department uses GPS GGA NMEA sentence to locate the transmitter and the receivers. National Marine Electronics Association (NMEA) is a standard data format supported by all GPS manufacturers [2]:

2.2. Allan Variance

The tool to calculate the clock's stability used in this project is the Allan Variance (AVAR). The AVAR is a statistical parameter applicable to a continuous series of values of fractional frequency, calculated through the following equations [3]:

$$y(t_0) = \frac{f(t_0)-f_0}{f_0} \text{ ; (1)}$$

Here the f_0 is the nominal frequency and the $f(t_0)$ the instantaneous frequency at time t_0. For instance, if the nominal frequency is 300 MHz and the frequency calculated in this instant is 300,0000002 MHz, the fractional frequency will be (300.000002 MHz – 300 MHz)/ 300MHz= 6.67 x 10^9 MHz.
The expression to calculate the AVAR is [4]:

$$\sigma_y{}^2(\tau) = \frac{1}{2(N-1)}\sum_{i=1}^{N-1}(y_{i+1} - y_i)^2\text{; (2)}$$

The AVAR estimates the deviation on frequency, or stability, of an oscillator and performs better when more samples are considered.

2.3. Delay estimation

The technique that will be used in this project to obtain the distance between the transmitter and the receivers is the delay estimation. The transmitter will sent a positioning signal and depending on the moment that the receiver receives the first edge of the data, it will provide information about the distance using the light speed.

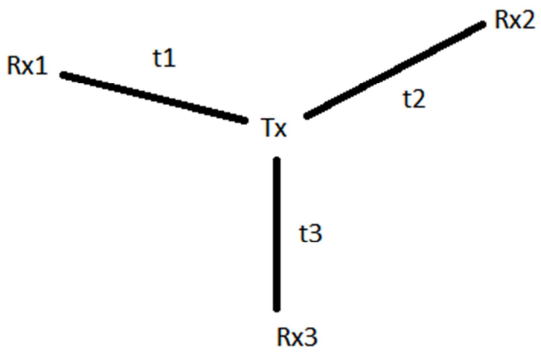


Figure 6 Delay beetwen the transceivers

3. Determination of the clock`s stability:

3.1. Frequency measurements

The main idea is to choose the best clock source for the system in order to have the minimum error in the estimation of the positions of the transceivers. The Xilinx evaluation board has two clocks. The first one is the 100 MHz Quartz oscillator and the second one is the clock PLL. At the beginning of the project the frequency chose was 100 MHz, but later, as explained in the incidences part, this frequency caused problems and was changed to 300 MHz.

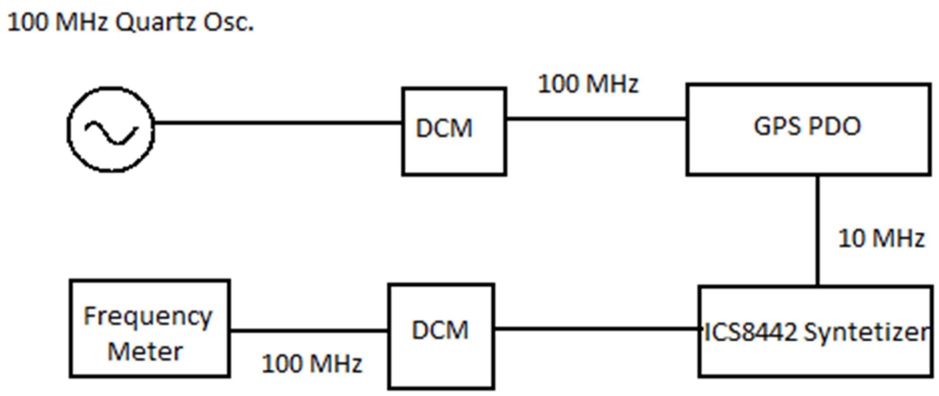


Figure 7 First approach to calculate the reference frequencies

Finally, the frequency selected was 300 MHz. To obtain this value a DCM (Digital Clock Manager) of the Xilinx evaluation board is used. Both clocks were modified to obtain 30MHz. This frequency is obtained in the block GPS Pseudo-Disciplined Oscillator (GPS PDO) that was created by the Project Coadvisor Jorge Querol and slightly modified by me. The GPS PDO uses the signal Pulse Per Second (PPS) given by the Adafruit GPS to obtain a purest signal.

Figure 8 GPS ANT-55 antenna

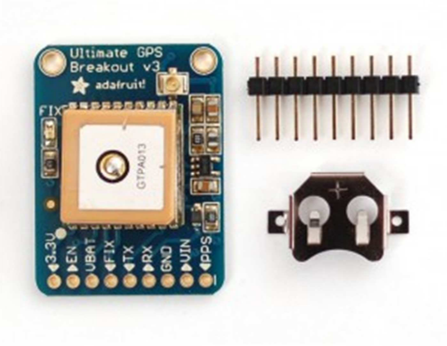


Figure 9 Adafruit GPS

Also in this block, there is a Frequency meter that calculates the frequency of each signal every second. Then, the block generates a 32 bit vector with the measure of each frequency (300 MHz and 30 MHz from 100 MHz Quartz oscillator and 300 MHz and 30 MHz from clock PLL). With the help of a real frequency meter, the measures can be checked and see if the clocks really have the frequency desired. However, frequency measurements are not synchronized because they are written and read with two different clocks, for this reason the measurements are not read correctly in the serial port. The solution was to use a register First In, First Out (FIFO) and modify the signal Pulse Per Second (PPS) detector. Once the data passes to the FIFO, a 128 bit vector with the four measurements is generated. To send this vector to the computer by the serial port, it is necessary to create a parallel to serial block to convert the vector and then implement a Universal Asynchronous Receiver-Transmitter (UART) [5]. Then, the plan is to leave on the board during a long time to save the four frequencies measured every second using a MATLAB code. Figure 11 shows the block diagram used to obtain the measures.

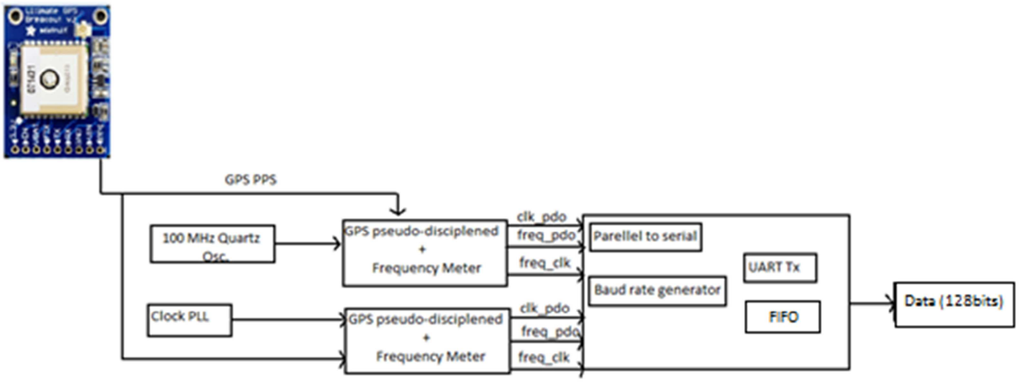


Figure 10 Frequency meter system

3.2. Allan Variance

After a weekend saving data, 259.200 samples were saved (1 every second). The results that MATLAB saved in a .txt file, are loaded and processed. First of all, the fractional frequency is calculated and then the AVAR. The AVAR resulting from the measurements are represented in figure 12

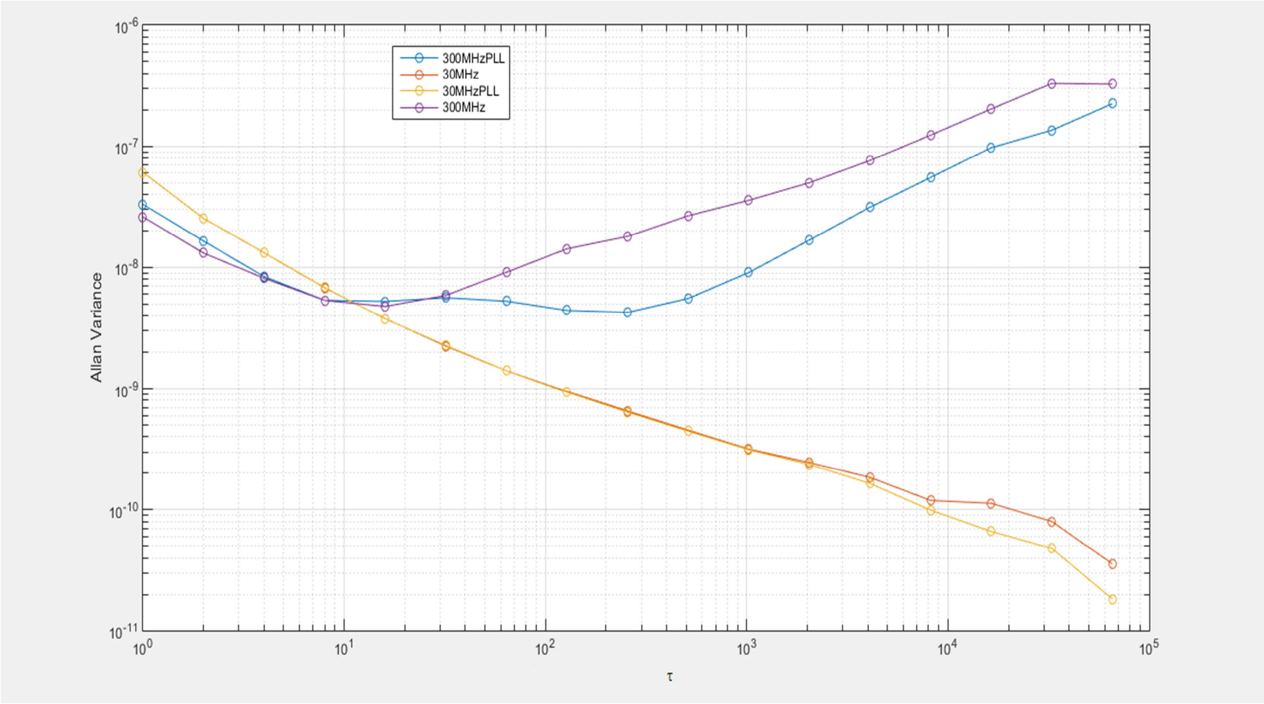


Figure 11 AVAR results

In this graph the error of the four frequencies can be seen. There are four frequencies because the more stable one to obtain the better precision in the estimation of the distance will be chosen. The purple and blue lines represent the 300 MHz of both clocks (100 MHz quartz oscillator and clock PLL). Both lines start a falling curve, arriving to a minimum, and then following a rising curve. The other two lines represent the 30MHz generated by the GPS PDO. As opposed to the 300 MHz lines, the 30 MHz follows a falling curve. The reason is that the disciplined oscillators have better stability because they are generated by the GPSPDO. Current oscillators have a AVAR form like the 300 MHz. Seeing the graphs, the better clock is the PLL and the most precise one in the estimation of the distance. Taking $\tau = 10s$, the AVAR value is 6x10^{-9} . Doing the product between the AVAR value and the light speed 3x10^{8} we obtain a precision of 18 metres as we expected in the project proposal.

4. Transceivers implementation and communication

4.1. Transmitter implementation

As explained in the introduction, the idea is to have one transmitter and several receivers. The transmitter has to send every second a positioning signal. To implement the transmitter a 3DR transceiver is required to be connected to the Xilinx evaluation board. Inside the board, a counter and a Read-Only Memory (ROM) are programmed to achieve our goal. As it has been chosen in the Determination of clock stability, the 300 MHz PLL is the best one because it will cause the smallest error in the estimation of the distance. A 300.000.000 counter has to be created. Once the counter achieves this value, a binary signal is activated and used to enable a ROM that generates the vector of bytes used as positioning signal. This vector will be also used as an identifier of the transmitter. Then, the vector will be sent to a block that will convert the parallel bytes to serial and after passing through a transmission UART, the identifier can be sent by the 3DR transmitter. Connecting another 3DR[4] transceiver in a computer and using a the TeraTerm program, can be checked if the identifier is received every second. With the help of an oscilloscope, the bits frame can be checked and the time of bit can be calculated, in this case is 8.64 microseconds.

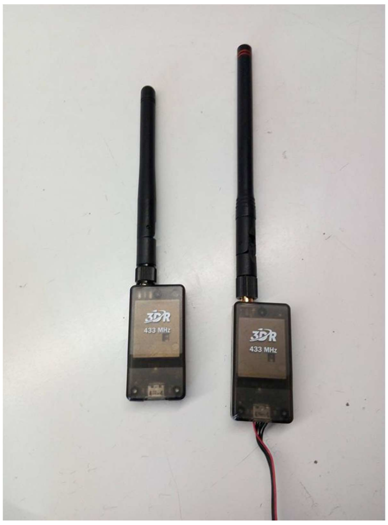
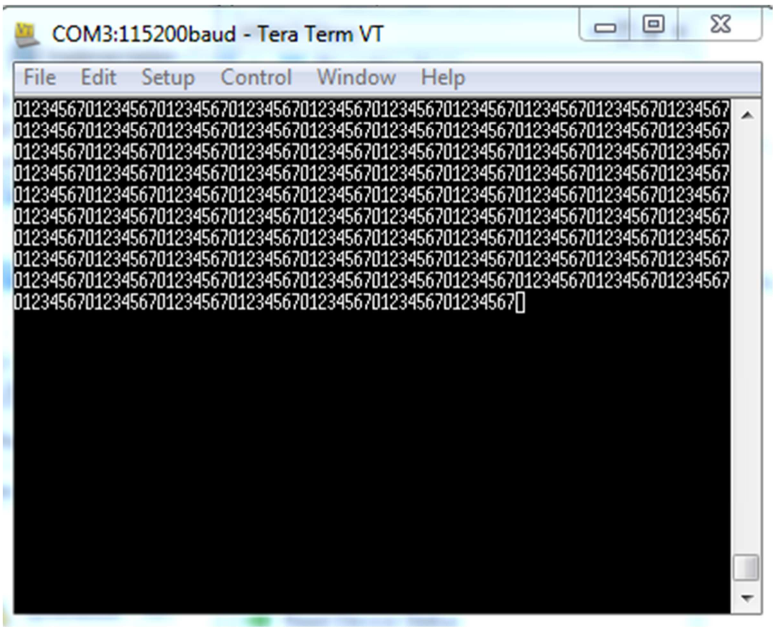


Figure 12 3DR receiver and transmitter

Selecting the pins of the Xilinx evaluation board, the transmitter implementation can be checked. In Figure 14, there is a screenshot of the Tera Term. The ID of the transmitter is “01234567” and is received every second.



```
01234567012345670123456701234567012345670123456701234567012345670123456701234567
01234567012345670123456701234567012345670123456701234567012345670123456701234567
01234567012345670123456701234567012345670123456701234567012345670123456701234567
01234567012345670123456701234567012345670123456701234567012345670123456701234567
01234567012345670123456701234567012345670123456701234567012345670123456701234567
01234567012345670123456701234567012345670123456701234567012345670123456701234567
01234567012345670123456701234567012345670123456701234567012345670123456701234567
01234567012345670123456701234567012345670123456701234567012345670123456701234567
01234567012345670123456701234567012345670123456701234567012345670123456701234567
01234567012345670123456701234567012345670123456701234567
```

Figure 13 Tera Terms creenshot showing the ID sent every second

The problem is that when the 3DR is connected, the ID is not correct because the bits are out of phase. Checking it with the oscilloscope, the bit frame is good, so the problem is in the 3DR transmitter. This is not a very bad problem because the ID can be modified to be recognizable, but trying to solve this problem, several weeks were wasted and endangered the rest of the project.

4.2. Receiver implementation

The receiver has to receive the identifier. Once the information is received by the 3DR transceiver, send it to the Xilinx evaluation board. Creating a reception UART, the data passes from serial to parallel. Connecting now a 3DR transceiver to a computer and opening two TeraTerms windows, the reception system can be tested. Sending characters from the 3DR (transmitter) connected to the computer and checking if they really arrive in the receiver.

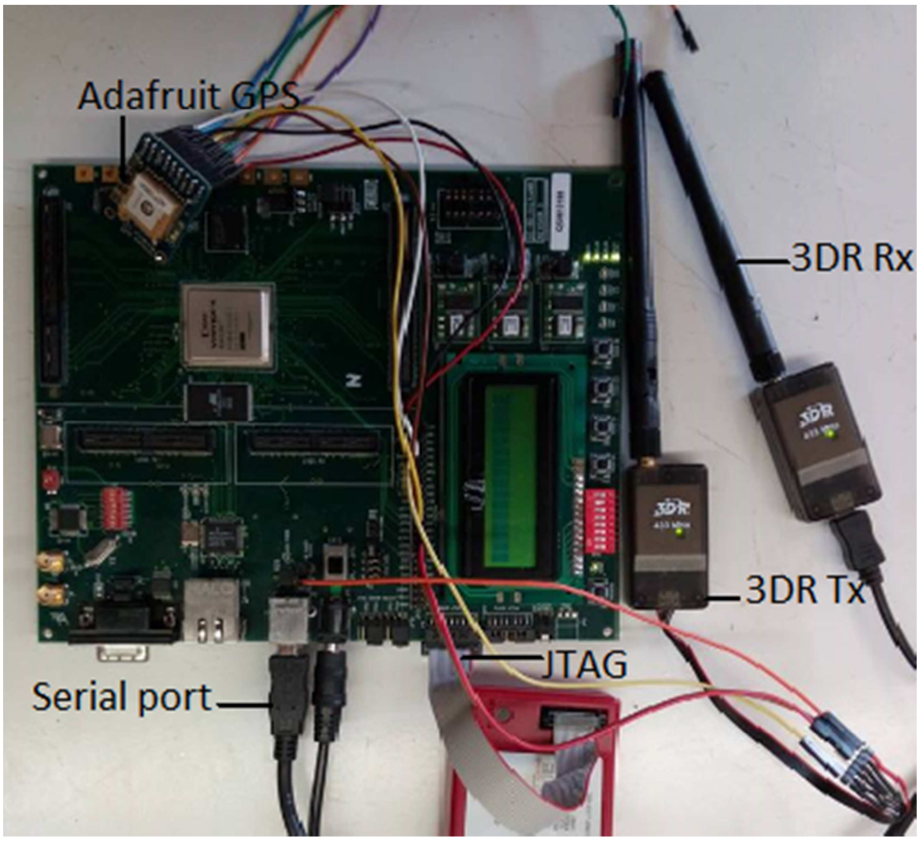


Figure 14 Transceivers connection

4.3. Sending latitude and longitude

A HUAWEI dongle is used to send the information from Adafruit GPS to the RTS Server configured by Dayal Kewalramani. First of all, a MATLAB code chooses the part of needed information (GGA latitude and longitude). Once is selected, it goes to the dongle. The dongle sends it to the next address:

```
tcpip('172.26.39.56',20008,'LocalPort', 20000);
```

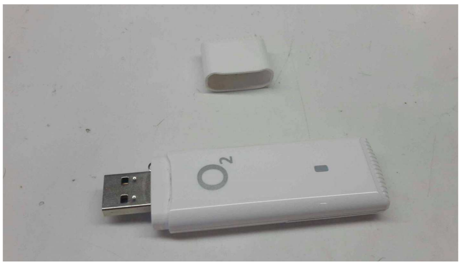
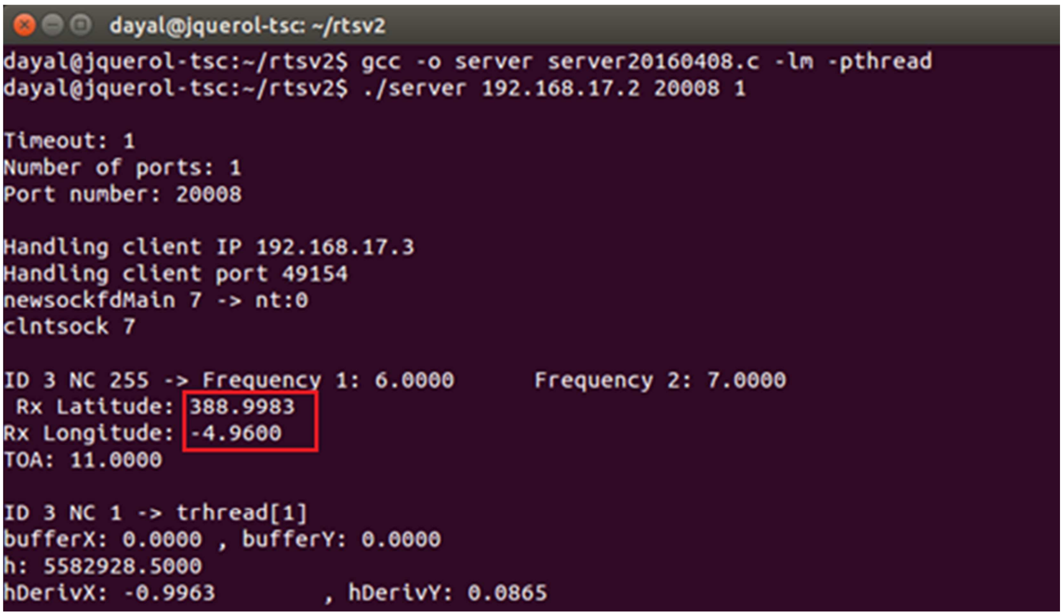


Figure 15 HUAWEI dongle

This values, the latitude and longitude, arrive to the RTS server and can be checked on Linux terminal.



```
dayal@jquerol-tsc:~/rtsv2$ gcc -o server server20160408.c -lm -pthread
dayal@jquerol-tsc:~/rtsv2$ ./server 192.168.17.2 20008 1

Timeout: 1
Number of ports: 1
Port number: 20008

Handling client IP 192.168.17.3
Handling client port 49154
newsockfdMain 7 -> nt:0
clntsock 7

ID 3 NC 255 -> Frequency 1: 6.0000      Frequency 2: 7.0000
 Rx Latitude: 388.9983
Rx Longitude: -4.9600
TOA: 11.0000

ID 3 NC 1 -> trhread[1]
bufferX: 0.0000 , bufferY: 0.0000
h: 5582928.5000
hDerivX: -0.9963        , hDerivY: 0.0865
```

Figure 16 RTS server receiving latitude and longitude

5. Delay estimation and error calculation

The process begins once the 3DR Rx receives the identifier (ID) from the transmitter. The information goes to two ways. The first one goes to a UART and then to a ROM that identifies the Information and checks if this was sent by our transmitter. The other way is the delay estimation block. When the ID arrives in this block, a signal is generated when the first edge of the information is detected. At the same time, there is a counter at 300 MHz. When the first edge detector signal is activated, the block saves the value of the counter in this moment. This value will be called Time Of Arrival (TOA). Using the speed light and the value of TOA, the system will be able to determinate de position of each receiver. The final step is to send the TOA to the mircroBlaze as seen in Figures 2 and 3. The microBlaze also will receive the latitude and longitude and the frequency each second. Figure 18 shows the process graphically.

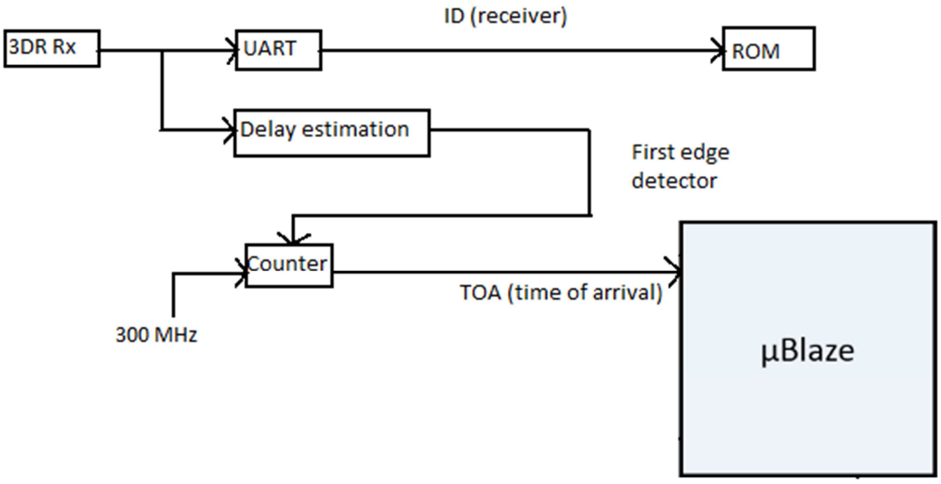


Figure 17 Delay estimation system

This part of the project only lacks the implementation of the 300MHz counter because the signal first edge detector is already created. And then the calculation of TOA will be sent to the microBlaze.

In Figure 19, a screenshot of a simulation from ISE Project Navigator program (ISE Design Suite 14.7) is shown. “Datain” represents the positional signal received by the receiver. Signal “q” represents the first edge detector of the positional signal. When the signal “datain” is active, the signal “q” is activated to, so the first edge detector is working well.

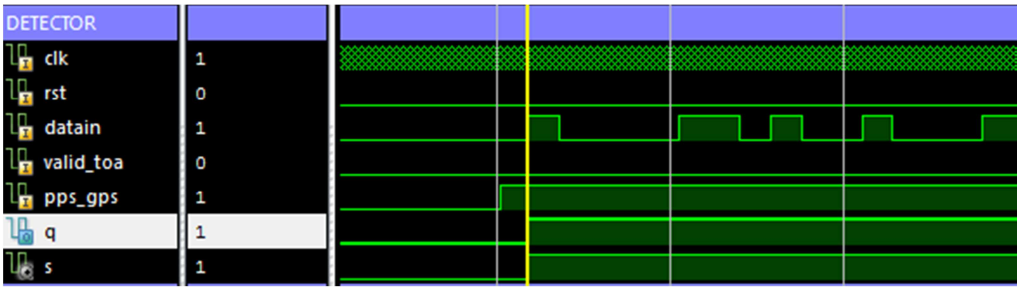


Figure 18 Simulation of first edge detector

Clk: 300 MHz PLL clock.

Rst: the system reset.

Datain: the ID received.

Valid_toa: this signal is activated when the TOA’s value is valid

Pps_gps: signal PPS from GPS.

Q: first edge detector signal.

S: auxiliary signal.

6. Budget

The total budget of the project development is:

	Cost (€)
Indirect cost (40%)	3.254,1
Instruments	135,25
Personnel	8.000
Total	11.389,35

Table 1: Total budget

The indirect costs of the project as, the place, electrical energy are the 40 % of the instrument budget and personnel budget.

The cost of material and personnel is:

Intrument	Price (€)	Years	Amortization/hour (€)	Used (h)	Cost (€)
Tester	10	10	0,000114155	8	0,00
Laptop	500	10	0,011415525	1000	11,42
Adafruit GPS v3.0	39,95	3	0,001520167	60	0,09
GPS antenna	30	5	0,000684932	60	0,04
Xilinx Virtex-4 DB	1.500	10	0,017123288	1000	17,12
Miscellaneous	10	1	0,001141553	60	0,07
3DR Radio v2 (x2)	30 (x2)	3	0,002283105	700	1,60
MATLAB	2000	5	0,0456621	500	22,83
ISE	3.595	5	0,082077626	1000	82,08
Total					135,25

Table 2: Instrument and material budget

Name	Rank	€ hour/pers	Total hours	Cost (€)
Patrick James Cara Reyes	**Junior Engineer**	8	1.000	8.000
Total				8.000

Table 3: Personnel budget

7. Conclusions and future development:

Summarizing, the determination of clock stability is calculated well with the Allan Variance and the accuracy of RTS is in order of tenth meters. The transceivers implementation is good except the problem of the ID, but it can be solved. The communication between the transmitter and the receiver is working well. Finally the part of the delay estimation, as commented in point 5, only lacks the counter. Implementing it, the system will be able to calculate the TOA. Doing this, all the values that microBlaze need, will be given to it (Frequency, latitude, longitude and TOA).

The main points in the future work are: connect an RA-UHF amplifier to the 3DR transmitter, test all the system with Dayal Kewalramani project and do a long range distance test of the system to see the behaviour in real field and assure a good SNR in the communication,

8. Bibliography:

[1] Anonymous. “3DR Radio v2 Quick Start Guide”, 3DR Robotics, 2013. [Online] Available: http://www.3dr.com. [Accesed: 13 May 2015].J. P. Wilkinson, “Nonlinear resonant circuit devices,” U.S. Patent 3 624 125, July 16, 1990.

[2] Anonymous. "Standard NMEA-0183 sentences description". *NMEA 4.00, November 2008*. [Online] Available: http://www.freenmae.net/docs. [Accessed: 22 May 2015].

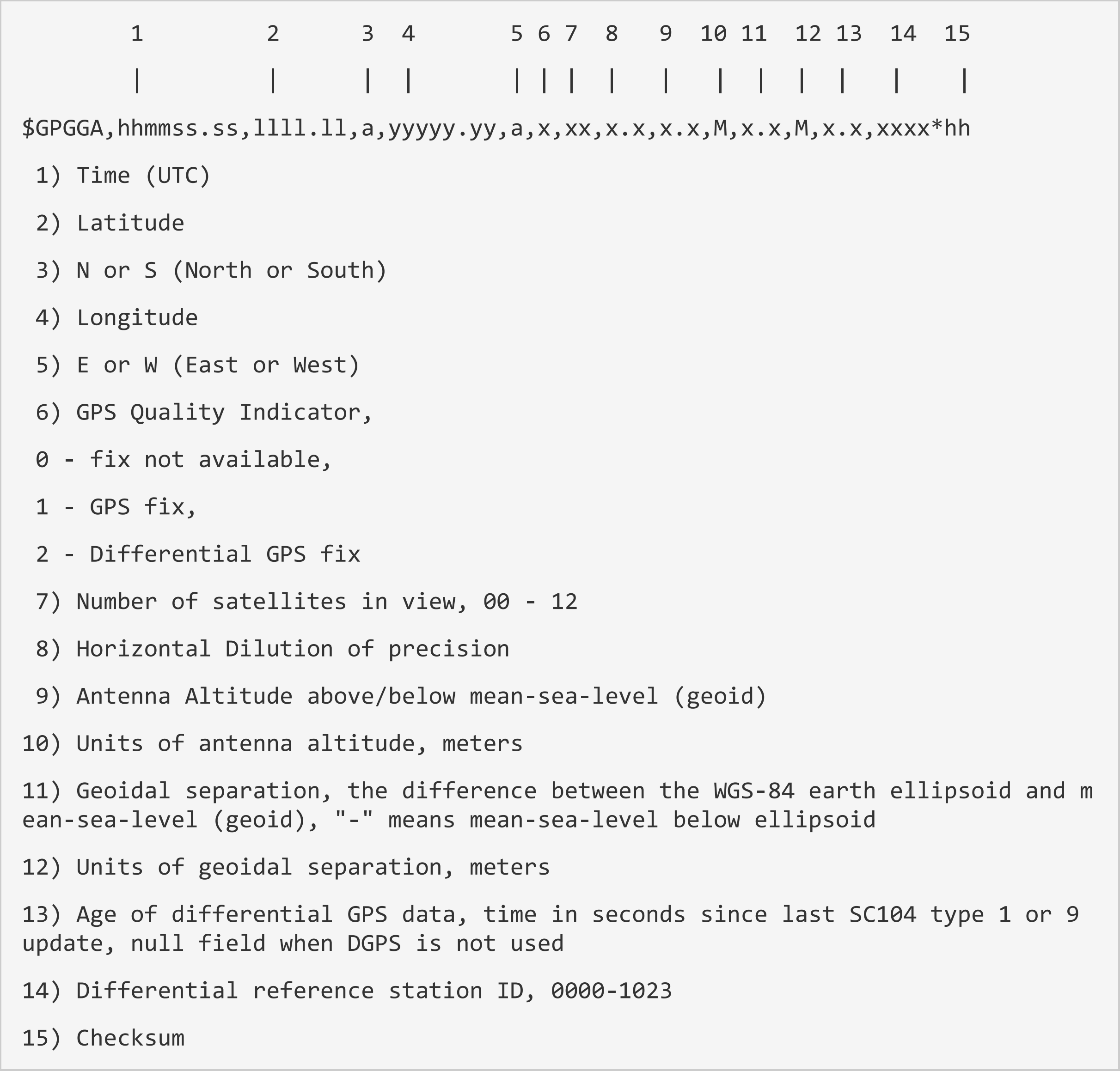
[3] Anonymous. "¿Sabías que la varianza de Allan es el estadístico adoptado para la caracterización de la inestabilidad de relojes de precisión?". [Online] Available: http://www.e-medida.es/documentos/Numero-7/sabias-que-varianza-allan. [Accessed: 10 Oct 2015].

[4] Wikipedia : Allan Variance [Online] Available: https://en.wikipedia.org/wiki/Allan_variance#Time_error [Accesed: 10 Oct 2015].*Internet Protocol, Version 6 (IPv6) Specification*. IETF RFC 2460, December 1998.

[5] Anonymous. " Hardware Design with VHDL Design Example: UART [Online] Available: http://ece-research.unm.edu/jimp/vhdl_fpgas/slides/UART.pdf. [Accessed: 30 Sep 2015].

9. Appendices (optional):

9.1. GGA string



```
        1         2      3  4       5 6 7  8   9  10 11  12 13  14  15
        |         |      |  |       | | |  |   |   |  |  |  |   |     |
$GPGGA,hhmmss.ss,llll.ll,a,yyyyy.yy,a,x,xx,x.x,x.x,M,x.x,M,x.x,xxxx*hh
 1) Time (UTC)
 2) Latitude
 3) N or S (North or South)
 4) Longitude
 5) E or W (East or West)
 6) GPS Quality Indicator,
 0 - fix not available,
 1 - GPS fix,
 2 - Differential GPS fix
 7) Number of satellites in view, 00 - 12
 8) Horizontal Dilution of precision
 9) Antenna Altitude above/below mean-sea-level (geoid)
10) Units of antenna altitude, meters
11) Geoidal separation, the difference between the WGS-84 earth ellipsoid and m
ean-sea-level (geoid), "-" means mean-sea-level below ellipsoid
12) Units of geoidal separation, meters
13) Age of differential GPS data, time in seconds since last SC104 type 1 or 9
update, null field when DGPS is not used
14) Differential reference station ID, 0000-1023
15) Checksum
```

Figure 19 GPS GGA NMEA sentence

Where:

GP is Global Position

GGA is System Fix Data

And we need number 2, 3, 4 and 5 flags.

9.2. Allan Variance code (MATLAB)

```
%Allan Variance Matlab Code

%Monday, 06 August 2007

%ALLAN VARIANCE MATLAB SOURCE CODE
function [sig,sig2,osig,msig,tsig,tau]=avar(y,tau0)
% function [sig,sig2,osig,msig,tsig,tau]=avar(y,tau0)
% INPUTS:
% y = signal
% tau0 = sampling period (s)
%
% OUTPUTS:
% sig = N samples STD DEV
% sig2 = Normal Allan STD DEV, 2 samples STD DEV.
% osig = Sigma(y)(tau) = Allan Standard Deviation with Overlapping
estimate
% msig = Modified Allan Standard Deviation
% tsig = Time Allan Standard Deviation
% tau = measurement time (s).
%
% Copyright Alaa MAKDISSI 2003
% free for personal use only.s=[];
x=[];
n=length(y);
jj=floor( log((n-1)/3)/log(2) );for j=0:jj
fprintf('.');
m=2^j;
tau(j+1)=m*tau0;
D=zeros(1,n-m+1);
for i=1:n-m+1
D(i)=sum(y(i:i+m-1))/m;
end
%N sample
sig(j+1)=std(D(1:m:n-m+1));
%AVAR
sig2(j+1)=sqrt(0.5*mean((diff(D(1:m:n-m+1)).^2)));
%OVERAVAR
z1=D(m+1:n+1-m);
z2=D(1:n+1-2*m);
u=sum((z1-z2).^2);
osig(j+1)=sqrt(u/(n+1-2*m)/2);

%MVAR
u=zeros(1,n+2-3*m);
for L=0:n+1-3*m
z1=D(1+L:m+L);
z2=D(1+m+L:2*m+L);
u(L+1)=(sum(z2-z1))^2;
end

uu=mean(u);
msig(j+1)=sqrt(uu/2)/m;

%TVAR
tsig(j+1)=tau(j+1)*msig(j+1)/sqrt(3);

end
```

9.3. Process the frequencies saved with allan variance code

```
D = dir('Frequencyresults201511*.txt');
mydata = load(D(6).name, '-ascii');
A = mydata(:,4);
B = mydata(:,3);
G = mydata(:,5);
D = dir('Frequencyresults201511*.txt');
mydata = load(D(8).name, '-ascii');
N = mydata(:,4);
H=fracfreq(A,300000000);
P=fracfreq(N,300000000);
I=fracfreq(B,30000000);
J=fracfreq(G,30000000);
```

9.4. UART Tx

```
entity tx_uart is

       generic
            (
                  DBIT: integer := 8;
                  SB_TICK: integer := 16
                  );
    Port ( clk : in  STD_LOGIC;
           rst : in  STD_LOGIC;
           tx_start : in  STD_LOGIC;
           s_tick : in  STD_LOGIC;
           din : in  STD_LOGIC_VECTOR (7 downto 0);
           tx_done_tick : out  STD_LOGIC;
           tx : out  STD_LOGIC);
end tx_uart;

architecture arch of tx_uart is

            type state_type is (idle, start, data, stop);
            signal state_reg, state_next: state_type;
            signal s_reg, s_next: unsigned(3 downto 0);
            signal n_reg, n_next: unsigned(2    downto 0);
            signal b_reg, b_next: std_logic_vector(7 downto 0);
            signal tx_reg, tx_next: std_logic;

begin
```

```
process (clk, rst) -- FSMD state and data regs.
            begin
            if (rst = '1') then
                  state_reg <= idle;
                  s_reg <= (others => '0');
                  n_reg <= (others => '0');
                  b_reg <= (others => '0');
                  tx_reg <= '1';

            elsif (clk'event and clk='1') then
                  state_reg <= state_next;
                  s_reg <= s_next;
                  n_reg <= n_next;
                  b_reg <= b_next;
                  tx_reg <= tx_next;
            end if;
            end process;

                  -- next state logic
      process (state_reg, s_reg, n_reg, b_reg, s_tick,tx_reg, tx_start,
din)
                  begin
                  state_next <= state_reg;
                  s_next <= s_reg;
                  n_next <= n_reg;
                  b_next <= b_reg;
                  tx_next <= tx_reg;
                  tx_done_tick <= '0';

                  case state_reg is
                        when idle =>
                              tx_next <= '1';
                              if (tx_start = '1') then
                                    state_next <= start;
                                    s_next <= (others => '0');
                                    b_next <= din;
                              else
                                    state_next <= idle;
                                    s_next <= (others => '0');
                                    n_next <= (others => '0');
                                    b_next <= (others => '0');
                                    tx_next <= '0';
                                    tx_done_tick <= '0';
                              end if;
                        when start =>
                              tx_next <= '0';
                              if (s_tick = '1') then
                                    if (s_reg = 15) then
                                          state_next <= data;
                                          s_next <= (others => '0');
                                          n_next <= (others => '0');
                                    else
                                          s_next <= s_reg + 1;
                                    end if;
                              end if;
                        when data =>
                              tx_next <= b_reg(0);
                              if (s_tick = '1') then
                                    if (s_reg = 15) then
                                          s_next <= (others => '0');
                                          b_next  <=  '0'  &  b_reg(7  downto
1);
                                          if (n_reg = (DBIT - 1)) then
                                                state_next <= stop;
                                          else
                                                n_next <= n_reg + 1;
                                          end if;
                                    else
                                          s_next <= s_reg + 1;
                                    end if;
                              end if;
                        when stop =>
                              tx_next <= '1';
                              if    (s_tick = '1') then
                                    if (s_reg = (SB_TICK-1)) then
                                          state_next <= idle;
                                          tx_done_tick <= '1';
                                    else
                                          s_next <= s_reg + 1;
                                    end if;
                        end if;
                    end case;
                end process;
            tx <= tx_reg;
        end arch;
```

9.5. UART Rx

```
entity rx_uart is
        generic(
            DBIT: integer := 8;
            SB_TICK: integer := 16
            );
        port(
            clk, reset: in std_logic;
            rx: in std_logic;
            s_tick: in std_logic;
            rx_done_tick: out std_logic;
            dout : out std_logic_vector(7 downto 0)
        );
end rx_uart;

architecture arch of rx_uart is
        type state_type is (idle, start, data, stop);
        signal state_reg, state_next: state_type;
        signal s_reg, s_next: unsigned(3 downto 0);
        signal n_reg, n_next: unsigned(2 downto 0);
        signal b_reg, b_next: std_logic_vector(7 downto 0);

begin

process(clk, reset) -- FSMD state and data regs.
            begin
            if (reset = '1') then
                state_reg <= idle;
                s_reg <= (others => '0');
                n_reg <= (others => '0');
                b_reg <= (others => '0');
            elsif (clk'event and clk='1') then
                  state_reg <= state_next;
                  s_reg <= s_next;
                  n_reg <= n_next;
                  b_reg <= b_next;
            end if;
end process;
                  -- next state logic
process (state_reg, s_reg, n_reg, b_reg, s_tick, rx)
                  begin
                        state_next <= state_reg;
                        s_next <= s_reg;
                        n_next <= n_reg;
                        b_next <= b_reg;
                        rx_done_tick <= '0';

                        case state_reg is

                              when idle =>
                                    if (rx = '1') then
                                          state_next <= start;
                                          s_next <= (others => '0');
                                    end if;

                              when start =>
                                    if (s_tick = '1') then
                                          if (s_reg = 15) then
                                                state_next <= data;
                                                s_next    <=    (others    =>
'0');
                                                n_next    <=    (others    =>
'0');
                                          else
                                                s_next <= s_reg + 1;
                                          end if;
                                    end if;

                              when data =>
                                    if (s_tick = '1') then
                                          if (s_reg = 15) then
```

```
                                s_next   <=   (others   =>
'0');
                                b_next  <=  rx  &  b_reg(7
downto 1);
                                if  (n_reg  =  (DBIT  -  1))
then
                                    state_next <= stop;
                                else
                                n_next <= n_reg + 1;
                                end if;
                            else
                                s_next <= s_reg + 1;
                            end if;
                        end if;

                    when stop =>
                        if (s_tick = '1') then
                            if (s_reg = (SB_TICK-1)) then
                                state_next <= idle;
                                rx_done_tick <= '1';
                            else
                                s_next <= s_reg + 1;
                            end if;
                        end if;
            end case;
    end process;
dout <= b_reg;

end arch;
```

9.6. GGA code (MATLAB)

```
t = tcpip('172.26.39.56',20008,'LocalPort', 20000);

fopen(t);

s = serial('COM3','BaudRate',9600, 'InputBufferSize',2000,'Timeout',10);
fopen(s);

I = 15;
% Write to the host and read from the host.
for i = 1:I
        FB= fscanf(s,'%s\n');
        D='$GPGGA';
        C = strncmp(FB,D,6)

         if C==1

             B=FB;

             N = B(19:29)
             E=B(31:42)
             fprintf(t, '%s\n', sprintf('%s\t %s\t', N, E)); %mandar
postcion
         end


pause(0.5);
end
fclose (s);
fclose(t);

clear t;
```

10. Glossary

AVAR Allan Variance

DCM (Digital Clock Manager)

FIFO First In, First Out

GLONASS Global'naya Navigatsionnaya Sputnikovaya Sistema

GNSS Global Navigation Satellite Systems

GPS Global Positioning System

GPSPDO GPS pseudo-disciplined oscillator

ID Identifier

PPS Pulse Per Second

ROM Read-Only Memory

RFI Radio Frequency Interferences

RLS Recursive Least Squares

RTS Robust Tracking System

SNR Signal to Noise Ratio

TOA Time of Arrival

UART Universal Asynchronous Receiver-Transmitter