Timing Signals and Radio Frequency Distribution Using Ethernet Networks for High Energy Physics Applications

A thesis submitted for the degree of
Doctor of Philosophy (Ph.D)

by

Pedro Manuel Oliveira Fernandes Moreira

Department of Electrical and Electronic Engineering
University College of London

September 2014

Statement of Originality

I, Pedro Moreira, confirm that the work presented in this thesis is my own. Where information has been derived from other sources, I confirm that this has been indicated in the thesis.

Signed:

Date:

05/09/2014

Abstract

Timing networks are used around the world in various applications from telecommunications systems to industrial processes, and from radio astronomy to high energy physics. Most timing networks are implemented using proprietary technologies at high operation and maintenance costs.

This thesis presents a novel timing network capable of distributed timing with sub-nanosecond accuracy. The network, developed at CERN and codenamed "White-Rabbit", uses a non-dedicated Ethernet link to distribute timing and data packets without infringing the sub-nanosecond timing accuracy required for high energy physics applications.

The first part of this thesis proposes a new digital circuit capable of measuring time differences between two digital clock signals with sub-picosecond time resolution. The proposed digital circuit measures and compensates for the phase variations between the transmitted and received network clocks required to achieve the sub-nanosecond timing accuracy. Circuit design, implementation and performance verification are reported.

The second part of this thesis investigates and proposes a new method to distribute radio frequency (RF) signals over Ethernet networks. The main goal of existing distributed RF schemes, such as Radio-Over-Fibre or Digitised Radio-Over-Fibre, is to increase the bandwidth capacity taking advantage of the higher performance of digital optical links. These schemes tend to employ dedicated and costly technologies, deemed unnecessary for applications with lower bandwidth requirements. This work proposes the distribution of RF signals over the "White-Rabbit" network, to convey phase and frequency information from a reference base node to a large numbers of remote nodes, thus achieving high performance and cost reduction of the timing network. Hence, this thesis reports the design and implementation of a new distributed RF system architecture; analysed and tested using a purpose-built simulation environment, with results used to optimise a new bespoke FPGA implementation. The performance is evaluated through phase-noise spectra, the Allan-Variance, and signal-to-noise ratio measurements of the distributed signals.

Acknowledgements

First and foremost, I would like to express my sincere gratitude to my supervisor Professor Izzat Darwazeh for his advice, encouragement and support during my PhD road. He is an outstanding professor and wonderful person for a student to have as a role model. Thank you for the critical attitude, great freedom and confidence to undertake and conclude this research work, which became the best document I have created.

I would like to express much gratitude to my supervisor at CERN Javier Serrano for introducing me to the area of timing systems, hardware design, and for your endless support in providing the necessary equipment during the research program. I also would like to thank my CERN's colleagues Tomasz Wlostowski and Pablo Alvarez for all the guidance and great discussions.

I have the fortune of being surrounded by great people along this path. From UCL, I would like to thank my officemates Sypros, Yo Chen, Ryan, Manoj for the long days.

From Geneva, I would like to thank Mário Pereira, Aleksandra, Tiina, Ignacio, Andrea, Edu and Miguel Santos for their valuable friendship.

And not forgetting my London's housemates Sukh, Tim and Eshan with whom I had a great time during the beginning of the research program.

A special word of appreciation go to Prof. Canas Ferreira and Prof. Machado e Silva for the technical support and office logistics during the period I worked from the University of Porto. Also, very grateful thanks are due to Pedro Mota for all the discussions and great mood.

To my great friend Miguel Pimenta, thank you very much for all the encouragement and for the life experiences we shared during this period. I will never forget those amazing spicy lunches in the basement of the Korean Supermarket at Store Street.

I am grateful to the Portuguese Foundation for Science and Technology (FCT) for their doctoral scholarship SFRH/BD/60294/2009.

A very special thanks go to Cláudia, for the friendship, happiness and love she brings to my life. She gave me the confidence to carry on when I needed the most.

Finally, I am indebted to my magnificent parents and great sisters for their love, their endless support and words to never loose the North and to work hard to bring this research work to a successful end.

University College London — Pedro Moreira
September 2014

"Employ your time in improving yourself by other men's writings, so that you shall gain easily what others have labored hard for."

Socrates

"The only source of knowledge is experience."

Albert Einstein

"A person who never made a mistake never tried anything new."

Albert Einstein

Contents

List of Figures

List of Tables

List of Abbreviations

ADEV	Allan Deviation
BMC	Best Master Clock algorithm
CDR	Clock and Data Recovery
CRS	Carrier Sense Signal
DFF	D Flip-Flop
DDS	Direct Digital Synthesiser
EBI	External Bus Interface
FPGA	Field-Programmable Gate Array
GbE	Gigabit Ethernet
GMII	Gigabit Media Independent Interface
GMT	General Machine Timing
HEP	High Energy Physics
IP	Intellectual Property
LAN	Local Area Network
LPF	Low Pass Filter
LVDS	Low-voltage differential signalling
LVPECL	Low-voltage positive emitter-coupled logic
MAC	Media Access Control
NTP	Network Time Protocol
PCS	Physical Coding Sublayer
PDF	Probability Density Function
PHY	Physical Layer
PLL	Phase Locked Loop

PMA	Physical Medium Attachment Layer
PON	Passive Optical Network
PPS	Pulse per Second
PTP	Precise Time Protocol
RMS	Root Mean Square
RTL	Register-Transfer Language
SFP	Small Form-Factor Pluggable transceiver
SMF	Single Mode Fibre
SNR	Signal to Noise Ratio
SSA	Signal Source Analyser
UTC	Coordinated Universal Time
VCO	Voltage Control Oscillator
VCTCXO	Voltage Controlled Temperature Compensated Crystal Oscillator
VCXO	Voltage Control Crystal Oscillator
VHDL	VHSIC Hardware Description Language
VHSIC	Very High Speed Integrated Circuits
WDM	Wavelength Division Multiplexing

Chapter 1

Introduction

> "The only reason for time is so that everything doesn't happen at once."
>
> Albert Einstein

Time is a physical quantity, considered a fundamental property of the universe. However, there is little agreement between researchers about its nature. Clocks give us an idea about how time works, because a clock requires a movement as a gauge to measure time. It can be the oscillation of a quartz crystal [1] or the ejection of a particle from a radioactive atom [2]. When there is movement there is change in time.

There are two classical notions of time, one proposed by Isaac Newton who stated that time provides the framework in which events take place. He treats time with a big level of rigidity; only when there is enough confidence in the time frame it is possible to know the distance and the speed of an object. Einstein, on the other hand, showed that time passes at different rates depending upon an object motion and the strength of gravity pulling on it. His theory abandons the notion that space and time exist in themselves. His theory leads to the idea that change is the fundamental property of the universe and that time emerges from our mental efforts to organise the changing world we see around us.

In industrial systems there is the need to time aligned the events as they occur. In humans, the sharing of the same notion of time is used for instance to performance synchronise dance [3]. In nature, fish share the same notion of time to swim in an organised manner and thus prevent attacks from predators [4]. In telecommunication systems, time is used to allow nodes to get access and transmit data to a shared medium. All these actions depend on a process called time synchronisation. Synchronisation is the act of making synchronous the operation of different entities on the evolving to different processes by aligning their time scales.

Large control systems have distributed nodes that need to be synchronised. That is, at a given instant of time all the nodes in a system must agree with the same notion of the actual time. When this occurs we are in the presence of a synchronised system. Synchronisation is necessary in a system to establish a global ordering of events and tasks.

Synchronisation may be accomplished by having all nodes using the same external time source. For example, the Coordinated Universal Time (UTC) can be known via telephone, radio, GPS, etc. each one giving different levels of accuracy. However, when radio signal reception is not possible, such as the case of underground particle accelerators, systems need alternative synchronisation solutions. In such cases, the system may have a single central clock and communications equipment to exchange messages between the nodes and the central clock. Yet, when the central clock system fails, all nodes are left without time reference.

It is often preferable to allow each node to use its own clock, and to limit the difference between them with a synchronisation algorithm. The synchronisation algorithm must ensure that the difference between the clocks of any nodes is never higher than a given value. This threshold value defines the accuracy of the synchronisation algorithm.

Synchronisation algorithms typically have three phases: distribution of clock information, estimation of clock delay, and adjustment of clock values. They may be classified according to the methods used by nodes to inform one another of their

current clock values. Hardware algorithms use a dedicated network to broadcast the clock reference signal. They use the principle of phase-locked loops in order to achieve a tight synchronisation between the clocks with almost no time overhead. For instance, this is the synchronisation technique being currently used in the General Machine Timing (GMT) at CERN, which is discussed later in this thesis. Other algorithms make use of computer networks and messages exchange schemes to distribute synchronisation information between clocks. This type of algorithm is used for instance in Precise Time Protocol (PTP) or Network Time Protocol (NTP).

Time synchronisation networks assure the correct temporal order of information processing in communication systems, computation and control [5].

1.1 Motivations and Aims of the Work

Most timing systems deployed for High Energy Physics (HEP) applications use dedicated timing transmission links to ensure timing is delivered accurately with a bound latency and specified accuracy to the timing nodes. In addition, these timing systems are implemented using custom-made protocols due to the specific requirements present in HEP applications. Testing, debugging and operation require the development of custom-made tools that are implemented in both hardware and software layers. These factors increase the resources necessary to properly manage this type of timing networks.

The aim of the work described in this thesis is two-fold. The first goal is to investigate the use of the standard Ethernet protocol, IEEE 802.3, as an alternative for dedicated timing links, to distribute timing with sub-nanosecond accuracy over 1000 distributed nodes located 10 km apart from each other. Additional requirements are to maintain the latency of the transmitted timing and data deterministic and bounded within the network. This new synchronisation network, codename White Rabbit (WR), uses a multiplexed network topology one for data (mesh) and another for the timing (hierarchical) data. For this purpose it is necessary to design, implement and verify

the operation of digital circuits to assist the synchronisation network in achieving the purposed requirements.

The second goal is to investigate and develop a technology for the distribution of RF signals using the timing network previously proposed.

The main goal of existing distributed RF schemes, such as Radio-Over-Fibre or Digitized Radio-Over-Fibre, is to increase the bandwidth capacity taking advantage of the higher performance of the digital optical link. These schemes tend to employ dedicated and costly technologies, which are deemed unnecessary for applications with lower bandwidth requirements.

This proposed RF distribution technology aims to be more competitive and efficient than current technologies such as radio-over-fibre or digitized-radio-over-fibre in particular for applications where signal's bandwidth is relatively small, below 2MHz.

This work took place at CERN and at UCL, at CERN was part of a group project and individual contributions of the author are presented in Section 1.2.

1.2 Main Contributions

The contributions of the present work are in the area of data network communications, and design, implementation, testing of digital circuits. They are listed as follows:

- Proposed a measurement based delay model to characterise timing variations in the Ethernet physical link.
- Designed a novel digital circuit that measures time difference between two digital timing clock signals with sub-picosecond time resolution. The highest time resolution obtainable of such measurement type at the time. This digital circuit is named Digital Dual Mixer Time Difference (DDMTD).
- Proposed a set of techniques for deglitching the output timing clock signals and

designed appropriate algorithms to implement these techniques on an FPGA.

- Experimentally validated the application of the DDMTD as phase detector in a PLL design.
- Verified the application of the DDMTD as a linear phase shifter with picosecond time resolution .
- Design of an integrated system that verifies the performance of the DDMTD. This was implemented using FPGAs, DAC, VCO, fibre links, and additional hardware required to transmit and receive Ethernet frames.
- Proposed a novel distribution RF over Ethernet system architecture.
- Built a computer-based numerical model of the distribution RF over Ethernet system to estimate the phase-noise spectrum and SNR of the reconstructed RF signal.
- Design and implemented the proposed system architecture to experimentally validate the RF distributing scheme through measurements of the phase-noise spectrum, SNR, and Allan-Variance of the reconstructed RF signal.
- Experimentally investigated the performance of the proposed distributed scheme for bandpass sampling.

1.3 List of Publications

The research work reported in this thesis and the above contributions have led to seven publications in international and one technical presentation at national conferences. These are listed in chronological order below:

1. P. Moreira, J. Serrano, T. Wlostowski, P. Loschmidt, and G. Gaderer. **White Rabbit: Sub-nanosecond timing distribution over Ethernet**. International Symposium on Precision Clock Synchronization for Measurement, Control and Communication, pages 1-5, October 2009.

2. P. Moreira, P. Alvarez-Sanchez, J. Serrano, I. Darwazeh, and T. Wlostowski. **Digital dual mixer time difference for sub-nanosecond time synchronization in Ethernet**. IEEE International Frequency Control Symposium, pages 449-453, June 2010.

3. P. Moreira and I. Darwazeh. **Digital femtosecond time difference circuit for CERN's timing system**. London Communications Symposium, September 2011.

4. P. Moreira, P. Alvarez, J. Serrano, I. Darwazeh **Sub-Nanosecond Digital Phase Shifter For Clock Synchronization Applications**. IEEE International Frequency Control Symposium, June 2012.

5. P. Moreira, P. Alvarez, J. Serrano, I. Darwazeh **Distributing RF Signals In An Ethernet Network**. IEEE International Frequency Control Symposium, June 2012. This paper has been awarded the **best student paper** for the Group 5 - Timekeeping, Time and Frequency Transfer, GNSS Applications.

6. P. Moreira, P. Alvarez, J. Serrano, I. Darwazeh, M. Lipinski **Distributed DDS in a White Rabbit Network: An IEEE 1588 Application**. IEEE International Symposium on Precision Clock Synchronization for Measurement, Control and Communication, June 2012.

7. P. Moreira, I. Darwazeh **An FPGA implementation of the Distributed RF over White Rabbit**. IEEE International Frequency Control Symposium, June 2013.

In addition, the author also contributed to three publications in the scope of this research work.

1. P. Loschmidt, G. Gaderer, N. Simanic, A. Hussain, and P. Moreira. **White Rabbit - sensor/actuator protocol for the CERN LHC particle accelerator**. IEEE Conference on Sensors, pages 781-786, October 2009.

2. J. Serrano, P. Alvarez, M. Cattin, E. Garcia Cota, J. Lewis, P. Moreira, T. Wlostowski, G. Gaderer, P. Loschmidt, J. Dedic, Cosylab, R. Br, T. Fleck, M.Kreider, C. Prados, and S. Rauch. **The White Rabbit Project**. International Conference on Accelerator and Large Experimental Physics Control System, October 2009.

3. M. Lipinski, P. Alvarez, J. Serrano, P. Moreira **Performance results of the first White Rabbit installation for CNGS time transfer**. IEEE International Symposium on Precision Clock Synchronization for Measurement, Control and Communication, 2012. This paper has been awarded the **best paper** at ISPCS 2012.

1.4 Organisation

The first part of this work investigates whether Ethernet networks are suitable for control command and timing distribution, with sub-nanosecond timing accuracy.

In addition, the solutions necessary to achieve the proposed goals, without breaking the network standard, are suggested in this thesis.

Following this introductory chapter a common structure is used throughout this thesis.

All chapters start with an introductory section and are concluded with a summary of the main contributions.

The organisation of this thesis is described as follows.

Chapter 2 provides an overview of CERN's accelerator complex followed by the description of the current timing systems running at CERN that deliver accurate time-stamping services to the accelerators chain. It follows with description of the timing system supporting the LHC's detectors. Current research on the next generation of detector timing systems is also presented. Section 2.3 is dedicated to the description of the ALMA timing system, one of the most complex dedicated timing systems ever designed, currently being used in the field of radioastronomy. The chapter ends with a description of the GPS system in particular describing how timing synchronisation is achieved between the satellites and ground stations.

Chapter 3 presents the Ethernet computer network standard, with particular care to its specification for gigabit rates. The chapter starts with the description of how

data transmission is defined by the standard as well as how clock synchronisation is achieved between the Ethernet stations. The Synchronous Ethernet and IEEE 1588 standards are also discussed. The chapter continues with a description of the White Rabbit Project and how this project proposes the use of IEEE 802.3 standard, the Synchronous Ethernet and IEEE 1588, to achieve sub-nanosecond timing accuracy over approximately 1000 nodes over 10 km links. In addition, Chapter 3 describes a link delay model that is used to compensate for the link's asymmetry and phase variations in the up-link and down-link network clocks, thus achieving sub-nanosecond timing accuracy.

Chapter 4 provides background information on how to characterise the distributed clocks in a synchronous digital network. It starts by describing the techniques used to characterise frequency and phase of a timing signal in both the frequency and time domain. The jitter and wander timing noise present in synchronous networks are discussed as well as the main causes for their occurrence that degrades the performance of a synchronous network. It was characterised in frequency and time domain the phase and frequency stability of a Rubidium and a Caesium atomic clocks.

Chapter 5 describes a novel digital circuit capable of measuring phase/time differences between clock signals with a sub-picosecond time resolution using a relative low frequency counter, this circuit is named Digital Dual Mixer Time Difference (DDMTD). The occurrence of glitches in the DDMTD output clock is described. Deglitching techniques that are able to filter and remove the glitches from the DDMD output clock are also proposed. The circuit design and the proposed deglitching techniques are analysed for diverse system configurations using a simulation environment built in MATLAB.

The work follows in **Chapter 6** with the implementation of the DDMTD circuit, analysed in Chapter 5, in an FPGA's custom-made board designed at CERN. The schematic of the FPGA board is described, followed by the hardware architecture used to exchange data among the different hardware modules implemented in the

FPGA. In addition, detailed description of the major components that composes a PLL is given. It outlines the basic analysis tools that are necessary to design a PLL system. The chapter continues with the design and implementation of the two different digital PLL in the FPGA system followed by the discussion of their phase noise figures obtained by both simulations and hardware measurements. In addition, this chapter shows the DDMTD performance in a clock phase shifter application with picosecond time resolution.

The second part of this work investigates and proposes a novel technology capable of distributing RF signals over the White Rabbit network.

Chapter 7 proposes a new system architecture to distribute RF signal over a non dedicated Ethernet link. It discusses the various analogue and digital components that constitute the proposed distribution scheme. It also investigates the different noise contribution from each element of the system in the degradation of the SNR in the reconstructed signal. This analysis is conducted using a MATLAB simulation model develop to understand and evaluate the performance of system under various system scenarios. In addition, the model assists in the optimisation of the system during its implementation phase.

Chapter 8 gives a detailed account of the implementation of the proposed system architecture presented in Chapter 7, using the SPEC hardware platform. It discusses the various optimisations conducted in the hardware design so that the proposed architecture can fit in the available FPGA. Finally, it is demonstrated the performance of the architecture in distributing RF signal with measurements of the phase-noise spectrum, Allan Variance and SNR of the reconstructed signal.

Chapter 9 outlines the guidelines for future work and presents the thesis conclusions.

Chapter 2

High Energy Physics Timing Systems

Timing systems are specialised networks, which provide a common notion of time in a distributed environment. In other words, they ensure that all clocks in the system are ticking at the same rate and showing identical time at every instant. Accurate time synchronisation is a major requirement in all real-time applications. Distributed control systems, factory automation and data acquisition systems all require the execution of operations with very tight time constraints. This becomes particularly difficult in large-scale systems such as CERN's accelerator complex, where distances between nodes are with in 10 km range, causing long and often unpredictable transmission delays due to variations in the environment factors. Most of the available off-the-shelf commercial timing solutions do not meet the requirements of large complexes such as CERN. They are not easily scalable to support thousands of nodes and cannot be used over large distances as their real-time performance level would be severely reduced. Therefore, the vast majority of timing networks used on accelerators (even those commercially available) are custom-made [6].

In this chapter, the CERN's accelerator chain complex is presented, followed by the description of its main particle detectors. Next, a detailed description of the

timing system currently in operation that enables a flexible operation between the various accelerators for the different physics research applications is given. Section 2.2 presents the timing and control system employed to synchronise the LHC experiments with the accelerator beam. In addition, research work currently being conducted to upgrade the accelerators timing and control system is introduced.

This chapter continues with the description of a reference timing system currently being installed and operated in Chile, the ALMA Project. Section 2.4 describes the operation of the GPS system, namely how synchronisation is achieved between the satellites and the ground nodes.

2.1 CERN Accelerator Complex

The Large Hadron Collider (LHC) is without doubt the biggest and most complex scientific instrument built by humankind. It is located in a 27 km long underground tunnel, crossing the Swiss-French border near the city of Geneva.

The main purpose of the LHC is to extend our understanding of the Standard Model, the currently accepted theory for describing elementary particles. Specifically, LHC is used to verify the existence of the Higgs boson and to study "new" physics beyond the Standard Model such as supersymmetry or extra dimensions. There are also expectations to clarify the origin and properties of the so-called dark matter, which is believed to constitute about 23% of all the matter in the Universe [7]. Theoretical expectations place these phenomena within the teraelectronvolt (TeV) energy range, accessible by the LHC with its 14 TeV center-of-mass collision energy. The probability of the creation of interesting particles in a single collision is however very small. This justifies the need for a collider, which can simultaneously provide high beam energy and high luminosity (expressed as the number of particles in the beam per unit area per unit time) [8].

The LHC is not a standalone device, it needs an injection chain consisting of smaller accelerators, which deliver a beam of particles with the required energy, luminosity

and spatial structure. A simplified diagram of the CERN's accelerator complex is depicted in Figure 2.1.

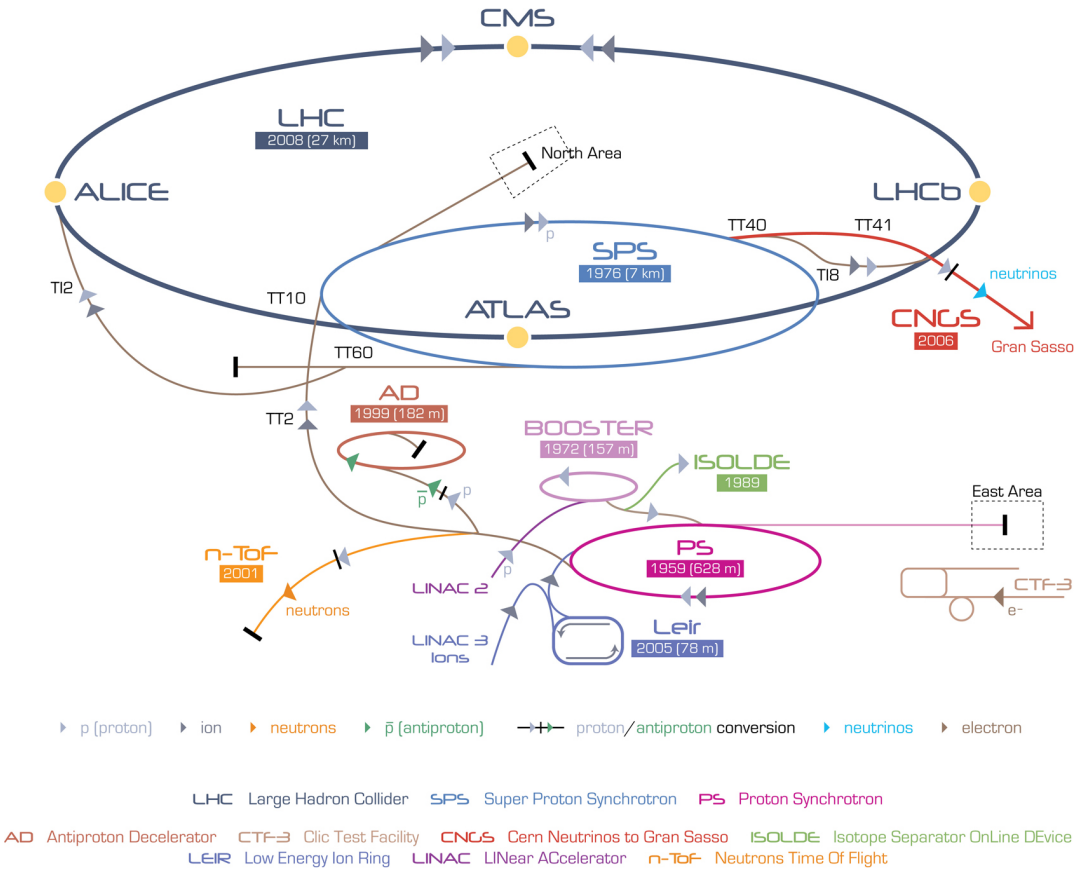


Figure 2.1: CERN Accelerator Complex [9]

The LHC can collide either protons or heavy ions, but for the purpose of this introduction, only the proton system is briefly discussed.

The proton beam originates from a small gas canister, containing pure hydrogen. Hydrogen atoms are supplied to a proton generator, which strips off the electrons in a high voltage electric field. The protons are accelerated to an energy of 50 MeV in Linac 2. Then, the beam enters the Proton Synchrotron Booster (PSB) where its energy is increased to 1.4 GeV. The PSB is made of four superimposed rings whose aggregate circumference is equal to the circumference of the next machine (Proton Synchrotron – PS). The beam is injected sequentially into separate rings and ejected into a single PS ring but with increased bunch luminosity. The PS output energy

reaches 32 GeV. The last machine in the injector chain, the 7 km-long Super Proton Synchrotron (SPS), provides two 450 GeV beams, which are fed in opposite directions to the LHC [8].

The 27 km long LHC tunnel is split into eight segments, with a length of 3.5 km each. Between the segments, there are eight underground caverns. Half of them host the collider equipment, namely:

- Point 4 contains the superconducting RF cavities. This is the only place in the LHC where the beams are accelerated.
- Point 6 is a beam dump station, where the beams can be safely ejected from the LHC and their energy dissipated in a graphite target.
- Points 3 and 7 host the beam collimators - a series of magnets forming the beam cross-section into the required shape.

The four remaining underground caverns are called "interaction points" because this is where the counter-running beams collide. There are six particle detectors within each interaction point:

- **ATLAS** (A Toroidal LHC Apparatus) and **CMS** (Compact Muon Solenoid) are general-purpose experiments, capable of detecting a wide range of particles produced in LHC collisions. The data from these detectors is used to search for new phenomena - the Higgs boson [10], supersymmetry [11] and extra dimensions [12].
- **ALICE** (A Large Ion Collider Experiment) is a specialised heavy ion detector, where heavy ions collisions at extreme energy densities will be analysed. It is expected that a new phase of matter, called the quark-gluon plasma, will form the existence and properties of which are key issues in quantum chromodynamics [13].

- **LHCb** (LHC Beauty Experiment) is used to study the asymmetry between matter and antimatter (called CP violation) present in iterations with B-particles (particles, which contain the bottom or "beauty" quark) [14].
- **TOTEM** and **LHCf** are two smaller detectors. TOTEM is dedicated to precise measurement of proton-proton interaction cross-section and the deep study of proton structure [15]. LHCf uses forward particles created in LHC collisions to simulate and study cosmic rays in laboratory conditions [16].

The systems described are able to inter-operate thanks to the sharing of a common and highly accurate notion of time. Providing this timing service in a reliable manner is the role of the timing systems.

2.2 Timing systems at LHC

The LHC beam production involves a long chain of injectors that need to be tightly synchronised. Moreover, the different types of beams produced for the LHC are only a subset of the beams that the injectors have to produce, for example beams for fixed target physics, for the On-Line Isotope Mass Separator Experiments (ISOLDE), for Antiproton Decelerator (AD). Typically, these beams are sequentially produced with intervals of a few tenths of a second. The composition of these sequences changes many times a day and the transition between different sequences has to be seamless.

To manage the settings of the accelerators and to synchronise precisely the beam transfer from one injector to the other up to the LHC, a Central Beam and Cycle Manager (CBCM) is used [17], [18]. The timing provided by the CBCM is carried by two dedicated networks:

- General Machine Timing (GMT)
- Beam Synchronous Timings (BST)

The CBCM elaborates the "telegrams" specific to each machine and broadcast over the GMT network. A "telegrams" describes the type of beam that has to be produced and provides detailed information for sequencing real-time tasks running in the Front-End Controllers (FEC) and for setting up the equipment [19].

The CBCM drives seven separate GMT networks dedicated to the different CERN accelerators, including the LHC and four BST networks of which three are for the LHC and one for the SPS. To achieve extreme reliability, a CBCM is running in parallel on two different machines, with a selector module capable of seamlessly switching between them.

2.2.1 General Machine Timing (GMT)

The GMT uses a dedicated network, called RS-422, to transmit a clock reference, send UTC timestamps and real-time control data. A GPS receiver provides two clock references, a pulse per second (PPS) and a 10 MHz clock signals. The GPS also provides an UTC time-stamp, via a serial link, that refers to the last PPS tick. The Master Timing Station multiplies the 10 MHz reference, via a PLL circuit [20], to produce a 40 MHz clock, which is used to produce a Manchester encoding stream at 500 kb/s. The master timing node encodes and transmits the UTC timestamps.

At the receiver side, the slave PLL recovers the stable 40 MHz reference from the 500 kb/s messages. The recovered clock is specified to have RMS jitter of less than 1 ns [21]. The 40 MHz signal is locked to UTC and can be used to generate pulses or to time-stamp external events with high accuracy.

The slave node receives an UTC time message every second from the master. It then starts a 40 MHz count to keep an internal UTC time register with a granularity of 25 ns. This register is used to time-stamping all the events produced in the node.

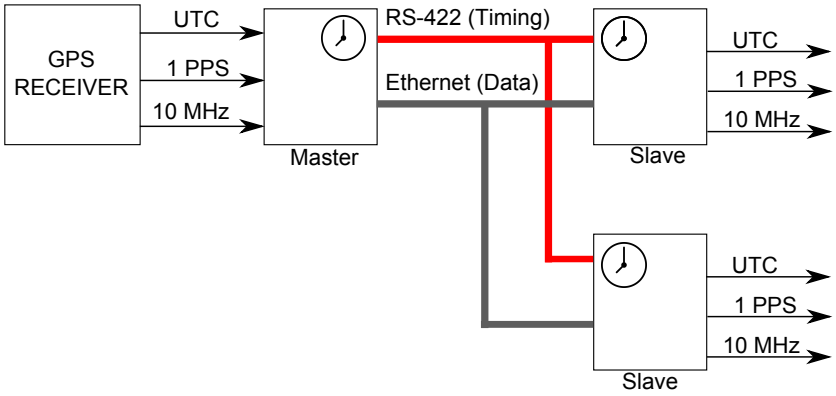


Figure 2.2: CERN's General Machine Timing System

2.2.2 Beam Synchronous Timing Network (BST)

The Beam Synchronous Timing (BST) is based on the CERN Timing Trigger and Control (TTC) network.

The BST cards are supplied with the RF bunch clock ~40 MHz and revolution clocks ~11.26 kHz for the LHC. The BST message is a byte stream sent each revolution, and it is Manchester encoded using the bunch clock. The BST messages carry triggers for beam instrumentation equipment, the UTC time, and parameters from the control telegram. The extremely high timing accuracy is possible because the physical transport layer uses TTC hardware.

2.2.3 Time Trigger and Control (TTC)

The TTC network is a point to multipoint optical link developed at CERN to allow Timing Trigger and Control communications between the accelerators and the detectors.

In this system, two command data channels are distinguished, channel A and channel B. Channel A is reserved for first-level trigger-accept (L1A) commands that last a single clock cycle and are transmitted with minimum latency. Channel B is used to send framed commands that can:

- be broadcast to all destinations

- individually address the internal registers of the receiving TTCrx
- send data to external electronics connected to an individually addressed TTCrx

Fully framed B-channel commands can take either 16 or 42 clock cycles to transmit and are used to initiate processes vital for the synchronisation of the different subsystems in the experiments. Both channels have a bandwidth of 40 Mb/s and have unidirectional exchange of messages.

The TTC develops a general strategy for synchronisation of front-end electronics and for the delivery of correctly phased clock, bunch-crossing identification and L1A signals to a large numbers of channels. The L1A trigger is generated by a set of algorithms during a given data time period and instructs the subsystems to send the recorded data to the Data Acquisition System for further analysis. The L1A is sent via channel "A" of the TTC system while BGo commands are sent via channel "B". Using a separated A-channel for the time-critical L1A signals avoids introducing extra latency due to the time needed for signal decoding into the trigger path.

In this timing distribution application, the extracted clock phase stability is of primary importance, while efficiency of channel utilisation for the transmission of the digital control information is a secondary consideration. The TTC network provides for distribution of clock trigger and fast control signals using two time division multiplexed channels transmitted over an optical passive distribution network.
The key features of the Timing, Trigger and Control (TTC) system are [22]:

- Clock distribution with individual adjustable phase adjustment per destination.
- Clock distribution with low jitter (30-50 ps RMS) at each destination.
- Provision of two independent channels for control commands.
- Clock and control data are Bi-Phase Mark (BPM) encoded and transmitted via an optical fibre distribution tree.

The key enabling parts that allow this system to achieve the required functionality are:

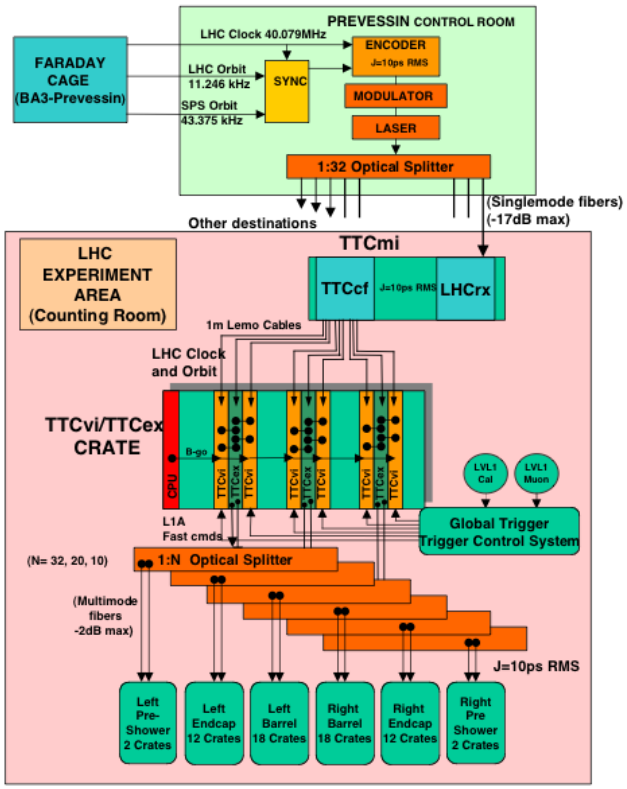


Figure 2.3: TTC Architecture [6]

- Low-jitter optical encoder and transmitter modules [6].
- Optical fibre distribution network based on optical couplers.
- Radiation-tolerant Receiver ASIC (TTCrx) [23].

Each experiment uses the TCC system to receive the timing information from the LHC machine (RF bunch clock and orbit signal).

The TTC system is unidirectional. This means that status information vital to the smooth running of the experiments data-taking cannot be collected by this system. Some experiments have developed systems, like the Trigger Throttling System (TTS) [22], to allow real-time response to synchronisation-loss within the system.

2.2.4 Current Research Work for TTC upgrade

The PON-TTC aims to be the future of the TTC system. The existing TTC links and some of its key components, such as the TCC laser transmitter, need to be upgraded. The PON-TTC project aims to simplify the network and to enhance the system functionality as well as make the links bidirectional in such a way that the uplink responses could be done using the same transmission medium infrastructure. This upgraded system exploits the Field Programmable Gate Arrays (FPGA) technology.

The published results on this ongoing work [24] shows a PON-TTC system with following specifications:

1. The downstream link is a low and fixed latency, gigabit link that is capable of carrying the L1A and commands.
2. The 40 MHz LHC bunch clock is recovered from the serial data of the receiver and the deskewing process based on the FPGA digital clock management (DCM) module.
3. The recovered clock can be used to drive a high speed serialiser for further distribution of data to the detector front-end components.
4. Communication is bidirectional. The upstream link is used to deliver feedback information and to monitor the feeder fibre latency.

This ongoing work highlights the following areas of improvement:

- Downstream Latency - The work shows a downstream latency of 216.8 ns that needs to be further reduced. The authors mention that this might be improved by increasing the serial data rate.
- Full Ranging
- Non-blocking upstream Architecture
- Clock Frequency Agnostic TTC

A PON-TTC bidirectional system has been demonstrated by using passive optical networks optical transceivers and FPGA. It satisfies the timing mandate of the TTC

network, namely fixed and deterministic downlink latency and distribution of a reference clock with low jitter with latency monitory capability.

2.3 Timing Control for Radio-astronomy

This section presents a unique timing distribution system implemented for the Atacama Large Millimetre Array (ALMA) where the distribution of the LO reference must be accurate within 38 fs. ALMA is an international radio astronomical facility in Chile that results from international collaboration. The facility consists of an array up to 64 parabolic antennas, ranging in diameter from seven to 12 metres, that can detect millimetre and sub-millimetre wavelength radio wave in the frequency band between 31-950 GHz. At these wavelengths, the radiotelescope array will be able to reveal the structure of the cold regions of the universe, otherwise dark at visible wavelength, with an unprecedented sensitivity and a resolution of 10 milliarcseconds[†] [25]. This resolution is an order of magnitude higher than the Hubble telescope or the very large array operating in New Mexico. ALMA achieves its exceptional resolution and sensitivity by linking all of the 64 antennas into a interferometer array [25].

To accurately measure the phase of the sky signal over entire array or subarray, every antenna must receive a highly stable common reference signal known as the local oscillator (LO) reference. For ALMA, this LO reference signal is composed by two optical waves sent throughout a single optical fibre [26]. Both waves have a wavelength around 1.556 μm to allow transmission trough conventional telecommunication optical fibre with little loss. One optical wave is generated by a Maser Laser (ML) [27], while the other is generated by phase-locking a slave laser at a given frequency offset from the master. Both lasers are located in a central building. The frequency offset between the two lasers ranges from 27-142 GHz.

Each antenna sends the photonic LO reference signal to a photodetector, which acts similarly to a conventional RF mixer but at optical frequencies. Therefore, the output

[†]A unit of angle equal to one thousandth of an arcsecond(1/3600th of a degree)

of the photodetector contains an RF beat note signal at the frequency between the ML and slave lasers, that is, the original 27-142 GHz offset frequency, as illustrated in Figure 2.4. This LO reference signal is used to phase-lock the antenna oscillators whose outputs are converted to 27-938 GHz LO signals using cryogenically cooled frequency multipliers [26]. The LO signal is then used to down-convert the sky signals to an intermediate frequency (IF) band 4-12 GHz. The IF signal is digitised and sent back to the correlators in the central building for processing to extract the phase information and ultimately generate astronomical images.

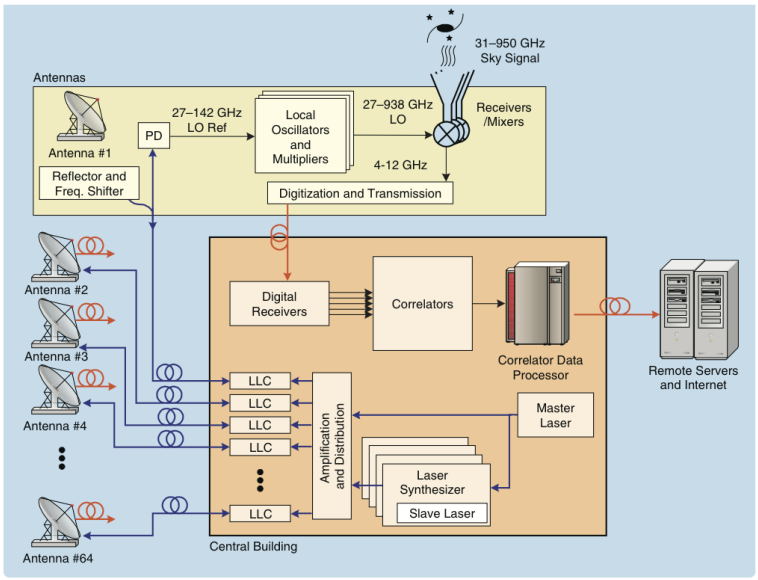


Figure 2.4: ALMA Timing System Architecture [25]

One critical requirement for ALMA is that the timing of the LO reference signal arriving at any antenna must be stable within 38 fs [26]. Larger timing fluctuations would degrade the accuracy with which the signal source can be pinpointed and would consequently result in a blurred image, reducing the scientific value of the data. Maintaining a propagation delay within the specification implies that the length of the fibre used to transmit the LO reference signal must be stabilised to within a few micrometers, even thought the path length may be as great as 18 km. Several dynamic techniques have been created to improve fibre stability, which can be compromised by environment factors [26].

There are three critical subsystems that allow the optical distribution of a low-noise LO reference signal to the antennas with the required stability. These subsystems are the laser synthesiser (LS), the line length corrector (LLC), and the Master Laser (ML). The LS phase-locks a slave laser optical signal to an ML to generate an extremely pure beat note used as a reference signal, which can be distributed to the antennas through optical fibres. The LCC system controls the length of the fibres to ensure that the propagation delay of the references signals to the antennas is constant. Finally, the ML stabilises the optical frequency using an atomic gas reference to provide an accurate reference to perform the line length correction.

2.4 Global Positioning System (GPS)

The basis of GPS-based navigation is triangulation from satellites. The GPS system uses a set of satellites located in space as reference points for locations here on Earth. To triangulate, a GPS receiver measures distance using the travel time of radio signals from the satellite to the device. To measure travel time, GPS needs very accurate timing that it achieves with special calibration schemes [28]. Indeed, the development of ultra-stable clocks and stable space platforms in predictable orbits are key technologies that made GPS possible. Along with distance, this system requires the exact spacial coordinates of the satellites. High orbits and careful monitoring are the special characteristics that make GPS feasible. Finally, corrections are made for any delays the signal experiences as it travels down through the atmosphere.

The architecture of the GPS system is similar to other satellite systems. There is a space segment, a ground segment (for control) and the user segment (terminals). The baseline GPS satellite constellation is made up of 24 satellites in nearly circular 26560 km orbits with an orbital period of nearly 12 hours. This segment is in six orbital planes of four satellites each. The ground segment provides satellite management to monitor and control GPS on-orbit operations.

Each GPS satellite transmits simultaneously on two L band frequencies [†], called L1 and L2, 1575.42 MHz and 1227.6 MHz, respectively. Each L1 signal is modulated by a pseudo-random noise (PRN) sequence called coarse acquisition code. Each coarse acquisition code is 1023 Gold code with a chip rate of 1023 Mb/s, i.e the sequence repeats every millisecond. The L2 signal is encrypted and primarily intended for use by the Department of Defence. Each signal, like any CDMA signal, consists of 3 parts, the carrier, the PRN spread spectrum code and data message. PRN codes are chosen for their favourable autocorrelation and cross correlation properties. For coarse acquisition code, for example, the autocorrelation function is 24 dB lower for all shifts greater than one chip width.

GPS signals received on Earth are extremely weak. The specification for the minimum power level received for the users on Earth is -160 dBW. Such a weak signal is clearly vulnerable to noise and jamming and this is one of the most important GPS user concerns. Performance perceived by GPS users is dynamic and changes with time and place depending on the satellite geometry and measurement errors.

Safety-critical applications such as civil aviation require exceptional navigation accuracy. Currently, GPS systems do not have the ability to detect anomalies in their signals and convey it to the user. Thus, system and data integrity is largely the responsibility of the user. The FAA and ESA have been developing systems (WAAS and EGNOS) that will overcome these deficiencies. On May 1, 2000 the US. deactivated the civilian GPS feature, known as selective availability (SA), which intentionally skewed position and timing information for civilian use. This improves accuracy from about 100 to better than 20 meters for the general public user.

2.5 Summary

This chapter provided an overview of CERN's accelerator chain including LHC and smaller accelerators. Additionally, a brief description of the several LHC's detectors

[†]L band refers to a band of the electromagnetic spectrum, 1 to 2 GHz (IEEE)

is given. The accelerator chain needs to be tightly synchronise to allow the transfer of beams among the various accelerators. This is accomplished with to a sophisticated, however, outdated timing system architecture. The timing systems at CERN can be divided in two main groups, one dedicate to the accelerator synchronisation and another for the detector synchronisation. Both of these timing systems use dedicated network links to distribute timing and control messages around the complex.

Section 2.3 describes one of the most accurate timing system in the world which is currently being deployed in Chile for radio-astronomy research. This timing system designed for ALMA has very strict timing specification in the femtosecond range. The timing is distributed over a dedicated optical links in a new and innovative custom-made design. Next, the GPS system, the most widely known synchronisation system, is presented. GPS provides timing information using wireless communications between space satellites and earth stations.

In conclusion, this chapter described different timing systems for applications that require a very accurate timing distribution. However, these systems have a high deployment cost due to the fact that they require dedicated links to distribute timing. In addition, regular calibration procedures are required to maintain the link delay variation between the nodes compensated. This procedure is realised during the year by a timing operator that carries a high stable atomic clock from the master node to the slave nodes with the intent of updating their timing with one from the master clock.

The White Rabbit project, described in Chapter 3, aims to overcome all the above mentioned problems by using a standardised network protocol, the Ethernet, to multiplex both the timing and data information in the link. The White-Rabbit and the PON-TTC are technologies with very similar requirements, however, the White-Rabbit aims to be used by the accelerators' complex and is locked to a very timing stable source deriving from UTC. PON-TTC on the other hand is to be used by the experiments around the accelerators and is locked to the LHC's Beam RF signal,

which has frequency modulation due to the Beam acceleration. The Distribution RF signal discussed later in this thesis aims to provide the White Rabbit network as an alternative to the PON-TTC.

The link delay variations due to external factors are measured and compensated using a technique described and implemented in Chapters 5 and 6, respectively.

Chapter 3

Synchronisation in Ethernet

Ethernet is today the backbone in network connectivity for both home and office environments. It is the most popular solution for networking services in enterprises. The growing popularity of Ethernet as a connectivity medium for networks has to do with the fact that Ethernet networking equipment is easy to install, administrate and maintain at a very competitive cost.

Ethernet is also becoming increasingly popular for factory automation, industrial and other networking applications, despite the fact that the original protocol is inherently non-deterministic. However, a number of manufacturers have presented different proposals that aim to bring this computer network to the real time communication requirements of a factory floor [29].

The use of full duplex Ethernet and switches constitutes a class of solutions that support all types of Ethernet nodes and still offer guarantees as long as all the relevant protocols are fully used. The use of switches to offer real-time guarantees in factory communications has been suggested and analysed by a number of authors [30] [31] [32] [33]. The main idea is to build a network with regular Ethernet nodes, switches, and full duplex mode. This means that each node is hooked to a single port of a switch. This architecture suppresses all collisions, however, the time delivery of the message is not deterministic unless traffic shaping [34], [35] or message scheme

priority, such as IEEE 802.1Q, is used [29].

Another point of importance when discussing the use of Ethernet in an industrial environment is how is the issue of synchronisation between the network nodes addressed. This issue is the focus of this chapter.

In Chapter 3, the methods used to synchronise Ethernet links are discussed. The chapter begins with a description of the several layers responsible for synchronising an Ethernet node to the incoming encoded data symbols. Section 3.2 provides a detailed description of the Synchronous Ethernet standard. The Synchronous Ethernet proposed by ITU-T aims to achieve levels of synchronisation equivalent to SONET/SDH networks by using the Ethernet physical layer. Section 3.3 introduces the IEEE 1588 protocol. IEEE 1588 specifies a mechanism to broadcast a set of timing messages over a packet-based-network to distribute timing and to synchronise the network clocks. The White Rabbit (WR) project is detailed in Section 3.4. The WR project is a project managed by CERN in cooperation with several academic and industrial partners with the goal of distributing timing in an Ethernet network with sub-nanosecond accuracy, the so-called White Rabbit network. In addition, the WR network aims to deliver control messages with a bounded latency. The ultimate goal is to make the WR network a standardised Ethernet timing network for High Energy Physics and other applications where both highly accurate timing and control data are required. In Section 3.4.1, a link delay model of the White-Rabbit network is proposed. In addition, based on the delay link model, a measuring methodology is proposed to provide White Rabbit with the sub-nanosecond timing accuracy.

3.1 The IEEE 802.3

Ethernet is one of the most popular networking technologies used in today's local area networks. Its first official standard was published in 1983 by the IEEE 802.3 committee. The most recent review was published in 2011 and it defines operational speeds from 10 Mbps to 10 Gbps [36]. Throughout this thesis, 1Gb/s physical interface is

primarily described.

Gigabit Ethernet allows connection between two devices in either a full, or a half-duplex mode. In half-duplex mode, Gigabit Ethernet uses the Carrier Sense Multiple Access with Collision Detection (CSMA/CD) access method, just like its 10 and 100 Mbps predecessors. When communication between two devices is done in full-duplex mode there is no need for CSMA/CD. The majority of modern-day networking scenarios are generally full duplex, however, support for half duplex communications does exist to meet the requirements of the 802.3 standard and to support legacy devices.

The Physical layer specification for 1 Gbps speeds is defined by clause 36 through 39 of the IEEE 802.3 standard [36]. Figure 3.1 illustrates the relationship between the OSI Model [37] and the IEEE 802.3 standard [36].

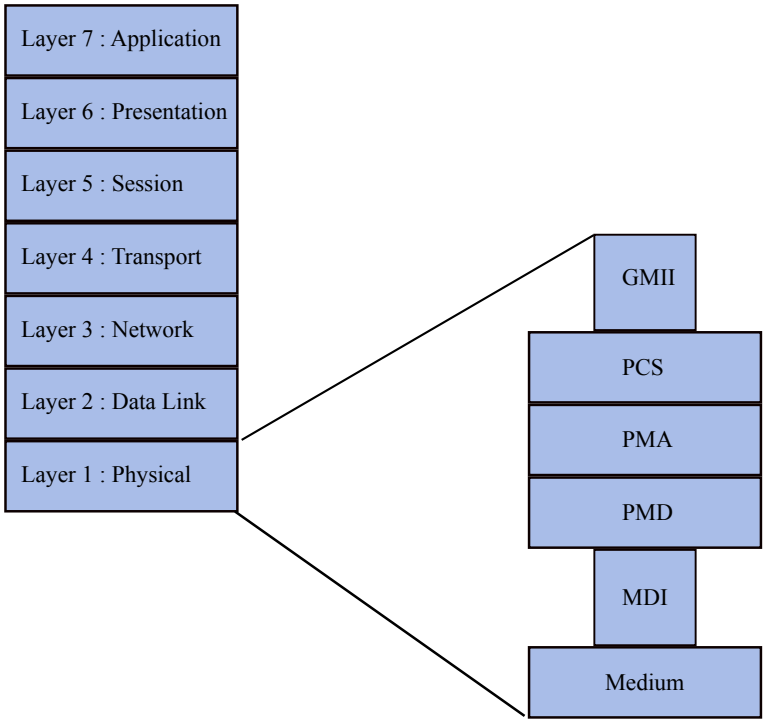


Figure 3.1: The relationship between the OSI reference model and the Ethernet model

The physical layer of Ethernet is implemented with a dedicated physical layer, which is connected to the MAC sub-layer via Gigabit Medium Independent Inter-

face (GMII). The interconnection between the GMII and the next layer is via a 125 MHz bus with an 8-bit data wide data path, plus some control bits, as shown by the function diagram in Figure 3.2.

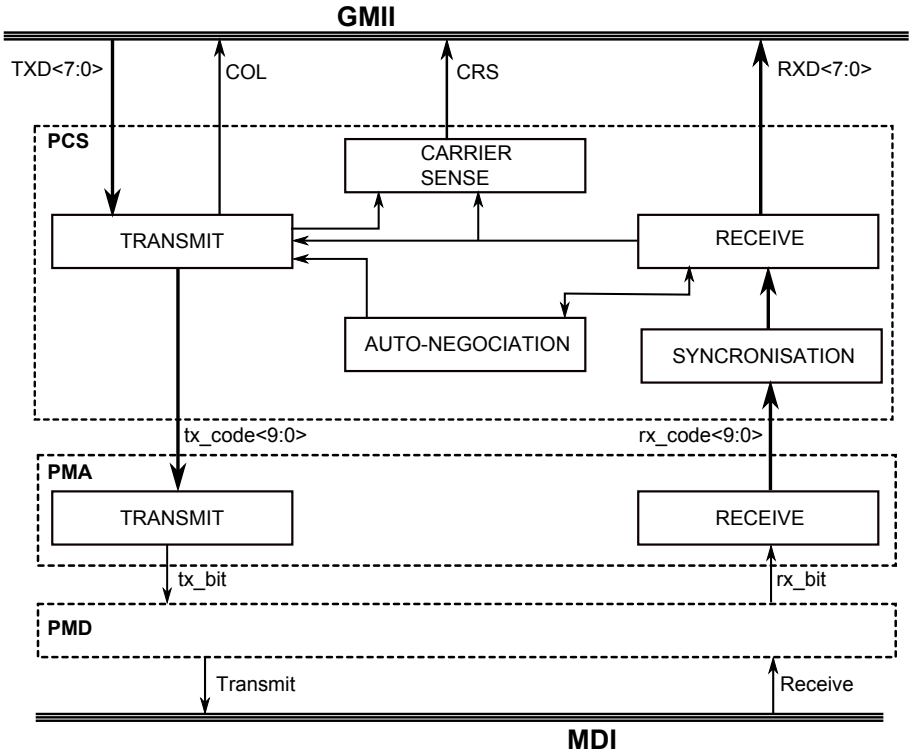


Figure 3.2: Ethernet Function Diagram [36]

The GMII is designed to make the various media transparent to the MAC sub-layer. The Physical Coding Sub-Layer (PCS) and the Physical Medium Attachment (PMA) are both contained within the physical layer of the OSI reference model. The PCS and the Gigabit Media Independent Interface (GMII) communicate with one another via 8-bit parallel data lines and several control lines and perform the encoding (and decoding) of the 8B/10B [38] transmission coding. In addition, the PCS manages the synchronisation and the auto-negotiation processes.

The PMA performs the 10-bit serializing functions, it receives 10-bit encoded data at 125 MHz from the PCS and delivers serialised data to the Physical Medium Dependent (PMD) sub-layer. In the reverse direction, the PMA receives serialised data from the PMD and delivers 10-bit width data to the PCS.

3.1.1 Physical Coding Sub-layer (PCS)

The PCS is responsible for encoding each octet transmitted from the GMII into ten-bit code groups. The PCS is also responsible for decoding ten-bit code groups received from the PMA into octets for use by the upper layers. The PCS controls the auto-negotiation process that chooses the transmission capabilities, such as link rate, in which two separate Ethernet devices are capable of exchanging data with maximum efficiency.

The PCS examines each incoming octet passed down by the GMII and encodes it into a ten bit code group. This is referred to as 8B/10B encoding [38]. Each octet is given a code group name according to the bit arrangement.

Each data code group is divided into two groups and named /D$x.y$/, where the value of (x) represents the decimal value of the five least significant bits and (y) represents the value of the three most significant bits. For example:

/D6.2/ = 010 00110

/D30.6/= 110 11101

There are also 12 special octets that are also encoded into ten bits. The PCS differentiates between special and data code words via a signal transmitted from the GMII.

Special code words follow the same naming convention as data code words except they are named /Kx.y/ rather than /Dx.y/.

One of the motivations behind the use of 8B/10B encoding lies in the the ability to control the characteristics of the code words such as the number of ones and zeros and the consecutive number of ones or zeros. Another motivation behind the use of this encoding scheme is the ability to use special code words for which link control and synchronisation would be impossible if no encoding was performed [38].

A special sequence of seven bits, called a "comma" [†], is used by the PMA in the

[†]the code group containing a comma sequence is represented as /COMMA/

alignment of the incoming serial stream. /COMMA/ Comma is a unique data pattern that is contained within only the /K28.1/, /K28.5/ and /K28.7/ special code groups. The comma is also used by the PCS in acquiring and maintaining synchronisation.

DC balancing is achieved through the use of a running disparity calculation. Running disparity is designed to keep the number of ones transmitted by a station equal to the number of zeros transmitted by that station. This should keep the DC level balanced halfway between the "one" voltage level and the "zero" voltage level.

Ordered-Sets

Eight ordered-sets, consisting of a single special code-group or combinations of special and data code-groups are specifically defined. Ordered-sets which include /K28.5/ provide the ability to obtain bit and code-group synchronisation and establish ordered-set alignment.

Ordered-sets provide for the delineation of a packet and synchronisation between the transmitter and receiver circuits at opposite ends of a link.

The notation used for ordered-sets is similar to that used for code-groups. Code-groups are written as either /D$x.y$/ or /K$x.y$/. Ordered-sets are written in the form of /XY/ where X is a letter and Y is sometimes used and contains a number. The defined ordered-sets are: /C/, /C1/, /C2/, /I/, /I1/, /I2/, /R/, /S/, /T/ and /V/.

/K28.5/ is used as the first code-group because it contains a comma, which is a unique data pattern that was defined previously. The reception of this code-group will not happen during a data packet unless there is a data error. This makes it very useful for use with very specific ordered-sets such as Idle and Configuration.

PCS Functions

The PCS sub-layer can be broken down into four major functions:

- Synchronisation Process

Table 3.1: Defined Ordered-Sets

Code	Ordered-Set	Number of Code-Group	Encoding
/C/	Configuration		
/C1/	Configuration 1	4	/K28.5/D21.5/config_reg[7:0]/config_reg[15:8]/
/C2/	Configuration 2	4	/K28.5/D2.2/config_reg[7:0]/config_reg[15:8]/
/I/	IDLE		
/I1/	IDLE 1	2	/K28.5/D5.6/
/I1/	IDLE 1	2	/K28.5/D16.2/
/R/	Carrier_Extend	1	/K23.7/
/S/	Start_of_Packet	1	/K27.7/
/T/	End_of_Packet	1	/K29.7/
/V/	Error_Propagation	1	/K30.7/

- Transmit Process
- Receive Process
- Auto-negotiation Process

The auto-negotiation process is performed during the initial setup of the link by exchanging a orderer set of configuration (/C/) code-group data. Auto-negotiation can be instructed to restart if /C/ orderer set data are received from the other station after the link has been established.

The purpose of the PCS synchronisation process is to provide the PMA with the data properly aligned from the received serial stream.

Synchronisation is acquired upon the reception of three ordered-sets each starting with a code-group containing a /COMMA/.

Each comma must be followed by an odd number of valid data code-groups. No invalid code-groups can be received during the reception of these three ordered-sets.

Once synchronisation is acquired, the synchronisation process begins counting the number of invalid code-groups received. That count is incremented for every code-group received that is invalid or contains a comma in an odd code-group position. That count is decremented for every four consecutive valid code-groups received (a comma received in an even code-group position is considered valid). The count never goes below zero and if it reaches four, *sync_status* is set to FAIL.

Startup Protocol

The PCS start-up protocol for auto-negotiating devices can be divided into two components:

- Synchronisation Process
- Auto-Negotiation Process

Auto-Negotiating and manually configured devices are unable to interpret any received code-group until synchronisation has been acquired. Once synchronisation has been acquired, the PCS is then able to receive and interpret the incoming code-groups.

Acquiring Synchronisation

After powering on or resetting, the PCS synchronisation process does not have synchronisation and it is in the LOS[†] state. The synchronisation process looks for the reception of a /COMMA/. It then assigns that code-group to an evenly aligned code-group. The next code-group received is assigned to an odd code-group. Code-groups received thereafter are alternately assigned to even and odd code-group alignments.

The number of valid code-groups following the /COMMA/ does not have an upper limit. The synchronisation process moves to the SA1[†] state and asserts the flag

[†]LOSS_OF_SYNC
[†]SYNC_ACQUIRED_1

sync_status when synchronisation is acquired. If at any time prior to acquiring synchronisation the PCS receives a /COMMA/ in an odd code-group or if it receives an /INVALID/, a code-group that is not found in the correct running disparity tables, the PCS synchronisation process returns to the LOS state.

The following section defines how the PCS sub-layer can lose synchronisation once it has been acquired.

Maintaining and Losing Synchronisation

While in the SA1 state, the PCS synchronisation process examines each new code-group.

If the code-group is a valid data code-group or contains a /COMMA/ when rx_even is FALSE, the PCS asserts the variable cggood and the synchronisation process toggles the rx_even variable. Otherwise, the PCS asserts the variable *cgs_bad* and the process moves to the SA2 state, toggles the rx_even variable, and sets the variable *good_cgs* to 0.

If the next code-group is a valid code-group that causes the PCS to assert the variable cggood, the process transitions to the SA2A state, toggles the rx_even variable, and increments *good_cgs*. Otherwise it continues on to the SA3 state.

While in the SA2A state, the process examines each new code-group. For each code-group that causes the PCS to assert cggood, the variable *good_cgs* is incremented. If *good_cgs* reaches three and if the next code-group received asserts cggood, the process returns to the SA1 state. Otherwise, the process transitions to the SA3 state.

Once in the SA3 state, the process may return to the SA2 state via the SA3A state using the same mechanisms that take the process from the SA2 state to the SA1 state. However, another invalid code-group or comma received when rx_even is TRUE will take the process to the SA4 state.

If the process fails to return to the SA3 state via the SA4A state, it transitions to LOS where *sync_status* is set to FAIL.

Thus, once *sync_status* is set to OK, the synchronisation process begins counting the number of invalid code-groups received. That count is incremented for every code-group received that is invalid or contains a comma when rx_even is TRUE. That count is decremented for every four consecutive valid code-groups received (a comma received when rx_even is FALSE is considered valid). The count never goes below zero and if it reaches four, *sync_status* is set to FAIL.

PCS Transmission Process

The PCS transmit process is responsible for the 8B/10B encoding of the data octets from the GMII.

The PCS transmit process sends the signal "transmitting" to the Carrier Sense function of the PCS whenever the PCS transmit process is sending out data packets. The signal "receiving" is sent out by the PCS receive process whenever it is receiving packets. The PCS transmit process is responsible for checking to see if the PCS is both sending and receiving data. If (receiving = 1 AND transmitting = 1) then the collision signal COL is sent to the GMII.

The Carrier Sense mechanism of the PCS reports Carrier events to the GMII via the CRS signal. CRS is asserted when (receiving = 1 OR transmitting =1) and is de-asserted when (transmitting= 0 AND receiving = 0)[†].

PCS Receive Process

The PCS receive process is responsible for decoding the incoming 10-bit code-groups into their corresponding octets for transmission by the GMII. The receive process sends out the signal "receiving" to the PCS transmit process and the Carrier Sense process for use in notifying the upper layers that there is carrier on the line or that

[†]CRS is asserted for repeaters when receiving=TRUE and de-asserted when receiving=FALSE

a collision has occurred.

Carrier Detection is used by the MAC for deferral purposes and by the PCS transmit process for collision detection. A carrier event is signalled by the assertion of "receiving".

Figure 3.3 illustrates the PCS layer during an Ethernet packet reception.

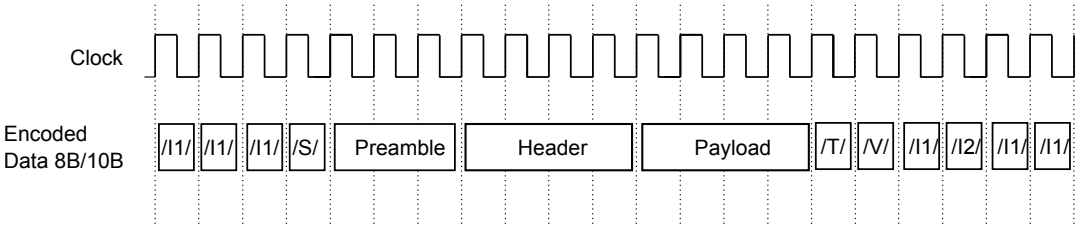


Figure 3.3: PCS Symbols During Link Activity

3.1.2 Physical Medium Attachment (PMA)

The PMA sub-layer is responsible for aligning the incoming serial stream of data. The PMA receive process is allowed to lose up to four 10-bit code-groups in the alignment process. This is referred to as code slippage. The PMA receive process aligns the incoming code-groups by finding /COMMA/s in the data stream and aligning the subsequent code-groups according to the /COMMA/. Alignment by the PMA is essential if the PCS receive process is to operate correctly. If misaligned /COMMA/s and/or data are constantly sent by the PMA, synchronisation cannot be achieved by the PCS.

The PMA is responsible for the serialisation of the incoming 10-bit code-groups from the PCS. This is accomplished through the use of a ten times clock multiplier. The PCS transmits the code-groups in parallel at 125 MHz and the PMA sends them out bitwise at a rate of 1.25 GHz.

The PMA is also responsible for the deserialisation of the data coming in from the PMD. The data arrives serially at a rate of 1.25 GHz and is transmitted in parallel to the PCS at a rate of 125 MHz. The PMA is also responsible for recovering the

clock from the incoming data stream.

3.1.3 The Physical Layer (PHY)

The standards for Gigabit Ethernet include operation over a variety of physical interconnection media including:

- 62.5 μm and 50 μm multimode fibre (MMF)
- Single mode fibre (SMF)
- Short copper links (25 meters)
- Horizontal copper (100 meters)

A variety of PHY types have been specified in the Gigabit Ethernet draft standards. This includes support for a broad range of distances including various options optimised for important cost/distance design points. PHYs incorporated into the IEEE 802.3z draft standard include 1000BASE-CX, which supports interconnection of equipment clusters; 1000BASE-SX, which is targeted for horizontal building cabling; and 1000BASE-LX, which supports backbone building, cabling and campus interconnections.

3.2 Synchronous Ethernet

The Ethernet, defined by the IEEE 802.3 standard [36], does not transport synchronisation information that can be referred to an external source, in other words Ethernet is a non-synchronous network.

Synchronous Ethernet (Sync-E) provides a mechanism for transferring frequency over the Ethernet physical layer, which can be traceable to an external source such as a network clock. As such, the Ethernet link may be used and considered part of the

synchronisation network. This solution uses the physical layer, which has been shown to be immune to traffic load and packet delay variation [39].

ITU-T Recommendation G.8261 [40], released in 2006, was the first document to introduce the Synchronous Ethernet concept as part of the studies related to the synchronisation aspects in packet networks. This technology has been standardised by the ITU-T as a direct result of the experience gained with the standardisation of timing distribution over SONET/SDH networks. Since the very beginning, G.8261 was recognised as the main reference for this technology. Although, the first version of the standard presented several aspects not fully defined, it aims at describing different methods to fulfil the synchronisation related requirements.

The primary idea behind Sync-E is that the input reference clock is not free-running with an accuracy of ±100 ppm, but rather locked and traceable to a primary reference clock as defined in ITU G.811 [41] and thus achieving long-term accuracy of ±10 ppt [39].

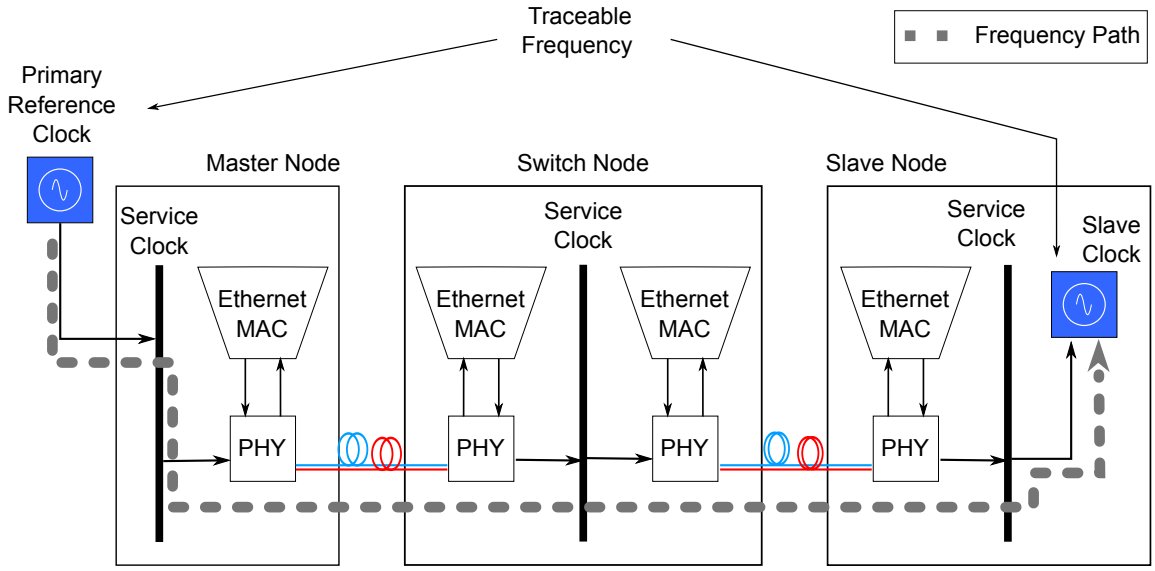


Figure 3.4: Synchronous Ethernet Frequency Reference Distribution Model

A typical Sync-E line card includes the Ethernet MAC and PHY transceiver. Frequency conversion adapts backplane frequencies to frequencies that are required as input reference clocks to the transceiver, usually 125 MHz or a multiple of this fre-

quency. The output of the timing device serves as a transmission clock reference to the transceiver, replacing the free-running crystals with ± 100 ppm accuracy in the conventional line card. Sync-E clocks are within ± 4.6 ppm accuracy, which is more than that required by the frequency range of Ethernet's specification. The clock reference is used to perform the physical line encoding, using 8B/10B for gigabit Ethernet, of the data transmission process towards the slave, as shown in Figure 3.4.

At the slave node, the received data is decoded and the network clock is recovered from the Ethernet's transceiver clock and data recovery (CDR) circuitry. From this point on, the slave station behaves like the master for the downlink Ethernet nodes. The frequency synchronisation is transported on a node-to-node basis, where each node participates in recovery and distribution.

It is with this process that synchronisation traceable to a primary reference source can be recovered at each node and distributed to the remaining downlink nodes. It is also important to note that Synchronous Ethernet does not disrupt the IEEE 802.3 architecture. The deployment of synchronisation networks using the Physical Layer provides a useful tool to distribute frequency that is not subject to packet delay variation [42].

3.3 IEEE 1588 - Precise Time Protocol

The IEEE 1588 standard[†] [43], first released in 2008, is a packet-based protocol designed to synchronise devices in distributed networks.

The standard defines two types of messages that are exchanged between PTP nodes: event messages and general messages. Both the time of transmission and the time of reception of event messages are timestamped. General messages are used by PTP nodes to identify other PTP nodes, establish clock hierarchy and exchange data, for example timestamps, settings or parameters. PTP defines several methods for node synchronisation. Figure 3.5 presents the message types exchanged between timing

[†]also known as Precise Time Protocol (PTP)

nodes when the delay request-response mechanism (with a two-step clock) is used [43].

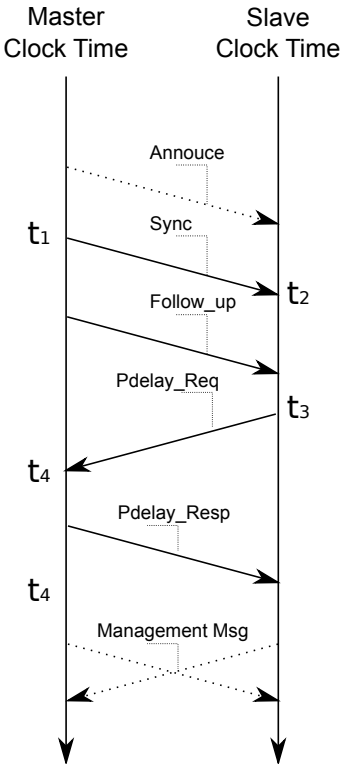


Figure 3.5: PTP Message Exchange

An Announce Message is broadcast periodically by the PTP node that is in the Master state. The message carries information about its originator and the originator's clock source quality. This enables other PTP nodes receiving the announce message to perform the Best Master Clock (BMC) Algorithm. This algorithm defines the role of each PTP node in the PTP network hierarchy; the outcome of the algorithm is the recommended next state of the PTP node and the node's synchronisation source (grandmaster). In other words, a PTP node decides to which other PTP node it should synchronise based on the information provided in the announce messages and using the BMC algorithm. A PTP node that is in the SLAVE state synchronises to the clock of another PTP node. A PTP node that is in the MASTER state is regarded as a source of synchronisation for the other PTP nodes.

Sync Messages and Delay Req Messages are timestamped at t1, t2, t3, t4, as shown

in Figure 3.5. The timestamps are used to calculate the offset and the delay between the nodes exchanging PTP messages. *Follow Up* and *Delay Resp* messages are used to send timestamps between Master and Slave (in the case of a two-step clock).

Management messages are used only for configuration and administrative purposes. They are not essential for PTP synchronisation. The flow of events in the PTP delay request-response (two-step clock) mechanism is the following (simplified overview):

1. The master sends Announce messages periodically.
2. The slave receives the Announce message and uses the BMC algorithm to establish its place in the network hierarchy.
3. The master periodically sends a Sync message (timestamped on transmission, t1) followed by a Follow UP message which carries t1.
4. The slave receives the Sync message sent by the master (timestamped on reception, t2).
5. The slave receives the Follow Up message sent by the master.
6. The slave sends a Delay Req message (timestamped on transmission, t3).
7. The master receives the Delay Req message sent by the master (timestamped on reception, t4).
8. The master sends the Delay Resp message which carries t4.
9. The slave receives the Delay Resp.
10. The slave adjusts its clock using the offset and the delay calculated with timestamps (t1, t2, t3, t4). It results in the Slave's synchronisation with the Master clock.
11. Repeat 1-10.

IEEE 1588 is a protocol capable of delivering timing with sub-microsecond accuracy. The sub-microsecond timing accuracy requires hardware-based time stamping, due to the unpredictable latency introduced by the upper network layers. However, even with hardware-based time stamping, IEEE 1588 is not able to achieve sub-nanosecond accuracy [44]. The reasons for this limitation are diverse and are mainly due to the packet-based nature of the standard. For example, IEEE 1588 is not able to measure effectively small delay changes (in the order of picoseconds) in the transmission link. Moreover, the granularity of the hardware-time stamping mechanism does not allow

IEEE 1588 to measure the arrival and departure of an IEEE 1588 packet with the required time resolution.

The White Rabbit project, discussed in the next section, aims to overcome the above issues providing a technology combining IEEE 1588 and Sync-E to achieve sub-nanosecond timing accuracy.

3.4 The White Rabbit Project

Accurate sampling of the accelerator events from several thousands devices requires a high performance clock and a control distribution system [22].

The White Rabbit (WR) project began as a potential successor for several legacy control and timing systems, such as the GMT, currently in operation at CERN for the LHC. However, due to its high timing performance requirements, several High Physics Institutes such as the GIS in Darmstadt, Germany, and other industrial companies, like National Instruments, are considering the use of the White Rabbit Timing system in their control systems where high accurate timing is required.

The White Rabbit network aims to combine the accuracy of dedicated timing systems with the data transmission performance, flexibility and scalability of Ethernet.

The WR network main goals are to provide:

- A deterministic, large-scale monitoring and control network based on Gigabit Ethernet, supporting approximately 1000 nodes spread over tens of kilometres, synchronised with a sub-nanosecond accuracy.
- A scalable and modular platform that does not require specialised configuration and maintenance.
- A robust mechanism that delivers critical messages with bound latency.
- A fully open design, not tied to any particular hardware or software vendor.

A simple and common case scenario that describes the problem of clocks distributed synchronisation, is illustrated in Figure 3.6. In Figure 3.6, an operator manages two

clocks in order to maintain them synchronised, in other words they need to show the same time information.

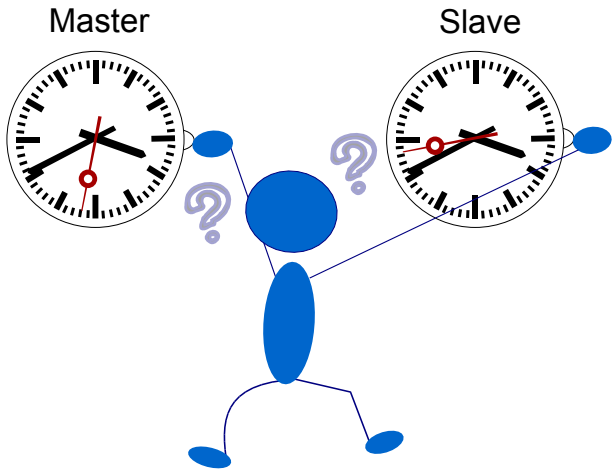


Figure 3.6: Timing Synchronisation

One of the clocks is the master clock, and is used as a time reference. The other clock is a slave clock that needs to be synchronised to the master clock reference.

In order to achieve synchronisation, both clocks must have the same frequency rate. The process of aligning the frequency rates of the two clocks is called syntonisation. Several methods exist to achieve syntonisation: one is to define a frequency rate by means of a local oscillator specification for both clocks. Nevertheless, although the local oscillators match the frequency specification, their frequency rate contains phase noise or jitter. As time is calculated by integrating the oscillator rate, the time of each clock starts to drift between each other due to the timing jitter in both oscillators.

Another process for syntonisation consists of transmitting the frequency reference from the master clock directly to the slave clock via a direct link.

Next in the synchronisation process, it is required to align the frequency rate of the slave clock to the temporal information of the master clock, so that both clocks display the same time. In the scenario illustrated in Figure 3.6, the operator calculates the time difference between the master clock and the slave clock. Next, the operator updates the slave's clock time accordingly with the measured time difference so that

the slave clock is aligned with the master clock. In addition, the operator needs to compensate for the delay between the reading of the master's time information and the correction in the slave's clock. A series of time corrections are done at the slave clock so that the time difference error between the two clocks is gradually reduced.

If the two clocks are tightly tuned up and their frequency rate is sufficiently stable, the synchronisation process requires less corrections to maintain the timing error bounded in the slave clock. Depending on the stability of the frequency reference, the time correction can be realised every second, or every year. The subject of clock stability characterisation is discussed in Chapter 4.

The synchronisation process previously described, although very simple, is commonly used by many timing synchronisation processes. For instance, this is the process used to synchronise timing nodes in the LHC timing system, where a highly stable clock reference (an atomic caesium clock) is transported between various access points to act as a "local" master time reference and compensate the slave clock for the timing deviations. This procedure usually occurs in the LHC timing system twice per year [19].

To summarise, in every synchronisation process there is a mechanism that provides the frequency reference to all timing nodes in the network. In addition, an algorithm is required that synchronises timing in the slave nodes with the master clock and compensates for the measurement's delay.

However, different strategies exist to organise timing distribution in a distributed system, depending on the timing requirements of the application.

For the White Rabbit, the strategy for timing distribution is based on the distribution of the timing reference from one clock, the master clock, to all others clock nodes, slave clocks, in the local network, directly or indirectly according to a tree topology connected using Ethernet links.

In a tree topology, timing is distributed and organised in hierarchical levels, as shown in Figure 3.7.

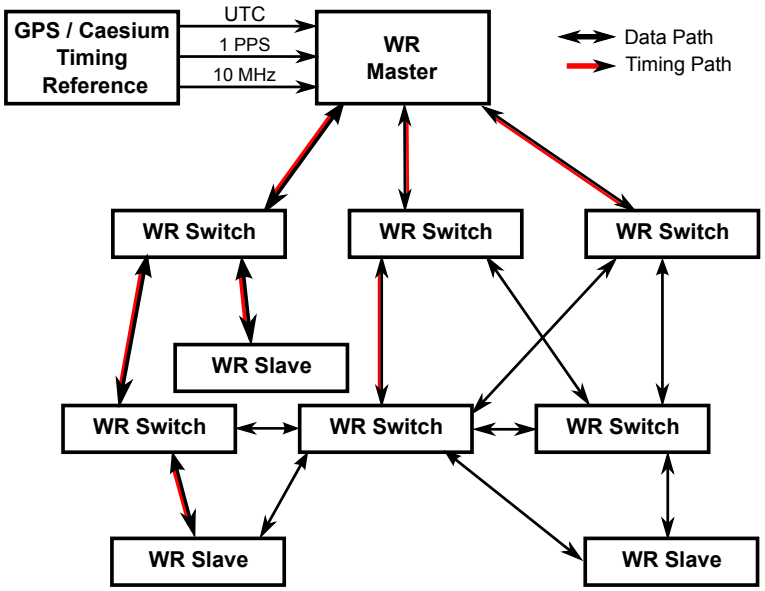


Figure 3.7: White Rabbit Network Timing/Data Topology

The WR master clock is a highly-accurate oscillator, typically a caesium atomic clock. The WR slave nodes are composed with less accurate oscillators and thus have a relatively low price.

The slave node clock reference locks, by means of a PLL, to the frequency reference generated and transmitted by the master node. The synchronisation method implemented in White Rabbit is defined by the Synchronous Ethernet specification.

When the PLL is locked to its reference clock, the slave clocks are traceable in frequency to the master clock. The PLL cleans the recovery clock from the decoded data by filtering the timing jitter generated by the CDR circuity. The loop time constant of the PLL controls the amount of phase-noise transferred from the input clock reference and the internal oscillator to the output clock. An important design requirement in the WR network is the design of the loop time constants for the slave's clock PLL. This is one of topics discussed in Chapter 6. The PLL coefficients are specified by the Sync-E which prevents for instance the accumulation of jitter in each cascade clock node.

The WR switch is an important piece of networking equipment in the WR network,

as illustrated in Figure 3.7. The WR switch works as a slave clock with respect to its uplink clock. However, for its downlink nodes, it acts as a master timing clock.

In terms of timing requirements, the main task of the WR switch is to effectively filter timing impairments on its clock reference, and to supply a highly stable timing reference to all downlink slave references. Therefore, its locking time is long enough to filter out as much jitter as possible. The WR switch is the node with the narrowest loop bandwidth, in other words with the highest input phase noise filtering capability and the best performing in holdover among all nodes in the WR network.

The WR switch is able to detect failures on the recovered clock and switch its input clock reference to either another redundant clock reference in the network or to maintain the clock in holdover mode.

Slave clocks are PLL-based, locked to an external reference. They act as a low pass filter on the input's timing signal phase fluctuations. They comply with ITU-T Rec. G.813 [45] specification.

To maintain the stability of the slave clock's timing within the specification for the WR network, the clock recovered from the decoded data in the slave node is used as a clock reference for encoding the transmitted data to the master node.

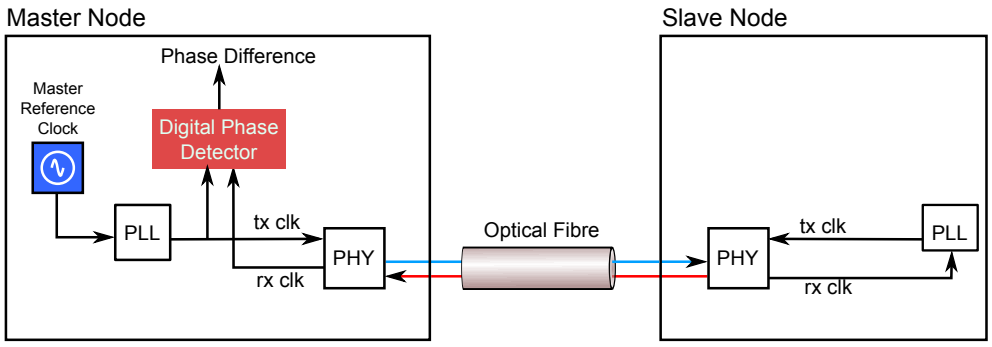


Figure 3.8: White Rabbit Frequency Loopback

In the master node, the phase between the transmitted clock and the recovered frequency is continuously measured, as illustrated in Figure 3.8. The time difference measurement, which is proportional to the phase difference between the master clock reference and the recovered clock, is sent to the slave to compensate the network

link phase variations. Long term phase deviations are due to thermal variation on the oscillators, electronics and the data link. The influence of these environmental factors is further discussed in Section 3.4.1.

The time difference measurement circuit needs to have a very small time resolution in order to maintain the phase stable within picosecond jitter. The circuit responsible for the time difference measurement is discussed later in this thesis in Chapter 5 and the circuit's implementation and performance validation is presented in Chapter 6.

To summarise, the problem of frequency syntonisation is tackled with the help of the Synchronous Ethernet specification. With Sync-E, the WR network is able to transfer frequency reference between the master and the slave nodes.

The remaining issues in the timing networks are the offset time synchronisation between nodes and the measurement of the communication link latency. These problems are both solved with the implementation of the PTP standard. By specifying hardware time-stamping, on the ingress and egress packets, PTP allows the precise measurement of the latency without the timing uncertainty of adding an operating system (OS) and remaining software layers [46]. Hardware based packet time-stamping reduces the jitter between the time samples.

Once the nodes are locked in frequency, the master node sends, via PTP messaging, its current time to the downlink slave nodes. The message containing the master time information arrives at the slave node ΔT_{ms} time later. This delay is mainly due to the transmission line. The slave nodes can easily update their timing information by setting their clock to the clock time received from the master plus the transmission latency. The transmission latency is defined by ΔT_{ms}, that is:

$$t_{slave} = t_{master} + \Delta T_{ms} \tag{3.1}$$

However, the process of obtaining the transmission line delay is not trivial [47]. The transmission line delay measurement, defined by IEEE 1588, involves the exchange

of two messages — a sync and a delay request message. Nonetheless, ΔT_{ms} as measured by the PTP requires a symmetric round trip so that $\Delta T_{ms} = \Delta T_{ms}$, which is to a certain degree of accuracy rarely the case. There are factors, some due to environmental changes, others due to the actual way data is transmitted in the link medium, that change the asymmetry of the communication link. The way in which environmental factors, namely the variation in the temperature of the link, affect the transmission line's round trip latency, is discussed in the following section.

3.4.1 Physical Layer

To achieve a sub-nanosecond clock accuracy between timing nodes in a distributed network it is necessary to have a good knowledge of the different variables that contribute to the packet transmission delay variation in an Ethernet link. The Ethernet communication link has several physical implementations such as copper or optical link. However, only the link delay model for an Ethernet optical link is presented and discussed here.

Link Delay Measurement

The WR nodes are connected by a single mode fibre (SMF) specified by ITU G.652 [48]. It adopts the latest link standard 1000Base-BX, which defines transmission over a single strand of fibre with different wavelengths being used to transmit data in each direction. The main advantage of 1000Base-BX is the reduction in installation costs.

Using this technique, the upstream signal, transmitted at a wavelength of 1310 nm, shares an optical fibre with the downstream signal, transmitted at a wavelength of 1490 nm, as illustrated in Figure 3.9. The 1300/1490 nm wavelength division multiplexing (WDM) optical transceiver is the key component in the bidirectional optical communication system.

The WDM optical transceiver can be divided into the optical part and electrical part, as shown in Figure 3.9. The optical part, called the bidirectional optical sub-assembly

(Bi-Di OSA), is responsible for converting the optical signal to the electrical signal and aligning the optical beam pass. One optical signal beam moves from the laser diode (LD) to the single-mode fibre (SMF) through the Little Connector (LC) and the other optical signal beam moves from the SMF to the photo-diode (PD). The electrical part is composed of the laser driver, transimpedance amplifier (TIA), and limiting amplifier.

The ITU G.652 fibre is the most popular standard SMF for communications infrastructures. This fibre has a simple step-index structure and is optimised for operation in the 1310-nm band. It has a zero-dispersion wavelength at 1310 nm and can also operate in the 1550-nm band, however, it is not optimised for this region. The typical chromatic dispersion at 1550 nm is 17 ps/nm-km [49]. The transmission range specified by ITU G.652 fibre can extend to 10 km.

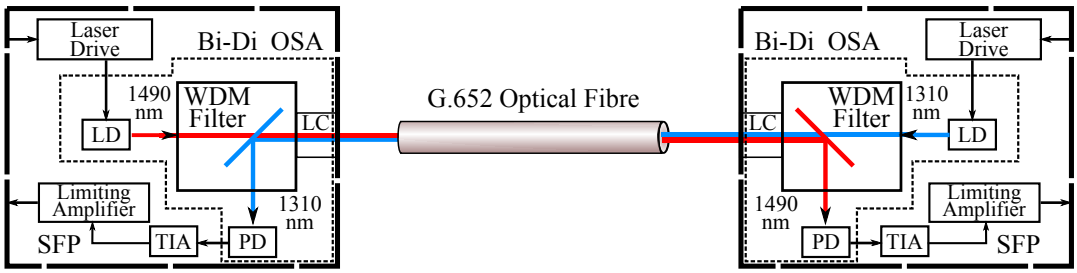


Figure 3.9: White Rabbit Transmission Link [50]

The master reference timing signal is used to encode the data that is transmitted to the slave node with a $\lambda_{ms} = 1310$ nm wavelength. At the slave node, the recovered clock, which is decoded from the data received from the master, is used to encode the slave's data link and the transmission back to the master has a $\lambda_{sm} = 1490$ nm wavelength.

The transmission delay of a packet between the master node and the slave node is defined as:

$$\Delta T_{ms} = \Delta T_{tx_m} + \Delta T_{L_d} + \Delta T_{rc_s} \tag{3.2}$$

where ΔT_{tx_m} is the delay in the master transmission circuit, ΔT_{L_d} is the delay due to the transmission of the packet in the link and ΔT_{rc_s} is the delay in the slave receiver circuit.

The packet transmission delay travelling in the opposite direction, between the slave and the master, is defined as:

$$\Delta T_{sm} = \Delta T_{tx_s} + \Delta T_{L_u} + \Delta T_{rc_m} \tag{3.3}$$

where ΔT_{tx_s} is the delay in the slave transmission circuit, ΔT_{L_u} is the delay due to the transmission of the packet in the link and ΔT_{rc_m} is the delay in the master reception circuit.

The delay represented by ΔT_{L_d}, ΔT_{L_u} have a higher order value than ΔT_{rv_m}, ΔT_{rv_s}, ΔT_{tx_m}, ΔT_{tx_s}. While the length of the links is in the order of km, the remaining delays are due to the traces on the circuit system, in order of cm.

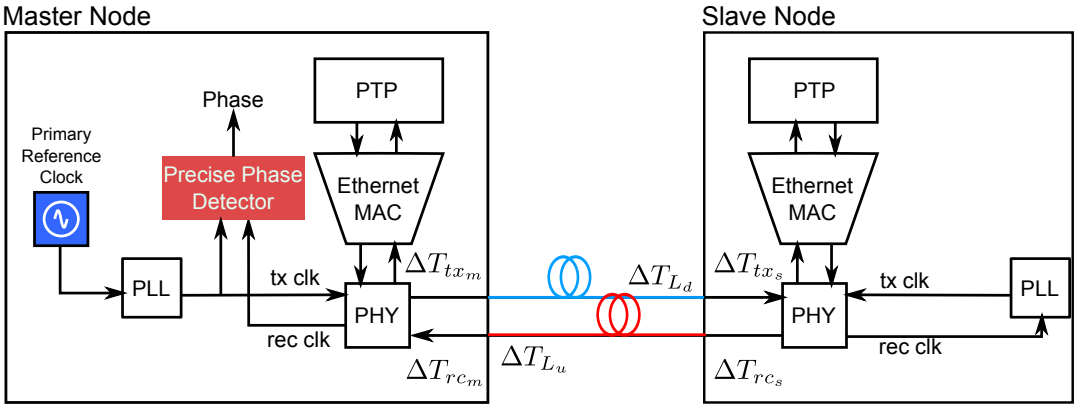


Figure 3.10: Delays in a White Rabbit Link

In order to achieve system synchronisation it is required, as described by Eq. 3.1, to know the transmission link delay, ΔT_{ms}. The link delay estimation is accomplished by PTP the protocol that continuously measures the round-trip link delay, t_{delay}.

$$t_{delay} = \Delta T_{ms} + \Delta T_{sm} \tag{3.4}$$

Assuming that $\Delta T_{ms} = \Delta T_{sm}$, PTP uses Eq. 3.4 to estimate one link delay ΔT_{ms}, thus achieving timing synchronisation.

For the estimation of ΔT_{ms} to be accurate it is required that the propagation delay between the uplink and the downlink is equal. However, the propagation delay of a pulse in an optical fibre differs for different wavelengths. This is due to the different wavelengths' refractive index [51].

The link delay measurement error is simply due to the difference in delay between the two transmission wavelength, which is caused by the difference in the group refractive index between the two wavelengths [52].

The relationship between the refractive index and the wavelength in a medium is given by Sellmeier's equation [53]:

$$n(\lambda)^2 = 1 + \sum_{i=1}^{k} \frac{a_i \lambda^2}{\lambda^2 - b_i} \tag{3.5}$$

where λ is the wavelength in μm, and the set of parameters a_i, b_i correspond to a specific type of material such as pure silica. A series of a_i and b_i have been taken from reference [54] and shown in Table 3.2.

Table 3.2: Sellmeier's Equation Parameters for Pure Silica Glass

Material	a_1	a_2	a_3	b_1	b_2	b_3
Pure Silica	0.6965325	0.4083099	0.8968766	0.004368309	0.01394999	97.93399

By adopting Sellmeier's equation with the parameters, given in Table 3.2, the relationship between the variation of the refractive index with the wavelength is calculated and plotted in Figure 3.11.

The transmission's delay of a wave in a medium is given as [55] :

$$t = \frac{d\, n}{c} \tag{3.6}$$

where t is the time delay, c is the speed of light in vacuum, n is the refractive index

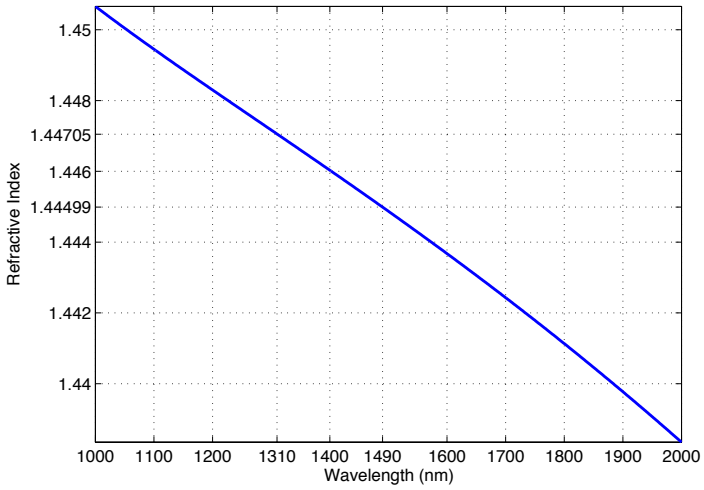


Figure 3.11: Refractive Index for a Pure Silica Glass

and d is the link's transmission length.

The round trip time asymmetry is estimated as the difference between the propagation delay of the uplink and downlink wavelengths, that is:

$$t_{asymmetry} = t_{up} - t_{dn} = \frac{d\,(n_{up} - n_{dn})}{c} \tag{3.7}$$

From Figure 3.11, is estimated that for λ_{MS}=1490 nm, n_{up} is ~1.445 and for λ_{SM}=1310 nm, n_{dn} is ~1.447.

For a 10 km link length the calculated time asymmetry between the two nodes is $t_{asymmetry}$ ~(-66.71) ns in a pure silica glass medium.

The calculation of the delay asymmetry was also realised for a G.652 type fibre, the SMF28 [56]. From the fibre's datasheet [56] the values for the refractive index were obtained. For a wavelength, λ, of 1310 nm, the refractive index is 1.4676, while for 1550 nm, the refractive index is 1.4682. By carrying out a linear numerical interpolation between these two values the refractive index for a wavelength of 1490 nm is estimated to be ~1.4681. For the SMF28 fibre, the estimated round trip asymmetry, $t_{asymmetry}$, for 10 km links is ~(-16.68) ns. For this fibre, the uplink

pulse travels faster than the downlink one.

A different method to calculate the uplink and downlink delay is through the values for the total chromatic dispersion of the fibre. The chromatic dispersion of a fibre is expressed in ps/(nm·km), representing the differential delay, or time spreading (in ps) for a source with a spectral width of 1 nm travelling on 1 km of the fibre. A first-order calculation of dispersive pulse broadening using the dispersive parameter D given a propagation length L and pulse spectral width $\Delta\lambda$ can be conveniently obtained as:

$$D(\lambda) = \frac{\Delta t}{d\Delta\lambda} \Rightarrow \Delta t = D(\lambda) d\Delta\lambda \tag{3.8}$$

By integrating Eq. 3.8 the time difference of arrival between two pulses transmitted with different wavelengths is obtained as:

$$t = d \int_{\lambda_{dn}}^{\lambda_{up}} D(\lambda) d\lambda \tag{3.9}$$

where t is the delay in ps , d in the link length in km and λ_{up} and λ_{dn} are the uplink and downlink wavelengths, respectively.

The total chromatic dispersion, given by the sum of the material and waveguide dispersion contributions, is specified by ITU-T G.652 using the following equation [53]:

$$D(\lambda) = \frac{S_o}{4} \left(\lambda - \frac{\lambda_o^4}{\lambda^3} \right) \tag{3.10}$$

where λ_0 is the zero dispersion wavelength and S_o is the chromatic dispersion slope at λ_0. The G.652 specification defines for the particular S_o the minimum and maximum values for λ_0. These values are λ_{min}=1300 nm, λ_{max}=1324 nm and S_o= 0.093 ps/(nm·km).

The chromatic dispersion limits for a G.652 fibre are calculated and plotted in Figure

3.12.

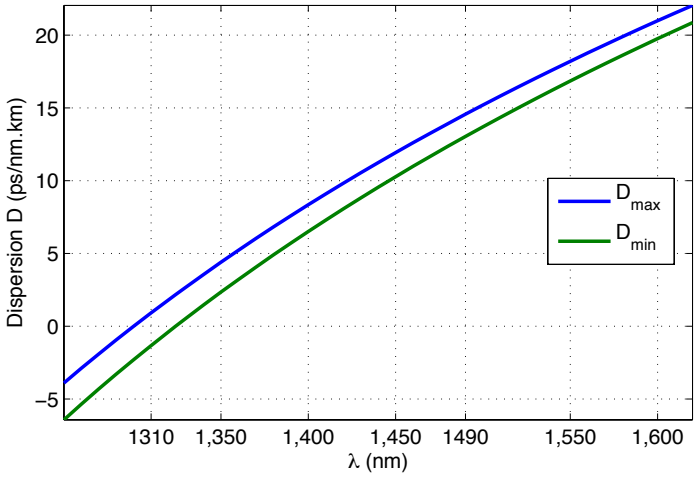


Figure 3.12: Chromatic dispersion of a single mode fibre (G.652)

Integrating Eq. 3.10 results in:

$$\int_{\lambda_1}^{\lambda_2} D(\lambda)d\lambda = \frac{S_o}{8}\left(\lambda^2 + \frac{\lambda_o^4}{\lambda^2}\right)|_{\lambda_1}^{\lambda_2} \tag{3.11}$$

The total chromatic dispersion is calculated using Eq. 3.11 with the parameters specified by G.652 with $S_{0max} = 0.093$ ps/(nm·km) and $\lambda_{0min} = 1300$ nm, $\lambda_{0max} = 1324$ nm. This results in a time asymmetry round-trip delay generated by the two wavelengths, 1310 nm and 1490 nm, for a distance of 10 km, between **-11.94** ns and **-14.66** ns.

The refractive index of the optical fibre is not only dependent on the wavelength but also on the temperature. The thermo-optic coefficient, dn/dT , is the refractive index variation with respect to temperature. This coefficient contains the electronic and optical phonons contribution. The electronic effects, in particular the temperature variations of the electronic absorption peak energy gap, have the dominant contribution [57]. Therefore, the thermo-optic coefficient can be described in terms of the linear expansion coefficient α and the temperature variation of the energy gap dE_g/dT[58].

$$\frac{dn}{dT} = \frac{n_0 - 1}{2n}\left(-3\alpha R - \frac{1}{E_{eg}}\frac{dE_{eg}}{dT}R^2\right) \tag{3.12}$$

where n_0 is the low-frequency refractive index in the infra-red region, α is the coefficient of the thermal expansion, E_{eg} is the excitonic band-gap, and R is the normalised dispersive wavelength, defined by

$$R = \frac{\lambda^2}{\lambda^2 - \lambda_{ig}^2} \tag{3.13}$$

where λ_{ig}^2 is the wavelength that corresponds to the isentropic band-gap [59].

Eq. 3.12 shows that the thermo-optic coefficient depends on two terms. The first term, describes the contribution from the thermal expansion coefficient. The second term, describes the contribution from the temperature dependence of the excitonic band-gap. The contribution of the first term is small, as the magnitude of α is in the order of $10^{-6}K^{-1}$[58]. However, the temperature variation of E_{eg} is in the order of $10^{-4}eVK^{-1}$ and is normally negative for glasses. The contribution from the second term is therefore positive and is the dominant term, resulting in a positive thermo-optic coefficient.

For the Corning single-mode fibre, SMF28, the experimental thermo-optic coefficient value dn/dT is 1.06323 $\times 10^{-5}/^{\circ}$ C [60].

Using the experimental thermo-optic coefficient value for the SMF-28 fibre, the refractive index variation with temperature for the uplink and downlink wavelengths was calculated and is shown in Figure 3.13.

From Figure 3.13 the link's delay variation, for a 10 km fibre link when it suffers a variation of temperature of 20^o, is calculated to be ~6.69 ns. The delay asymmetry between the uplink and downlink wavelengths changes by ~0.1 ns per 20^o for a 10 km link length.

The link's latency variation due to temperature variations in the link needs to be

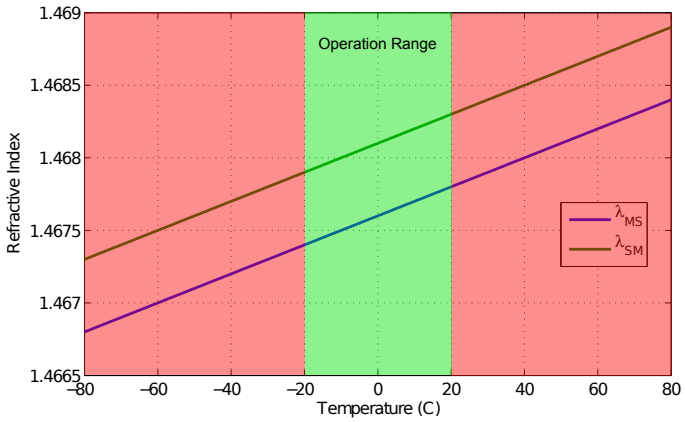


Figure 3.13: Refractive Index vs Temperature

measured and compensated to guarantee that the slave nodes maintain the sub-nanosecond accuracy.

The WR Master's clock reference is transported to the slave nodes via Ethernet, by encoding the transmitted data with the master's clock reference. On the slave node, the encoding clock and data is demultiplexed from the received encoded data. The encoding clock is sometimes referred to as the recovery clock of the slave node. The recovery clock guarantees that both the Master and Slave nodes share the same frequency reference. However, the phase difference between the nodes varies due to the link length and external environmental factors.

An example that demonstrates the effect of the link asymmetry in the synchronisation process between two WR nodes, is shown in Figure 3.14.

The synchronisation process begins with a message sent by the master node with the time-stamp t1, the Sync message. After ΔT_{L_d}, the Sync message arrives at the slave node and is time-stamped at t2. The slave clock can now update its clock to match the master clock so that:

$$t2 = t1 + \Delta T_{L_d} + \Delta_{\text{offset}} \tag{3.14}$$

Where Δ_{offset} is the time offset between the master and the slave node, transmitted

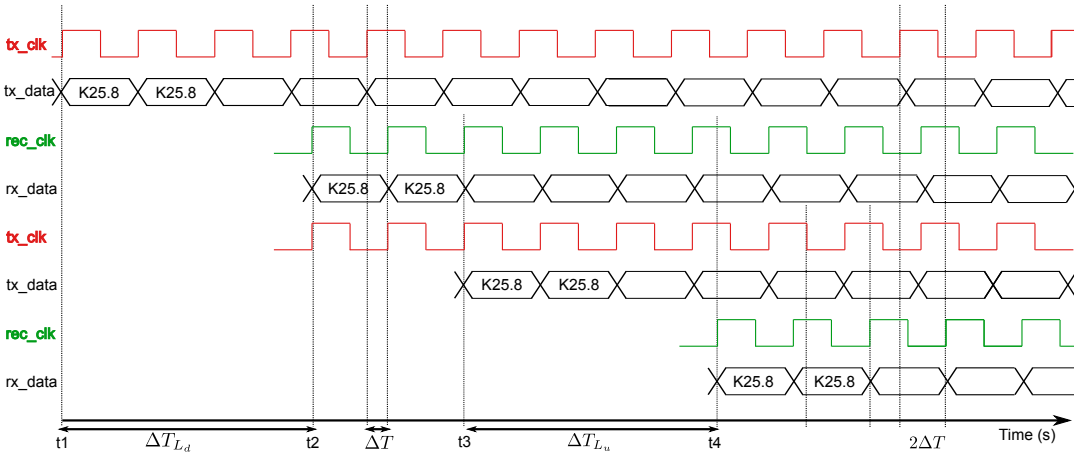


Figure 3.14: Delays in a White Rabbit single link

by the Sync Message. The ΔT_{L_d} is the transmission delay between the master and the slave node, and is an unknown time parameter.

To measure ΔT_{L_d}, the slave node sends a new message to the master that is timestamped in the slave node at t3. This message, when received by the master, is timestamped at t4. This results in:

$$\Delta T_{L_d} = t2 + \Delta_{\text{offset}} - t1 \tag{3.15}$$

$$\Delta T_{L_u} = t4 - (t3 + \Delta_{\text{offset}}) \tag{3.16}$$

In order to solve Eq. 3.15 and Eq. 3.16, the PTP standard assumes that the uplink and downlink have symmetric transmission delay so that $\Delta T_{L_d} = \Delta T_{L_u}$. With this assumption, the slave node can now calculate the unknown parameters.

$$\Delta_{offset} = \frac{(t1 - t2) - (t3 - t4)}{2} \tag{3.17}$$

$$\Delta T_{L_d} = \frac{(t2 - t1) - (t3 - t4)}{2} \tag{3.18}$$

$$\Delta T_{L_u} = \frac{(t1 - t2) - (t3 - t4)}{2} \tag{3.19}$$

The slave node at this point knows all the timing variables and can update its clock according to the BMC algorithm. However, in the case of WDM transmission, the round-trip delay is not symmetric. Due to the link delay asymmetry, the synchronisation process contains an error that degrades the accuracy of the timing network. For example, for a link length of 10 km, the downlink delay for n_{1310}= 1.4676 is ~48953.9 ns. The uplink transmission delay, for n_{1490}= 1.4681 is ~48970.5 ns.

Assuming a round-trip link delay symmetry for the uplink and the downlink, the estimation of ΔT_{L_d} would result in (48953.9 + 48970.5)/2 ns equal to 48962.2 ns. This result shows that the slave node clock would be inaccurate in the downlink transmission delay estimation by (49083.6 - 48962.2) ns, that is around approximately 121.4 ns. For applications where sub-microsecond timing accuracy is defined, the round-trip delay symmetry assumption is reasonable. However, for sub-nanosecond timing accuracy applications, such as the one specified by the WR network, the round-trip link asymmetry is another unknown variable that needs to be accounted for.

This is accomplished in the WR network by performing continuous measurements in the fibre, to know precisely the different transmission delay for the uplink and the downlink link, as illustrated in Figure 3.14.

Let's define the ratio between the downlink and uplink transmission delay as:

$$\frac{\Delta T_{L_u}}{\Delta T_{L_d}} = \frac{\Delta T_{L_u}}{\Delta T_{L_d}} = \frac{n_{1310}}{n_{1490}} = \beta \tag{3.20}$$

The new set of equations used in the synchronisation process in the BCM algorithm, to compensate for the link asymmetry, are:

$$\Delta_{offset} = \frac{t2 - t1 - \beta(t4 - t3)}{1 + \beta} \tag{3.21}$$

$$\Delta T_{L_d} = \frac{(2 + \beta)(t2 - t1) - \beta(t4 - t3)}{1 + \beta} \tag{3.22}$$

$$\Delta T_{L_u} = \frac{(t1 - t2) + (2\beta + 1)(t4 - t3)}{1 + \beta} \tag{3.23}$$

The term β compensates for the asymmetry in the transmission link delay and decreases the static time error between the network nodes.

Nonetheless, there is still a remaining issue which needs to be solved in order to achieve the sub-nanosecond timing accuracy. The master and slave clock have the same clock reference, however, they have a phase difference. The phase difference is derived not just from the length of the transmission link but also due to the fibre temperature variations. The phase difference between the two clocks cannot be measured using the packet time-stamping process. The reason for this is that the slave node can only time-stamp the transmitted/received packet with a time counter of just 8 ns time resolution (the 125 MHz data embedded clock). For example, let's define a network link with 10 km length; the approximate transmission delay of a packet between two nodes is ~48953.9 ns. However, due to the 8 ns time-stamp granularity of the recovery clock, the measured arrival time would be ~48952.00 ns. In the example presented, the slave clock would be inaccurate by ~1.9 ns. This inaccuracy is represented as Δ_T in Figure 3.14.

In order to meet the specification of sub-nanosecond time accuracy synchronisation it is necessary to phase shift the slave node clock by an amount Δ_T. This phase difference is impossible to measure directly, due to the distributed nature of the nodes. An indirect measuring method is to loopback the frequency reference recovered from the data and measure the time difference between the master recovery clock (from the slave) and the source master clock in the master node, as illustrated in Figure

3.8. This frequency loopback technique can be seen in many timing distribution applications [25], where it is required to compensate for phase variation due to external interferences in the transmission line.

The delay added to the time offset is estimated as being half the time difference that is measured in the master node. In addition, this method allows the WR slave nodes to compensate for the temperature variations on the fibre, as it is continually measuring the time difference between the two frequency references. The link delay variation due to temperature oscillations (wander) is solved by looping back the clock signal recovered in the slave to the master node.

The phase shifter and time difference circuit should be able to measure time difference with sub-picosecond time resolution. This circuit is the heart of the White Rabbit network in achieving the specified sub-nanosecond accuracy. A digital circuit capable of reaching such performance is one of the contributions of this work and is described in Chapter 5.

3.4.2 Communication Protocol

Ethernet is by definition a best-effort transfer protocol. It offers no guarantees with respect to the amount of bandwidth that can be used by one application on shared connections. The eight classes of services are only defined by the standard, leaving each manufacturer to choose a specific solution for how to handle prioritised traffic.

White Rabbit uses the same frame structure as defined by the IEEE 802.3Q standard. Its main difference is the access to the transmission medium. As described earlier, in full-duplex transmission, the access of an Ethernet frame to the transmission medium starts as soon as the MAC receives the signal that the last frame transmission has concluded, in other words when the communication link is free for transmission.

3.5 Summary

In this chapter, a detailed description of how an Ethernet network achieves synchronisation from the physical link was given. The Ethernet standard multiplexes data and clock signals to be transmitted to nodes by coding the data and clock using the 8B/10B encoding process. A set of specially encoded symbols is used to synchronise the Ethernet nodes, so that data can be transmitted and received. In Section 3.2, the Synchronous-Ethernet standard, defined as a mechanism to transport frequency over Ethernet links, was described. This standard specifies strict clock requirements for the network node, as well as the requirements of the PLL circuits. The IEEE 1588 standard was described in Section 3.3, which defines a mechanism of message exchange between the timing nodes to achieve clock synchronisation within the network.

In Section 3.4, an introduction to the White Rabbit Project was given. The objectives of the WR project were described, as well as some of the standard technologies used. In addition, in Section 3.4, the Ethernet's link delay model was investigated and the factors that need to be compensated to enable sub-nanosecond timing accuracy in the network were presented.

In Chapter 3, a link delay model of the White-Rabbit network was described. In addition, based on the model, a methodology was proposed to provide White Rabbit with the sub-nanosecond timing accuracy.

Chapter 4

Timing Characterisation

This chapter deals with the problem of timing clock noise characterisation in both frequency and time domain.

Chapter 4 starts with the mathematical description of a timing signal, where the characterisation of the phase noise is the main variable of interest. In Section 4.2, the characterisation of clock stability in the frequency domain is described. Section 4.3 provides methods to characterise the long term stability of a clock timing signal in the time domain. The described methods include the Allan Variance and the Modified Allan Variance. Section 4.4 introduces jitter and wander timing quantities, since they are used to specify timing noise in synchronous networks. The frequency and time domain characterisation of the oscillator stability from a Rubidium and a Caesium atomic clocks is given.

4.1 Timing Signal

A timing signal or clock signal may be defined as a periodic or pseudo-periodic signal used to control the timing of actions.

In principle, clocks consist basically of a generator of oscillations based on any periodic physical phenomenon such as the electrical signal generated by the mechanical

vibration of a crystal with piezoelectric material like Quartz [1].

Clocks may be differentiated in two groups: The autonomous clocks supply a timing signal generated from an internal oscillator, running independently of external influence. In a slave clock on the other hand, the oscillator is controlled by an external reference, by means of a PLL system.

Timing systems are usually described as sine waves for ease of mathematical analysis. The oscillator timing output signal can be described as:

$$u(t) = (A + \epsilon(t)) + \sin\ (2\pi \upsilon_0\, t + \varphi(t)) \tag{4.1}$$

where υ_0[†] represents the nominal frequency - the carrier frequency. The ϵ is the parameter representing the amplitude noise. The additive amplitude noise can be neglected due to the oscillator's amplitude regulation and also because timing systems are more interested in the zero crossing levels [5]. Thus, the instantaneous phase $\varphi(t)$ in Eq. 4.1 is the only parameter that contains the noise and is classified as a random process [61]. The phase noise is associated with random frequency fluctuations. In a case of sufficiently small phase fluctuations $\varphi(t)$, the Eq. 4.1 expresses the narrowband phase modulated signal.

Stability measurements utilise the method of a comparison between the measured oscillator and a reference source. For other purposes, the reference is considered as an ideal harmonic source with the zero term $\varphi(t)$ and has to be of an higher order stabler than the source of the measured signal.

A fractional frequency, $y(t)$, and time fluctuations, $x(t)$, are very helpful quantities used for stability analysis. The fractional frequency values, y_i, are dimensionless, while the time fluctuations values, x_i, are measured in seconds.

The fractional frequency can be obtained by comparing the measured oscillator with the reference according to:

[†]The term υ is commonly used to describe the fundamental frequency of an oscillator. This is differ from the term f generally used to describe a given Fourier Transform frequency.

$$y(t) = \frac{(\upsilon(t) - \upsilon_0)}{\upsilon_o} = \frac{1}{2\pi\upsilon_0}\frac{d\varphi}{dt} = \frac{dx}{dt} \tag{4.2}$$

where $\upsilon(t)$ represents the time variant frequency of the measured oscillator, υ_o is the fundamental frequency of the reference oscillator. Phase fluctuations $\varphi(t)$ are related to time fluctuations $x(t)$ by the term :

$$\varphi(t) = 2\pi\upsilon_0\, x(t) \tag{4.3}$$

Since the values y_i and x_i are random quantities, statistical parameters are needed for their description, some of which are classically used in statistical random process theory, for example correlation functions and spectral densities [62].

Characterising the quality of a clock, or equivalent of its timing signal, is one of the most debated issues in practical applications of clocks [63]. In most common sense, one simple term is used to refer to clocks quality: the precision, somehow denoting how much the clock under test is close to a reference clock, in both phase and frequency. Nevertheless, the term "precision" is even not included in the ISO International Vocabulary of Metrology [64] and therefore its use is not recommended.

The quality of a clock is usually defined by means of two other basic terms: the stability and the accuracy. These two terms are understood by people working in various fields with subtle different meanings, nonetheless, the terms definitions provided are considered quite general and widely accepted. The stability of a measuring instrument is its ability to maintain constant its metrological characteristics with time [64].

The stability of a clock is, therefore, its capability of generating a time interval (or frequency) with constant value, such a value may be different from the nominal value, but it is intended constant in time. In other words, the stability of a clock deals with the measurement of random and deterministic variations of its instantaneous frequency (or of the time generated) compared to the nominal value (in practice to

that of a reference clock), over a given observation interval.

For example, a stability measure of the instantaneous frequency based on the concept of variance $\sigma^2(\tau)$, where τ is the observation time interval, informs about the instantaneous frequency variance that is expected by collecting measurements over a time interval τ. When the observation interval τ is small, the expression short-term stability is commonly used, otherwise the expression long-term applies.

Several quantities have been defined aiming at characterising clock stability. With different properties, they highlight distinct phenomena in the phase noise or they are more oriented to specific applications.

On the other hand, the accuracy of measurement denotes the closeness of the agreement between the result of a measurement and a true value of the quantity that is being determined by the measurement [64]. In the context of clock quality definition, thus, accuracy of a time scale means its agreement with UTC, while accuracy of the frequency delivered by a frequency standard means its agreement with the nominal frequency value. Qualitatively speaking accuracy tells how much a clock is "right". The accuracy of a clock is defined as the maximum time or frequency error, which may be measured in general over the whole clock life (for example, 20 years), unless differently specified. Therefore, the time accuracy means how well a clock agrees with UTC over the specified period, while the frequency accuracy is the maximum frequency error $\Delta\upsilon_{max}$ compared to the nominal value υ_0 [5].

The difference between the concepts of stability and accuracy can be explained by illustrating a practical clock characterisation example. For instance, a clock could have a significant frequency errors, and thus poor frequency accuracy, while still keeping constant frequency error during its lifetime, featuring perfect frequency stability. A hydrogen-MASER clock typically has better frequency stability than a Caesium clock from second to second, or from hour to hour, but often not from month to month and longer. Quartz-oscillators can be highly stable in the short term, but they drift in frequency and do not feature the excellent long term frequency accuracy of atomic

clocks [65].

4.2 Clock stability in the frequency domain

The most straightforward and intuitive way to characterise the accuracy and stability of the timing signal, $u(t)$, supplied by a clock is to evaluate its Power Spectral Density (PSD) function.

Analysis in the Fourier frequency domain is of great importance both for theoretical and for practical application purposes, in terms of power spreading in the frequency domain, a primary specification for many applications. For these reasons, the spectral density concept has been widely used for oscillator stability characterisation.

In the real case of timing signal affected by some phase and amplitude noise, the PSD ideal pulse spreads and exhibits some spectral content also at frequencies $f \neq 0$. The base around the ideal frequency $f = 0$, therefore is commonly found in PSD measured on real oscillators.

The single side power spectrum, $S_u(f)$, is a continuous function, proportional to the timing-signal proportional to a matched load per unit of bandwidth centred on v_o. It is measured in [W/Hz] or [V^2/Hz] units.

Unfortunately, the spectrum $S_u(f)$ is definitely not a good tool to characterise clock frequency stability, in other words, given $S_u(f)$, it is not possible to determine unambiguously whether the power at various Fourier frequencies is the result of amplitude rather than phase fluctuations in the timing signal $u(t)$. Nevertheless, if the power due to amplitude modulation noise is negligible and the phase fluctuations are small, that is the mean square value $\langle \varphi^2(t) \rangle$ is much smaller than 1 rad^2 as it happens in practical clocks, then spectrum $S_u(f)$ has approximately the same shape of the phase noise spectrum $S_\varphi(f)$ [62]. However, it is still difficult to obtain quantitative results about $\varphi(t)$ from it.

4.2.1 Translation among frequency domain measures

In the Fourier frequency domain, the most commonly used clock stability measures are the single sided PSDs of $\varphi(t)$, $x(t)$, $y(t)$ denoted as $S_\varphi(f)$ with units [rad^{2}Hz^{-1}], $S_x(f)$ with units [s^2Hz^{-1}] and $S_y(f)$ with units [Hz^{-1}], respectively, because they describe directly the time and frequency stability characteristics.

The correlation function and spectral densities carry exactly the same information about the random process. However, the spectral density format is very often more desirable for engineering purposes [66].

The PSDs defined above are equivalent representations of the same noise process. In fact, the following relationships hold among them:

$$S_y(f) = \frac{f^2}{\upsilon_0^2} S_\varphi(f) \tag{4.4}$$

$$S_x(f) = \frac{S_\varphi(f)}{(2\pi\upsilon_0)^2} \tag{4.5}$$

$$S_y(f) = (2\pi f)^2 S_x(f) \tag{4.6}$$

The PSD $S_y(f)$ was first recommended in 1971 by IEEE as standard frequency stability measure in the Fourier frequency domain [67]. However, this recommendation is not valid any more.

It is worthwhile noticing that, under the assumption of Gaussian stationary random process, the power spectral density (or equivalently the autocorrelation function, which is its inverse Fourier transform) contains maximum information about the random process [66].

The other two spectral measures of phase and frequency stability that are encountered are the PSD of non-normalised frequency fluctuations $\Delta\upsilon(t)$

$$S_{\Delta v}(f) = v_0^2 S_y(f) \tag{4.7}$$

Measured in [Hz^2/Hz] , that is [Hz], and the single-sideband measure of phase noise

$$\mathscr{L}(f) = \frac{1}{2} S_\varphi(f) \tag{4.8}$$

The quantity $\mathscr{L}(f)$ is measured in [dBc/Hz] units (decibel of the carrier-over-noise power per bandwidth unit), as

$$\mathscr{L}(f)|_{\frac{\text{dBc}}{\text{Hz}}} = 10 \ \log_{10}(\mathscr{L}(f)) \tag{4.9}$$

This is the most common function used to specify phase noise; it's rather commonly in technical specification of commercial oscillators.

Currently, the IEEE standard 1139 [68] recommends reporting the phase noise spectra as $\mathscr{L}(f)$. The reason of its appeal lies in the physical interpretation of $\mathscr{L}(f)$, a signal-to-noise ratio that can be assessed readily on a spectrum analyser.

4.2.2 Noise expression in the frequency domain

In the frequency domain, the most frequently metric used to characterise the phase noise measured on timing clocks is by noise power-law model.

This model is based on the relation of the fractional frequency y(t) and gives the one-sided spectral density of frequency fluctuations [69].

$$S_y(f) = h(\alpha) f^\alpha \tag{4.10}$$

where f is the offset frequency from the nominal carrier frequency v_0. The constant $h(\alpha)$ is the transfer function proportional to α, in other words it is a measure of the noise level. The exponent α represents the power of the offset frequency that defines

parameters of the noise, usually called the power-law noise [63].

$S_y(f)$ is commonly depicted in the logarithmic-logarithmic plot, where the power α defines asymptotic slopes of the $S_y(f)$ curve. An example of the fractional frequency spectral density course covering all basic noise processes is illustrated in Figure 4.1.

The exponent α takes the natural values {-2, -1, 0, 1, 2} and characterise the kind of noise:

- White Phase Modulation (WPM) for α= 2
- Flicker Phase Modulation (FPM) for α= 1
- White Frequency Modulation (WFM) for α= 0
- Flicker Frequency Modulation (FPM) for α= -1
- Random Walk Frequency Modulation (RWFM) for α= -2

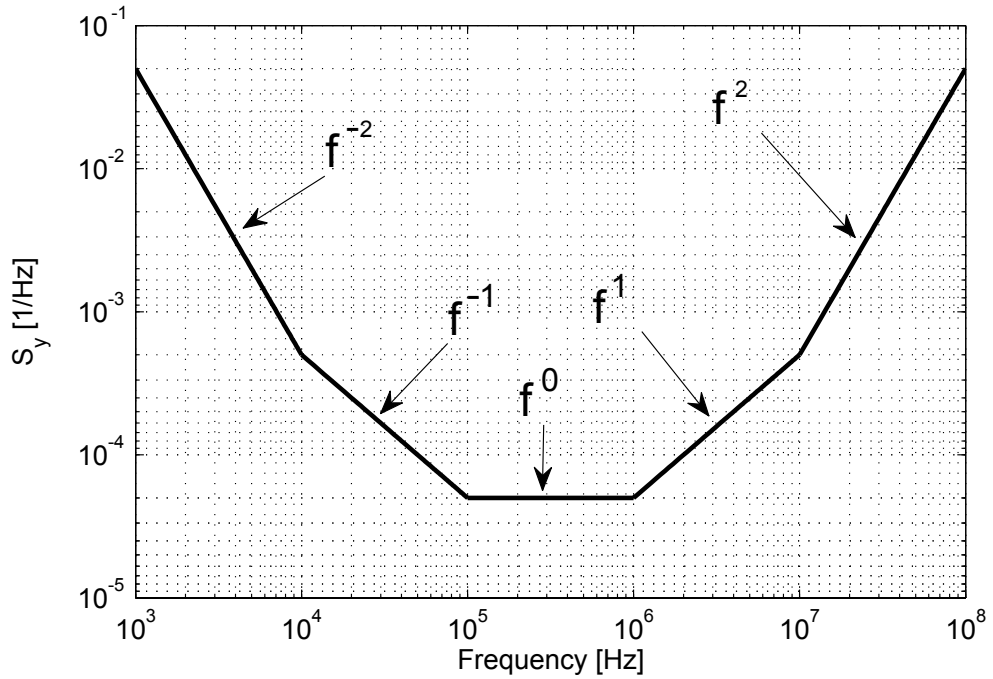


Figure 4.1: Noise Power Law $S_y(f)$

Another way of describing the phase noise in the frequency domain is the one-sided spectral density of phase fluctuations $S_\varphi(f)$.

The spectral density of phase fluctuations is also divided into five asymptotic parts, but their slopes are different from Eq. 4.10. The asymptotic expression describing the different noise power is given as:

$$S_\varphi(f) = h(\beta) f^\beta \tag{4.11}$$

where asymptotes slopes are proportional to β-th power of the offset frequency and $h(\beta)$ represents the transfer function for the certain slope of power β. At this time, powers β directly show slopes of the phase noise $\mathscr{L}$(f) that is the most common and transparent single sideband expression with the unit dBc/Hz [69].

The relationship between powers α and β can be calculated as $\beta = \alpha + 2$ [69]. Slope characteristic of the five independent noise processes are plotted vs the frequency in Figure 4.2.

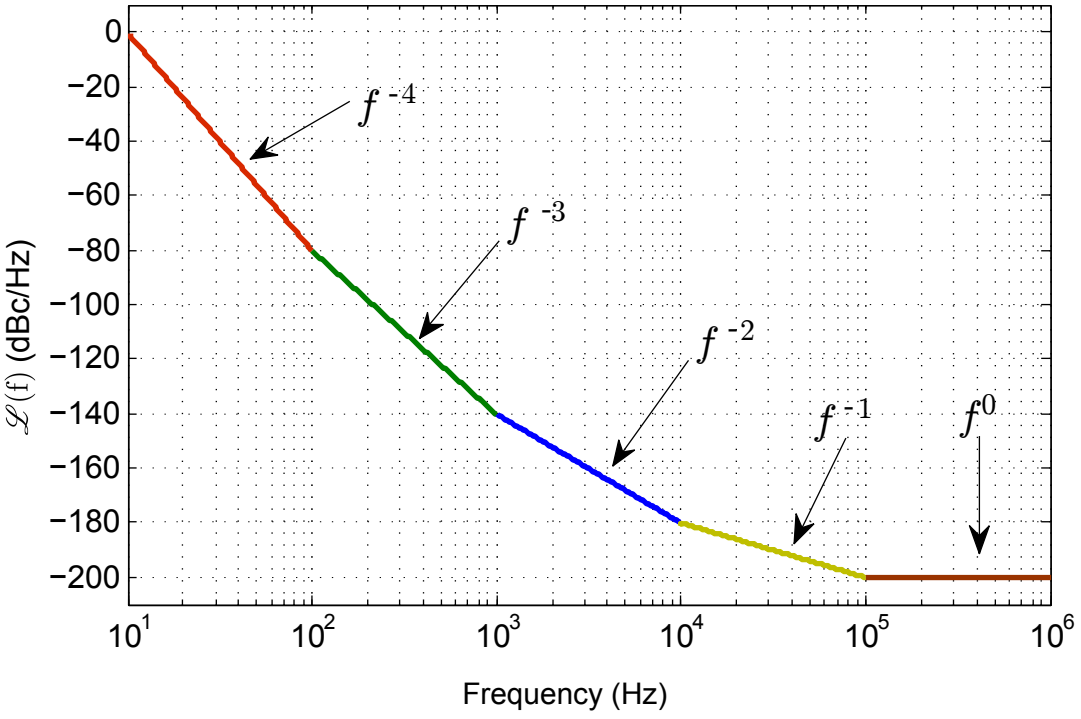


Figure 4.2: Noise Power Law $S_\varphi(f)$

The PSD of an oscillator's clock is in most cases a combination of different power-law noise processes. It is very useful and meaningful to categorise the noise processes.

The first job in evaluating a spectral density plot is to determine, which type of noise exists for a particular range of Fourier frequencies. In addition, the different power-law noise processes are described as follows [69].

1. **Random walk FM** $(1/f^4)$ noise is difficult to measure since it is usually very close to the carrier. Random walk FM usually relates to the oscillator's physical environment. If Random Walk FM is a predominant feature of the spectral density plot then mechanical shock, vibration, temperature, or other environmental effects may be causing "random" shifts in the carrier frequency.

2. **Flicker FM** $(1/f^3)$ is a noise whose physical cause is usually not fully understood but may typically be related to the physical resonance mechanism of an active oscillator or the design or choice of parts used for the electronics, or environmental properties. Flicker FM is common in high-quality oscillators, but may be masked by White FM $(1/f^2)$ or Flicker PM (l/f) in lower-quality oscillators.

3. **White FM** $(1/f^2)$ noise is a common type found in passive-resonator frequency standards. These contain a slave oscillator, often quartz, which is locked to a resonance feature of another device that behaves much like a high-Q filter. Caesium and rubidium oscillators have White FM noise characteristics.

4. **Flicker PM** $(1/f)$ noise may relate to a physical resonance mechanism in an oscillator, but usually is added by noisy electronics. This type of noise is common, even in the highest quality oscillators, because in order to bring the signal amplitude up to a usable level, amplifiers are used after the signal source. Flicker PM noise may be introduced at these stages, it may also be introduced in a frequency multiplier. Flicker PM can be reduced with good low-noise amplifier design (for example, using RF negative feedback) and hand-selecting transistors and other electronic components.

5. **White PM** (f) noise is broadband phase noise and has little to do with the

resonance mechanism. Probably produced by similar phenomena as Flicker PM $(1/f)$ noise [62]. Stages of amplification are usually responsible for White PM noise. This noise can be kept at a very low value with good amplifier design, hand-selected components, the addition of narrowband filtering at the output, or increasing, if feasible, the power of the primary frequency source.

A single oscillator can have all five model noise processes in its phase noise, but, in general, only two or three noise processes are dominant [62].

The phase-noise spectrum of a high-quality Rubidium oscillator is shown in Figure 4.3. The measurement was done with Agilent's E5052B, a signal source analyser [70].

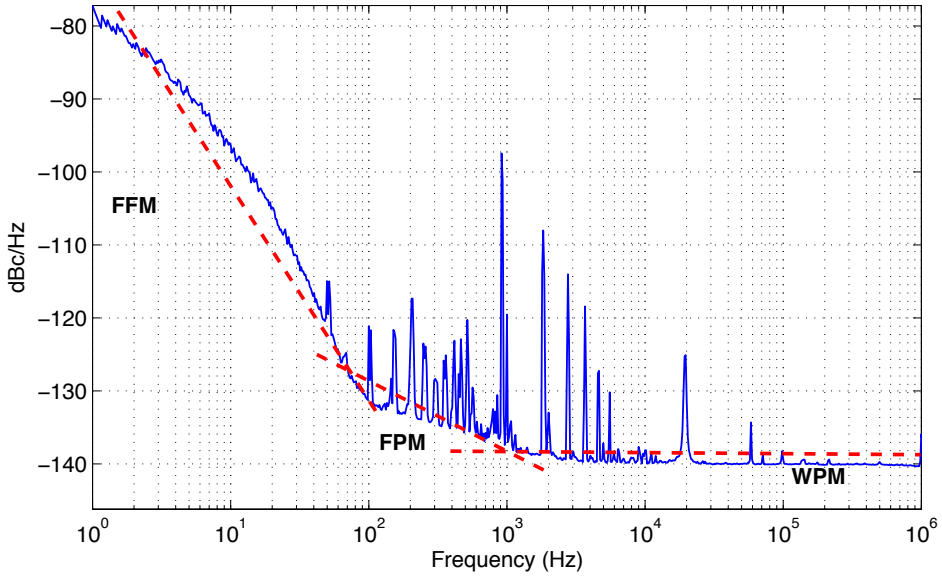


Figure 4.3: Phase-Noise Spectrum of a Rubidium Atomic Clock

In Figure 4.3, three noise processes are identified. First, for offset frequencies between 1 Hz and 50 Hz Flicker FM noise is found. Second, from offset frequencies between 50 Hz and 100 MHz the Flicker PM noise is detected with f^{-1} slope. Third, the remaining offset frequencies show a White PM noise process.

Once the noise characteristics have been determined, it is often straightforward to deduce whether the oscillators are performing properly or not and if they are meeting the manufacturers specification.

However, the phase noise close to the carrier frequency is not characterised efficiently using the described techniques. Thus, the phase noise measurements do not give a proper characterisation about the long term stability of the oscillator. The characterisation of the long term oscillator's stability is covered in the next section.

4.3 Clock stability in the time domain

Time-domain characterisation of time and frequency stability measures the time and frequency deviation of the timing signal under test over a given interval τ.

Since instabilities in oscillators are time variations of the quantities of interest, phase, frequency, they may be characterised by a measurement of the variations that occur over a specified time interval τ [63].

Time-domain stability quantities are basically a sort of prediction of the expected time and frequency deviation over an observation interval τ. Time domain analysis is often much more efficient in providing meaningful measures of long-term performance.

Since phase and frequency of real oscillators are random processes, measurements evolve some time-averaging and evaluation is based on the statistical behaviour of the phase or frequency fluctuations as functions of time. The long term oscillators stability are evaluated by plotting the values vs τ that vary from milliseconds to days, or months.

4.3.1 Analysis of time domain data

The oscillator instantaneous frequency $\upsilon(t)$, defined in Eq. 4.2, is not observable since any frequency measurement technique does involve a finite time interval over which the measurement is performed. For example, a digital frequency counter counts the number of cycles n_k of the input signal during the time interval τ at t_k provided by its time base driven by a reference oscillator.

The minimum sample time is determined by the measurement system. If the time

difference or the time fluctuations are available then the frequency or the fractional frequency fluctuations may be calculated from one period of sampling to the next over the data length. A typical method to characterise the stability of those measurements is to calculate their standard deviation $\sigma(t)$.

However, literature have reported what happens to the standard deviation when the data set may be characterised by power law spectra, which are more dispersive than classical white noise frequency fluctuations [71]. If the fluctuations are characterised by flicker noise or any other non-white-noise frequency deviations, what happens to the standard deviation for that data set? The standard deviation is a function of the number of data points in the set; it is also a function of the dead time[†] and of the measurement system bandwidth.

Using flicker noise frequency modulation as a model, as the number of data points increases, the standard deviation monotonically increases without limit. Some statistical measures have been developed that do not depend upon the data length and that are readily usable for characterising the random fluctuations in precision oscillators [71].

An IEEE subcommittee on frequency stability has recommend what has come to be known as the "Allan Variance" [71] taken from the set of useful variances developed, and an experimental estimation of the square root of the Allan variance is shown in the next section.

4.3.2 Allan Variance

To allow standard adoption of a unique time-domain measure that can be used unambiguously in all laboratories, the IEEE subcommittee on frequency stability recommended in 1971 the use of the sample variance with M = 2 and adjacent sample with zero dead time (that is $T_s = \tau$), following the pioneering work of Allan in 1966[71]. The definition, properties, and examples of application of the Allan Variance are

[†]Dead Time is the period of time when the measurement instrument is not able to acquired data

summarised next: The Allan variance of fractional frequency fluctuations is defined by:

$$\sigma_y^2(\tau) = \left\langle \frac{\left(\overline{y}_{k+1} - \overline{y}_k\right)^2}{2} \right\rangle \tag{4.12}$$

The quantity $\overline{y}_k$ is the mean value of the fractional frequency fluctuation over the time interval τ. The definition of the variance involves the calculation of a mathematical mean over an infinite number of samples, while, of course, experiment gives a finite number of measurements. Therefore, we can only calculate an estimate of the variance; the confidence of which improves as the number of measurements is increased. The Allan variance is a theoretical measure, as based on averaging an infinite number of samples. However, it has much greater practical utility than the classical variance σ^2, because it converges for all kinds of power-law noise.

The Allan Variance equation is simple to implement as it just needs to sum and accumulate the squares of the differences between adjacent values of y_i divide by the number of them and by two, and take the square root, as is shown by Eq. 4.13.

Experimentally, only estimates of $\sigma_y^2(\tau)$ can be obtained from a finite number of samples $\overline{y}_k$, taken over a finite duration. Therefore, an inherent statistical uncertainty (usually plotted as error bars) exists when a finite number N of samples of $\overline{y}_k$ is used to estimate $\sigma_y^2(\tau)$. A widely used estimator (adopted also by international telecommunications standard bodies) is [72] ,

$$\sigma_y^2(\tau) = \frac{1}{2\left(N-1\right)} \sum_{i=1}^{N-1} \left(\overline{y}_{i+1} - \overline{y}_i\right)^2 \tag{4.13}$$

Eq. 4.13 has the quantity that the IEEE subcommittee has recommended for specification of stability in the time domain denoted by $\sigma_y^2(t)$.

This quantity is itself a random variable, whose variance (that is, the variance of the estimated variance) may be used to assess the error bars on the plot of $\sigma_y^2(\tau)$ versus τ [73]. For long-term measurements, the size of N is severely limited by the overall measurement duration.

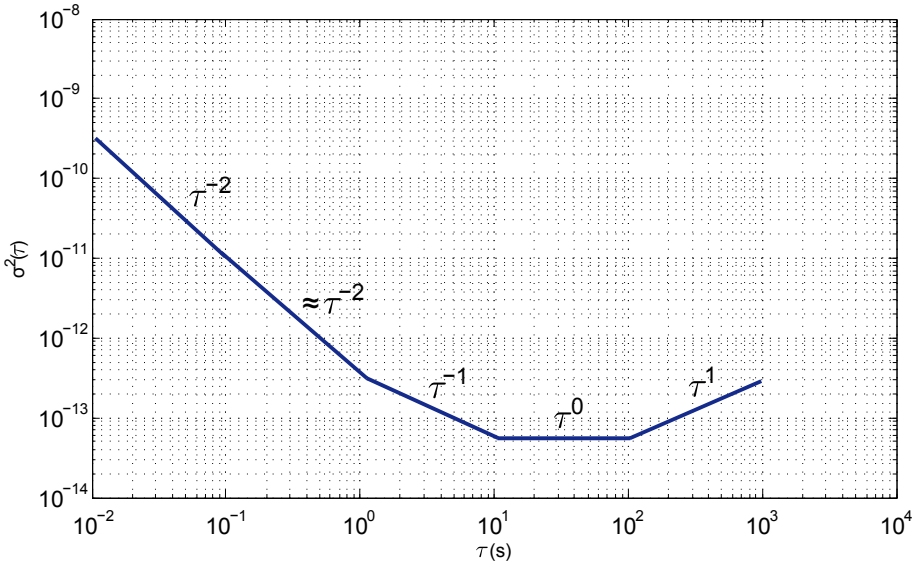


Figure 4.4: Log-log plot of the Allan variance

The Allan variance measurement can be used to identify different types of phase noise. However, White Phase Noise and Frequency Phase Noise results in very similar slopes that leads to an uncertainty of the classification of the noise when the asymptotic approximates τ^{-2}.

The Allan Variance calculation for a Caesium oscillator with a frequency centred at 10 MHz is shown in Figure 4.5. The calculation was based on fractional frequency measurements obtained directly form the Pendulum CNT-91 Timer Counter [74]. The frequency measurements are realised with a sampling time of 10 ms during a period of 5 hours.

From Figure 4.5 it is possible to observe different frequency response regions, each one showing a different type of noise power law .

The region containing WPM noise is presented for $\tau < 10s$. Between the time interval defined by $\tau > 10s$ and $\tau < 100s$ it is identified the FFM noise region. While for the region defined by $\tau > 100s$ it is found the RWFM noise region. Error bars represent the level of confidence for each Allan Variance time interval measurement. In general, along a plot of $\sigma_y^2(\tau)$ versus τ, error bars are negligible at the left (short τ) and large at the right (long τ).

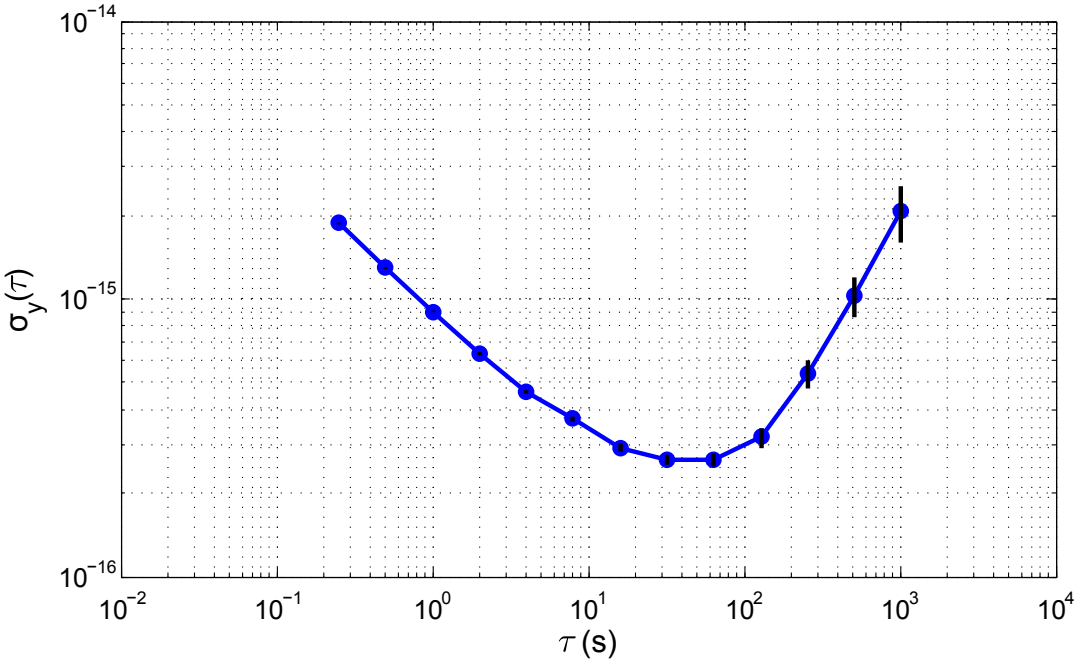


Figure 4.5: The square root of the Allan Variance of a Caesium Oscillator $\upsilon_o = 10$ **MHz**

4.3.3 Modified Allan Variance

From the original Allan variance, $\sigma_y^2(\tau)$, it is not possible to distinguish the WPM noise ($\sigma = -2$) from the FPM noise ($\sigma = -2$), since the dependence of the $\sigma_y^2(\tau)$ on the τ is the same for both these noise processes [75].

The Modified Allan Variance, Mod $\sigma_y^2(\tau)$, first presented in 1981 in [75], is another time domain measure of frequency stability define as :

$$\text{Mod } \sigma_y^2(nT_s) = \frac{1}{2}\left\langle \left(\frac{1}{n}\sum_{i=1}^{n} [\bar{y}_{i+n} - \bar{y}_i]^2 \right) \right\rangle \tag{4.14}$$

where T_s is the sampling period, the observation periods is given by $\tau = nT_s$ and the $y_i(t)$ are the fractional frequency deviation $y(t)$. From the definition above it is evident that for n=1 ($\tau = T_s$) the Modified Allan variance coincides with the Allan variance.

The Modified Allan variance includes an additional phase averaging operation, and has the advantage of being able to distinguish between white and flicker PM noise.

The Modified Allan variance is estimated from a set of M frequency measurements for averaging time $\tau = mT_s$, where m is the averaging factor and T_S is the measurement interval, by the expression:

$$\text{Mod } \sigma_y^2(nT_s) = \frac{1}{2m^4(M-3m+2)} \sum_{j=1}^{M-3m+2} \left(\sum_{j+m-1}^{i=j} \left(\sum_{i+m-1}^{k=i} [y_{k+m} - y_k]^2 \right) \right) \tag{4.15}$$

In terms of phase data, the Modified Allan variance is estimated from a set of N = M + 1 time measurements as

$$\text{Mod } \sigma_y^2(nT_s) = \frac{1}{2m^2\tau^2(M-3m+2)} \sum_{j=1}^{M-3m+2} \left(\sum_{i+m-1}^{k=i} [x_{i+2m} - x_{i+m} + x_i]^2 \right) \tag{4.16}$$

The confidence interval of a Modified Allan variance determination is also dependent on the noise type, but is often estimated as $\pm\sigma_y(\tau)/\sqrt{N}$ [69].

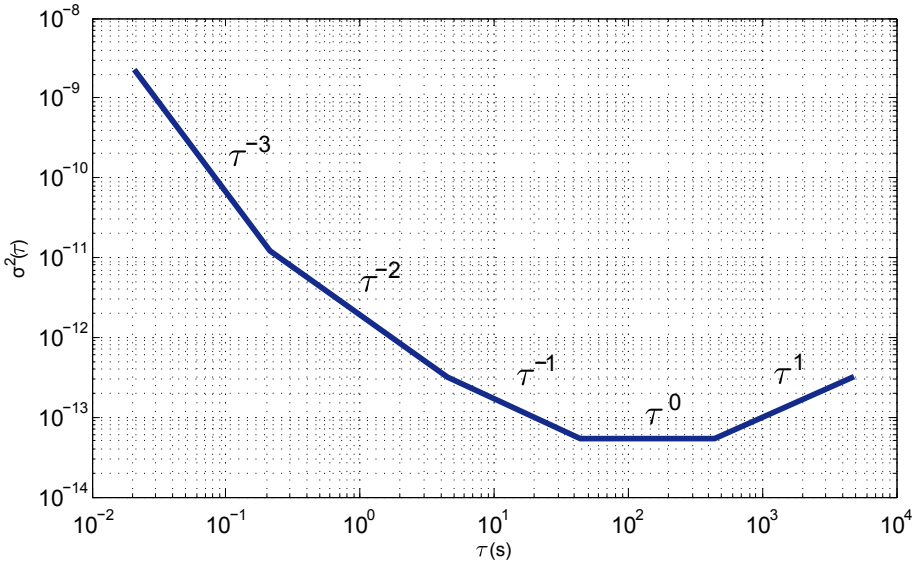


Figure 4.6: Log-log plot of the Modified Allan Variance

The Modified Allan Variance measurement on the other hand provides convergence to all types of measurements. In addition, its shows different slopes for all five noise

power types, that removes the uncertainty between White Phase Noise and Frequency Phase Noise as shown for the Allan Variance.

4.3.4 Time Variance

The time Allan variance, TVAR (or its square root TDEV), is a measure of time stability based on the Modified Allan Variance [75]. It is defined as:

$$\sigma_x^2(\tau) = \left(\frac{\tau^2}{3}\right) \bmod \sigma_y^2(\tau) \tag{4.17}$$

The time Allan variance is equal to the standard variance of the time deviations for White PM noise. It is particularly useful for measuring the stability of a time distribution network. The TVAR has been widely adopted by telecommunications international standards for the specification of timing interfaces [76].

This estimator has been defined by the ITU-T as:

$$\mathrm{TVAR}(\tau) = \sigma_x^2(\tau) = \frac{1}{6n^2(N-3n+1)} \sum_{j=1}^{N-3n+1} \left[\sum_{i=j}^{n+j-1} (x_{i+2n} + 2x_{i+n} + x_{i+}) \right] \tag{4.18}$$

where $\tau = \mathrm{n}T_s$ is the observation interval and x_i is the i^{th} of N phase values at averaging time τ.

4.3.5 Noise Relationship in the time domain

The relationship between the different PSDs $S_\varphi(f)$ and $S_y(f)$ is exact, and thus it is possible to convert between them, in both directions, without lost of information.

The stability of a frequency source can be specified and measured in either the time or the frequency domain. One domain is often preferred to specify the stability because it is most closely related to the particular application.

Similarly, stability measurements may be easier to perform in one domain than the other. Conversions are possible between these generally equivalent measures of frequency stability. Time domain frequency stability is related to the spectral density of the fractional frequency fluctuations by the relationship:

$$\sigma_y^2(\tau) = \int_0^{+\infty} S_y(f) \left|H(f)\right|^2 df \tag{4.19}$$

where $\left|H(f)\right|^2$ is the transfer function of the time domain sampling function.

The transfer function of the Allan Variance time domain stability measure is given as [62]:

$$\left|H(f)\right|^2 = 2\left(\frac{\sin^4(\pi\tau f)}{(\pi\tau f)^2}\right) \tag{4.20}$$

For the Modified Allan Variance the transfer function depends on the value m. On increasing m, the transfer function rapidly converges to:

$$\lim_{m\to+\infty} \left|H(f)\right|^2 = 2\left(\frac{\sin^6(\pi\tau f)}{(\pi\tau f)^4}\right) \tag{4.21}$$

where $\tau = mT_s$ [5].

The Eq. 4.19 can be evaluated for each noise power law and obtained a relationship for each noise power law. The various asymptotic behaviour of the different power law noise processes, in both the frequency domain and the time domain, are summarised in Table 4.1.

4.4 Jitter and Wander in synchronous networks

Jitter and Wander are phenomena that reduce the timing performance of a synchronous network.

Jitter is defined as “the short-term variations of the significant instants of a timing

Table 4.1: Relationships between frequency-domain and time-domain power laws

Noise Type	$S_y(f)$	$\sigma_y^2(\tau)$	mod $\sigma_y^2(\tau)$	TVAR(τ)
White Phase	$h_{-2}f^{-2}$	τ^{-2}	τ^{-3}	$\tau^{-1/2}$
Flicker Phase	$h_{-1}f^{-1}$	τ^{-2}	τ^{-2}	τ^{0}
White Frequency	h_0f^0	τ^{-1}	τ^{-1}	$\tau^{1/2}$
Flicker Phase	h_1f^1	τ^0	τ^0	τ^1
Random Walk Frequency	h_2f^2	τ^{+1}	τ^1	$\tau^{3/2}$

signal from their ideal positions in time (where the "short-term" implies that these variations are of frequency greater than or equal to 10 Hz)" [77].

As for wander, it is defined by "the long-term variations of the significant instants of a digital signal from their ideal position in time (where long-term implies that these variations are of frequency less than 10 Hz)" [77].

In Ethernet, the jitter generated by components must be limited and specified by [36], but the jitter transferred from one component to another is less important than for synchronous systems, because jitter can increase from component to component.

There are multiple factors that can add jitter and wander to a timing clock signal or synchronous network. One of such causes is the jitter related with the transmission of bit patterns. Certain bit patterns are more susceptible to the intersymbol interference (ISI) caused by signal distortion, essentially causing crosstalk interference between neighbouring pulses, known as pattern-dependency jitter. As mentioned in the previous chapter, Gigabit Ethernet (GbE) uses a 8b10b encoding scheme to transmit data that reduces jitter due to bit pattern.

Another source of jitter in a telecommunication network derives from crosstalk and impulse noise that produce phase variations that causes interference that leads to jitter.

An intrinsic source of jitter comes from phase noise, the transmission data clock is usually synchronised to a reference clock in synchronous networks (using PLLs); there are still phase fluctuations due to thermal noise or drift in the oscillators. The

faster phase variations caused by the clock noise leads to jitter.

Wander occurs in networks for several reasons. It is seen as low frequencies timing disturbances and the causes are usually related with temperature variations, vibrations or ageing.

Delay variations in the transmission path result in phase fluctuations, which are generally relatively slow and related with wander. Although this kind of wander is strongly present in long copper cable lines, its amplitude is far lower in optical fibre systems. Though, it may still reach several UI of amplitude if the bit rate of the transmission system is high.

Such wander cannot be reduced, as it is the result of uncontrollable changes in the cable environment. The most that can be done to reduce the effects of temperature variation is to bury the cable deep. This reduces the daily temperature variation to a fraction of a degree, although larger temperature swings will be seen annually [5].

The main reason of such wander is that the propagation delay of the light in an optical fibre depends on the refractive index of the fibre core and on the fibre length, which both depend on the temperature. Hence, the digital signals exhibit different propagation delays during the day and the night, as well as during the winter and the summer. This phenomenon yields a pseudo-periodical variation in the phase of the digital signal received, having a period of about one day or one year (diurnal and annual wander).

Even a slight variation in temperature of the optical fibre can cause a significant amount of wander over a long distance, since both the group index of refraction and the length of the fibre are temperature-dependent, this has been discussed in the previous chapter.

Fluctuations in the laser transmitter wavelength are another source of wander in fibre optic systems. Wavelength variations cause wander because the group refractive index of the optical fibre is wavelength-dependent. The drift in laser wavelength is mainly due to changes in laser temperature [52].

4.5 Summary

In this chapter the different methods used to characterise the stability and accuracy of a frequency source oscillator are discussed. The power spectrum density function of the random process defined by $\varphi(t)$ is a useful tool to understand the discriminated the different types of phase noise. The different types of phase noise are defined by the noise power law model. In addition, the time domain stability analysis of the phase noise random process was given with the definition of the Allan Variance, the Modified Allan Variance and the Time Variance. These definitions characterise the clock phase noise as a function of the averaging time. The time variance measure is mostly used to characterise the time stability of a distributed time network. Phase-noise spectrum measurements and Allan Variance of commercially available Rubidium and a Caesium atomic oscillators were made to characterise their short and long term frequency stability.

Chapter 5

Digital Circuit Design for White Rabbit

Chapter 3 described different standards and techniques employed in clock distribution applications using Ethernet. One of the standards described was the IEEE 1588 that uses packet based messaging to synchronise nodes in a local network. Another standard presented was the Synchronous Ethernet (ITU-T G.6223), which recommends a set of rules to distribute synchronisation from node-to-node using Ethernet's physical layer. These standards are implemented in the White Rabbit network to assist in the sub-nanosecond timing accuracy. Nonetheless, in Gigabit Ethernet the clock time granularity is 8 ns, defined by the 125 MHz clock used to encode data in the Ethernet link. To achieve the timing accuracy of the WR network, is required to measure the phase difference between the transmitted and received clock and to phase shift the slave clock so that both clocks are aligned. This process, already described in Chapter 3, compensates for the variations of phase in the recovered clock when external factors are acting on the optical link.

Unfortunately, current digital clock phase shifters with picosecond time resolution require very high sampling rate, in the order of GHz, because they rely on gate counters to measure phase difference between the two clock signals [78]. The biggest disad-

vantage of such systems is the need for a very high frequency counter to measure the phase between the input clocks, this becomes especially challenging in applications that require high phase accuracy in a high frequency clock signal. For instance, if the input clock signal has a frequency of 125 MHz and is required to phase shift the input clock by 1^o, equivalent to time shift the clock by 22 ps, then it would be necessary to have a frequency counter with at least 45 GHz sampling clock, which is unrealistic.

Another method to digitally phase shift a clock signal is based on sampling the timing signal using A/D converters [79]. After the analogue to digital conversion, the signal is digitally processed using DSP techniques to interpolate the phase difference between the two clock signals. However, this method suffers from the non-linear behaviour of the A/D converters, and also from the high sampling frequency required to sample high frequency clock signal.

This chapter proposes a novel digital circuit capable of picosecond time resolution and clock phase shifting using a relatively low system clock, in the order of hundreds of MHz. The chapter starts with a detailed description of the analogue dual mixer time difference circuit. This analogue circuit, first presented during the middle of the Nineteen Seventies, aims to characterise frequency clock sources with very high time resolution using commercial low frequency digital counters.

Next, Section 5.2 is devoted to the description of the Digital Dual Mixer Time Difference (DDMTD), a novel circuit proposed by the author, capable of picosecond time resolution. This digital circuit has the particular advantage of being able to be implemented in an FPGA system. In this section, the model of the DDMTD is derived, followed by a detailed analysis of the glitches that occur on the output of the DDMTDs clocks.

In section 5.3, a set of deglitching techniques that filter and remove the glitches in the DDMTD clocks are also proposed. The proposed deglitching algorithms are tested in a simulation environment and the performance of each algorithm is presented and discussed.

The work presented in Chapter 5 has published in:

- P. Moreira, P. Alvarez-Sanchez, J. Serrano, I. Darwazeh, and T. Wlostowski. **Digital dual mixer time difference for sub-nanosecond time synchronization in Ethernet**. IEEE International Frequency Control Symposium, June 2010
- P. Moreira and I. Darwazeh. **Digital femtosecond time difference circuit for CERN's timing system**. London Communications Symposium, September 2011.
- P. Moreira, P. Alvarez, J. Serrano, I. Darwazeh **Sub-Nanosecond Digital Phase Shifter For Clock Synchronization Applications**. IEEE International Frequency Control Symposium, June 2012.

5.1 Dual Mixer Time Differences Circuit

The Dual Mixer Time Difference (DMTD) circuit, first presented by David W. Allan [80], is an analogue circuit that compares two clock signals in order to obtain their frequency and phase deviation. In most of its applications, the DMTD is used to compare a test clock signal with a reference clock to measure its frequency stability, which can be used for instance to plot the Allan Variance of the reference clock under test.

The DMTD circuit is one of the techniques that can measure a time difference with high resolution using a relatively low frequency commercial time interval counter [80]. The DMTD circuit schematic [80] is illustrated in Figure 5.1.

The reference clock and the local clock are represented by $u_1(t)$ and $u_2(t)$, respectively. $\theta_a(t)$ and $\theta_b(t)$ represent the phase deviation of the clock signals $u_1(t)$ and $u_2(t)$, respectively. The term v_n is the fundamental frequency of both clocks.

The time difference, $\Delta_t(t)$, measured by the DDMTD circuit is defined as:

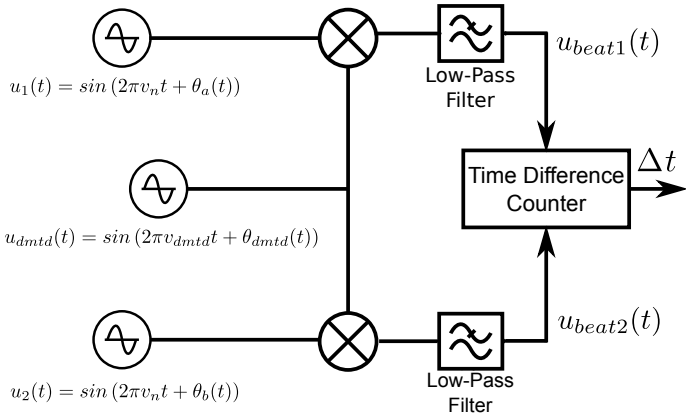


Figure 5.1: DMTD circuit schematic based on [80]

$$\Delta t(t) = \frac{\Delta\theta(t)}{2\pi v_n} \tag{5.1}$$

where $\Delta\theta(t) = \theta_a(t) - \theta_b(t)$.

In order to down-convert the frequency v_n to v_{beat} and maintain the same phase information of the input clocks, a local clock signal with frequency v_{dmtd} is required. The resulting beat-down frequency is defined as $v_{beat} = v_n - v_{dmtd}$. The down conversion is realised by multiplying the signals $u_1(t)$ and $u_2(t)$ by the signal $u_{dmtd}(t)$. The mixing operation is described as follows:

$$\begin{aligned} u_1(t) \cdot u_{dmtd}(t) &= sin\,(2\pi t v_n + \theta_a(t)) \times sin\,(2\pi t v_{dmtd} + \theta_{dmtd}(t)) \\ &= \frac{1}{2} cos\,(2\pi t\,(v_n + v_{dmtd}) + \theta_a(t) + \theta_{dmtd}(t)) \\ &\quad + \frac{1}{2} cos\,(2\pi t\,(v_n - v_{dmtd})) + \theta_a(t) - \theta_{dmtd}(t)) \end{aligned} \tag{5.2}$$

The first term of Eq. 5.2, composed by high frequency components, is removed by a low pass filter, leaving solely the second term. Likewise, the mixing between $u_2(t)$ and $u_{dmtd}(t)$ produces a similar result, however, with a phase $\theta_b(t)$ instead of $\theta_a(t)$. The low pass filter output signals are:

$$u_{beat1}(t) = \frac{1}{2}cos\left(2\pi t\left(v_{beat}\right)\right) + \theta_a - \theta_{dmtd}) \tag{5.3}$$

$$u_{beat2}(t) = \frac{1}{2}cos\left(2\pi t\left(v_{beat}\right)\right) + \theta_b - \theta_{dmtd}) \tag{5.4}$$

Subtracting the phase term between Eq. 5.3 and Eq. 5.4, results in:

$$\begin{aligned}\Delta\theta(t) &= \left(2\pi t\left(v_{beat}\right)\right) + \theta_a(t) - \theta_{dmtd}(t)) \\ &\quad - \left(2\pi t\left(v_{beat}\right)\right) + \theta_b(t) - \theta_{dmtd}(t)) \qquad (5.5) \\ &= (\theta_a(t) - \theta_b(t)) \qquad (5.6)\end{aligned}$$

The mixing process down-converts the frequencies of the input clock signals. Nonetheless, is verified by Eq. 5.6 that the phase difference, $\Delta\theta(t)$, between the two clock signals is kept unchanged before and after the mixing operation. The phase difference between $u_1(t)$ and $u_2(t)$ can be estimated by measuring the time difference between the zero crossing transitions of the down-converted clocks.

$$\Delta t_{beat}(t) = \frac{\Delta\theta(t)}{2\pi v_{beat}} \tag{5.7}$$

Accordingly, $\Delta t(t)$ is calculated as follows:

$$\Delta t(t) = \Delta t_{beat}(t)\frac{v_{beat}}{v_n} = \frac{\Delta\theta(t)}{2\pi v_n} \tag{5.8}$$

Thus, by using the DMTD circuit, the resolution of the phase measurement is improved by a factor (v_n/v_{beat}).

The DMTD circuit has excellent time resolution in measuring frequency deviation between continuous sinusoid wave clock signals [81].

However, some limitations arise when the application requires the continuous evalu-

ation of phase difference between two digital timing signals with a squared form [82]. The DMTD only performs correct time difference measurements if the input signals are sinusoids. If the clock signals are squared this measurement circuit cannot be used to measure its phase difference. Another disadvantage of the DMTD circuit is the usage of the two mixers, which are bulky, increase board size and add non-linearities errors to the mixing process and required special care to be minimised [83]. To implement the mixing process digitally the numerical oscillator would require too many digital resources in the integrated circuit from [84]. To measure the phase difference between two squared clocks signals efficiently in digital hardware, a novel digital version of the DMTD circuit is proposed in the following section.

5.2 Digital Dual Mixer Time Difference Circuit

The Digital Dual Mixer Time difference (DDMTD) is a custom-made digital circuit designed, in the scope of this work, to measure with picosecond time resolution the phase difference between two squared clock signals, used in digital applications. In addition, the DDMTD circuit, whose circuit schematic is shown in Figure 5.2, has the advantage of being fully implemented in digital integrated circuit such as FPGAs, increasing its system integration and modularity.

The DDMTD has two D Flip-Flops (DFF) instead of the two analogue mixers presented in the original design. The proposed circuit also has a deglitching circuit, which is further described later in this section.

The input clock signals are squared clock signals, derived from a sine signals, with the form:

$$u_1(t) = \begin{cases} 1, & \sin\left(2\pi v_n t + \theta_a(t)\right) > 1 \\ 0, & \sin\left(2\pi v_n t + \theta_a(t)\right) \leqslant 0 \end{cases} \tag{5.9}$$

and,

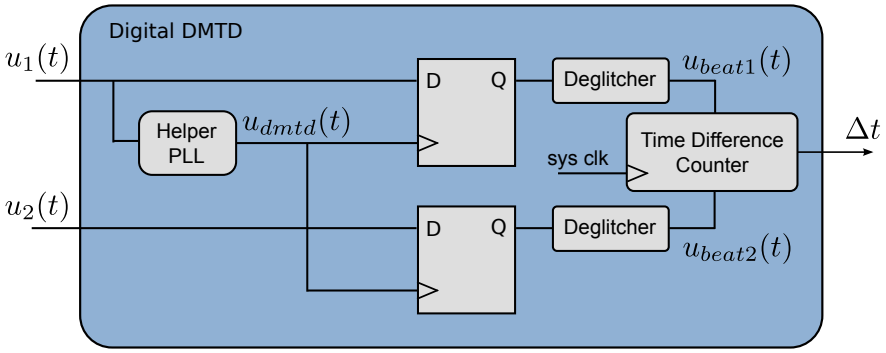


Figure 5.2: Proposed Digital DMTD Circuit

$$u_2(t) = \begin{cases} 1, & \sin(2\pi v_n t + \theta_b(t)) > 1 \\ 0, & \sin(2\pi v_n t + \theta_b(t)) \leqslant 0 \end{cases} \tag{5.10}$$

where v_n is the inputs clocks fundamental frequency, θ_a and θ_b are the phase of $u_1(t)$ and $u_2(t)$, respectively.

The input clock signals are digitally mixed, using a pair of DFFs, with another clock signal generated by the Helper PLL. The DFF outputs a down-converted clock signal, defined as u_{beat}, with a fundamental frequency v_{beat}. The v_{beat} is equal to the frequency difference between the input clock signal and the local oscillator frequency.

The DDMTD does not require a low-pass filter after the mixing process, as its analogue version, because the high frequency components are removed by the DFF behaviour. However, in the case where very high time resolution, in the order of picosecond, is required, the DDMTD circuit needs a deglitching circuit to remove the glitches that appears on the transitions of the $u_{beat}(t)$ signal. These glitches occurs due to the presence of phase noise in the input clock signals and the DFFs metastability phenomena [85]. This phenomena is intrinsic to all digital components and is caused by the small sweep in frequency that might violate the setup and hold time requirements of DFFs.

The local oscillator, which is used in the mixing process to down-convert the input clocks, is a clock synthesised by block Helper PLL, as shown in Figure 5.2. The

Helper PLL generates the $u_{dmtd}(t)$ clock signal, whose fundamental frequency, v_{dmtd}, is defined has:

$$v_{dmtd} = \frac{N}{N+1} v_n \tag{5.11}$$

where N is an integer number and v_n the fundamental frequency of the reference clock. The higher the N parameter is, the closer is the frequency v_{dmtd} to v_n, resulting in a smaller v_{beat} frequency.

A time diagram illustrating the behaviour of the DDMTD, for a N = 5, is shown in Figure 5.3.

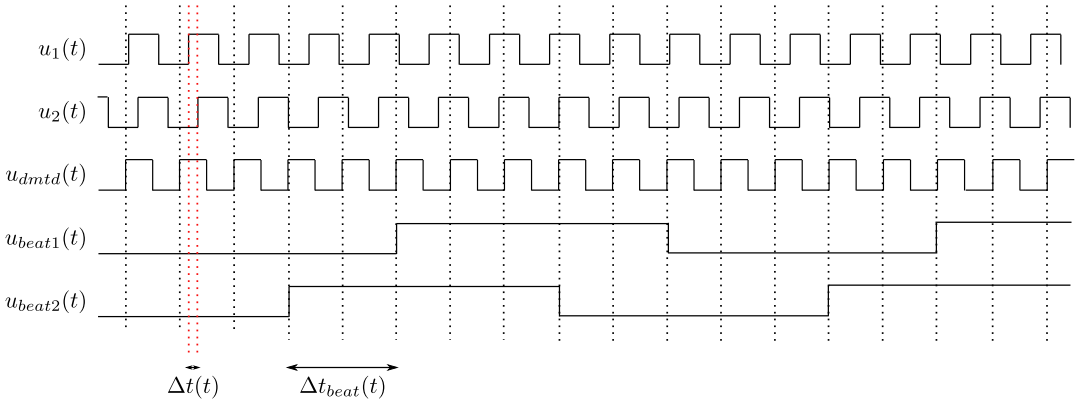


Figure 5.3: DDMTD time diagram for N = 5

Represented in Figure 5.3 are $u_1(t)$ and $u_2(t)$ the input clocks, $v_{dmtd}(t)$ the clock signal generated by the Helper PLL, $u_{beat1}(t)$ and $u_{beat1}(t)$ the DFFs output clock signal. The quantity $\Delta t(t)$ is the time difference between the input clocks $u_1(t)$ and $u_2(t)$,as previously defined. While $\Delta t_{beat}(t)$ is the time difference between the signals $u_{beat1}(t)$ and $u_{beat2}(t)$. These two quantities are proportional to each other by a factor $\frac{v_{beat}}{v_n}$.

The time difference, Δt, between the input clocks $u_1(t)$ and $u_2(t)$, is given as:

$$\Delta t(t) = \Delta t_{beat} \frac{v_{beat}}{v_n} \tag{5.12}$$

The down-conversion process implemented by the DFFs occurs during the positive edge transition of the u_{dmtd} clock signal. That is, the output DFF can change at every $1/v_{dmtd}$ seconds. This period represents the minimal time resolution, $\Delta\varepsilon$, for which the DDMTD circuit can measure the time difference, and is described as:

$$\Delta\varepsilon = \frac{1}{v_{dmtd}} \frac{v_{beat}}{v_n} = \frac{v_n - v_{dmtd}}{v_{dmtd} \cdot v_n} \tag{5.13}$$

Substituting Eq. 5.11 by Eq. 5.13 results in:

$$\Delta\varepsilon = \frac{v_n - \frac{N}{N+1} v_n}{\frac{N}{N+1} v_n \cdot v_n} = \frac{1}{N \cdot v_n} \tag{5.14}$$

Eq. 5.14 shows the relationship between the Helper PLL N parameter, used to synthesise the $u_{dmtd}(t)$ clock signal and the frequency of the input signal v_n to know minimal time resolution, $\Delta\varepsilon$, of the DDMTD. In order words, the closer, but not equal, is the helper frequency v_{dmtd} to the input clock frequency v_n, i.e. as N increases, the lower is time difference error measured by DDMTD circuit.

Figure 5.4 shows the relationship between the time resolution of the DDMTD with the circuit's beat frequency, for an input signal v_n = 125 MHz that is the recovery frequency signal of the White Rabbit network.

As it can be seen in Figure 5.4, the DDMTD circuit achieves very high time difference resolutions, below the picosecond mark for a v_{beat} less than 20 kHz. For v_n = 125 MHz and a v_{beat} = 1 Hz, that is the v_{dmtd} = 125 000 001 Hz, the time difference resolution measured by the DDMTD is ~0.1 fs. However, an increase in the time difference resolution comes with an increase of noise (glitches) in the edge transitions of the u_{beat} signals.

The occurrence of the glitches is mainly due to the timing jitter in the input clock signal, but also due to the metastability behaviour of the digital components. The phase noise of the input clock and the Helper PLL clock is sampled by the DDMTD DFFs during the digital down-conversion process. A phase noise increase in the

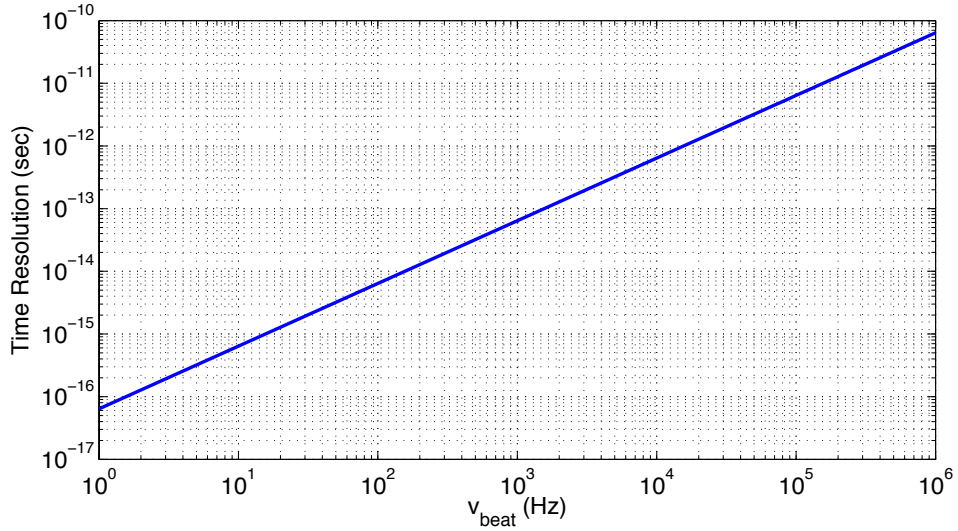


Figure 5.4: Time resolution vs v_{beat} for a v_n=125 MHz

DDMTD input clocks results in an increase of the transitory period where the glitches appear.

Glitches must be removed or filtered and their presence should be minimised to increase the accuracy of the DDMTD circuit. This is done by adding a deglitching module that evaluates the output of the DFFs for glitches and removes the glitches in the outputted clock signal. Thus, prevents erroneous time difference measurements from the time counter, and increases the accuracy of the system.

To illustrate, Figure 5.5 shows a time diagram describing the occurrence of glitches. The duration of a glitch has a minimal value of $1/v_{dmtd}$, the sampling period of the DFFs.

Both $u_1(t)$, $u_2(t)$ and $u_{dmtd}(t)$ are clock signals that like every clock have timing jitter. The clock jitter is illustrated in Figure 5.5 by a grey shadow surrounding the clock edge transitions.

The $u_{dmtd}(t)$ clock signal does not show a grey shadow area for simplicity purposes.

Nonetheless, the $u_{dmtd}(t)$ signal has timing jitter in its edge transitions, which is used to sample of the input of the DFF. So the contribution for the glitching time width is due to both input signal and the sampling clock signal.

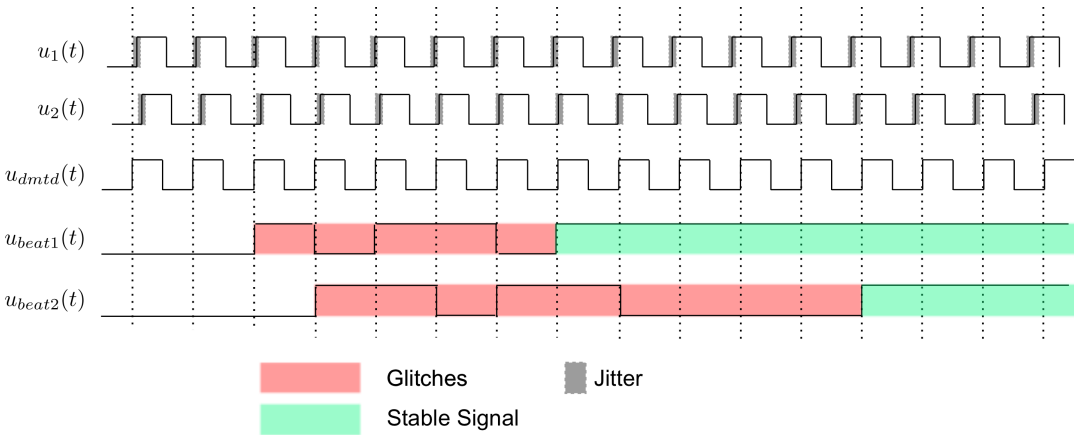


Figure 5.5: DDMTD Glitches

During the $u_{dmtd}(t)$ edge transition the DFF samples its D input, in the case the input is at logic level "1" the DFF's output changes to "1". Inversely, if the D-input shows a logic level "0" the output changes to "0". However, during the input transition the D-input of DFF may be at logic level "1" while in the next sampling edge it may be, (influenced by the clock jitter) at the logic level "0". This signal logic variation in the D-input is seen in the output of the DFF as glitches.

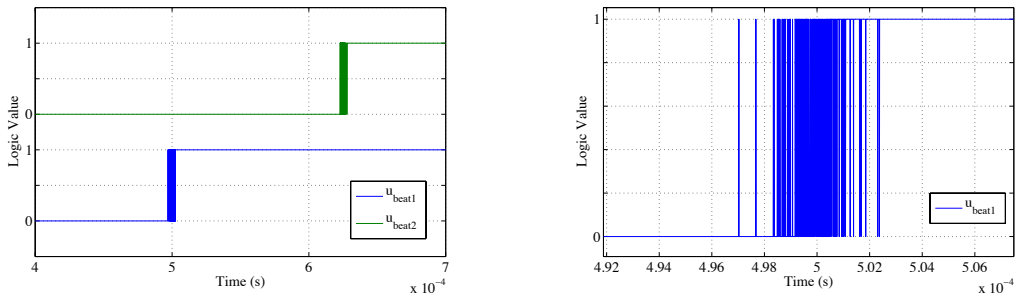


Figure 5.6: Digital DDMTD glitches

Figure 5.6 shows a simulated behaviour of the DDMTD circuit during the occurrence of glitches. Removing the glitches by selecting the most accurate edge transition is the subject of discussion of the following section.

5.3 Deglitching Techniques

Due to the presence of glitches in the output of the DFFs, the time difference counter may do erroneous measurements of the time difference between the two input clocks. For this reason, it is necessary to analyse the output of the DDFs to filter and remove the occurrence of glitches. Several deglitching techniques are discussed and their performance, measuring the correct time difference between the two input clocks is evaluated through a computer-based simulation environment implemented in MATLAB. A restriction in the design of these deglitching techniques is that they need to be implemented in an FPGA integrated circuit without excessive use of hardware resources.

The proposed deglitching techniques are listed as follows:

First Edge Selection

The First Edge Selection deglitching technique selects the first edge transition as a good edge for the time difference counter. The glitches that might occur after the first one are discarded. Figure 5.7 illustrates the output behaviour of this deglitching technique during the occurrence of glitches in the u_{beat} clock signal.

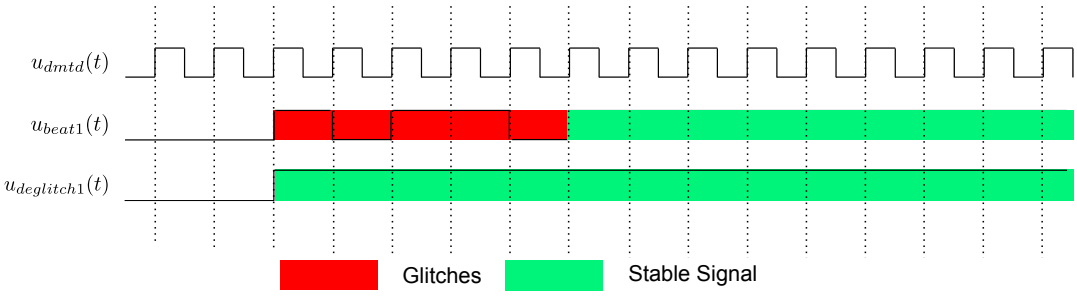


Figure 5.7: First Edge Selection Time Diagram

The implementation of the First Edge Selection deglitching technique is very simple and it requires low percentage of available hardware resources in a FPGA implementation. Figure 5.8 shows a flowchart describing the functional behaviour of this technique.

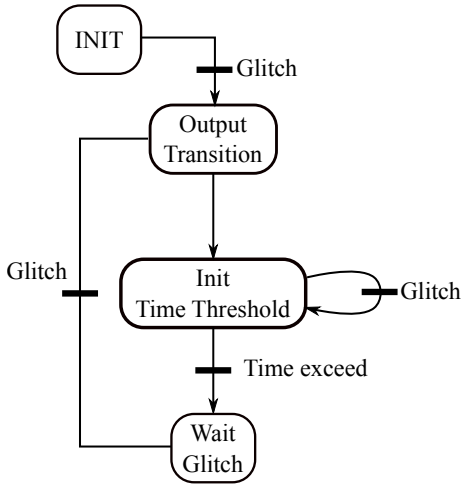


Figure 5.8: First Edge Selection Flowchart

On power-on or reset the system state goes to INIT. In the INIT state, the deglitcher samples the $u_{beat}(t)$ clock line for an edge transition or glitch. Upon the occurrence of the first edge transition, the algorithm assumes that edge as the correct and the edge is timestamped. After this first transition, a timer starts running, in the event of other glitches the timer is reset. When the timer counter reaches its time-out value, it should not occur any other glitches and the state changes to Wait Glitch. In the Wait Glitch state, the deglitcher samples the clock line for the next edge transition, and the process is repeated. The selection of the timer threshold was 25 % of the $u_{beat}(t)$ period, which is known before hand. For instance, if the $u_{beat}(t)$ has a period of 100 ms the timer threshold is set to be 25 ms.

Mean Edge Selection

The second deglitching technique proposed is the Mean Edge Selection. This technique timestamps the occurrence of the positive edge transitions of glitches in the u_{beat} clock line. The best edge selection is selected based the calculation of the mean edge position among all glitches. The hardware implementation is described as follows: After an edge transition, a timer is reset. When the running timer counter reaches its time-out value, it processes all the timestamps edge saved and selects the edge transition that is in the mean position. The deglitching technique described is

illustrated in Figure 5.9.

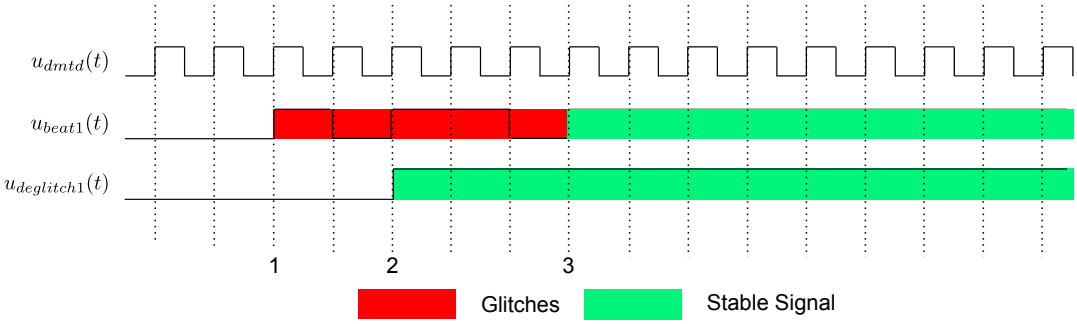


Figure 5.9: Mean Edge Selection Time Diagram

The Mean Edge Selection deglitching technique requires more memory resources than the First Edge deglitching technique, mainly due to the fact that this deglitching technique has to save all the glitches' timestamps.

The implementation of Mean Edge Selection deglitching technique is illustrated by the flowchart in Figure 5.10.

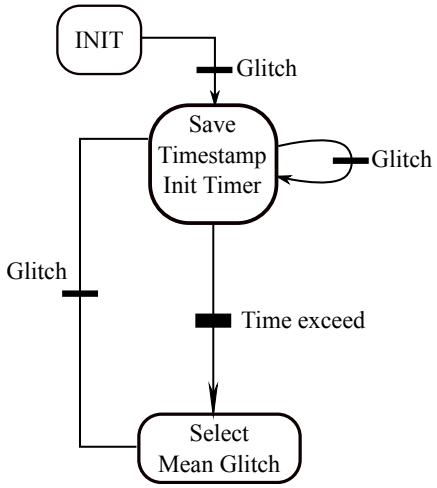


Figure 5.10: Mean Edge Selection Flowchart

From the initial state the clock line is continuously being sampled to detect an edge transition or a glitch. When a glitch is detected its timestamp is saved and the auxiliary time counter is initialised. In the occurrence of more glitches, the previous process is repeated. When the time counter threshold is surpassed, the system selects among the set of timestamps, the mean timestamp glitch to be the edge transition.

As for First Edge Selection, the selection for the time counter threshold was selected to be 25 % of the u_{beat} period.

Zero Count Selection

The last deglitching technique proposed selects as the output edge transition, among the set of glitches, the median value between the number of logic values "0" and the logic value "1". That is, the zero count deglitching technique samples the $u_{beat}(t)$ clock line and adds to a defined bin a sample that is either a "0" or a "1" depending on the $u_{beat}(t)$ logic level, as depicted in Figure 5.11. When the $u_{beat}(t)$ clock line reaches a stable region, guaranteed by a consecutive number of constant samples, the position where the number of "0" is equal to the number of "1" is considered to be the most accurate edge transition.

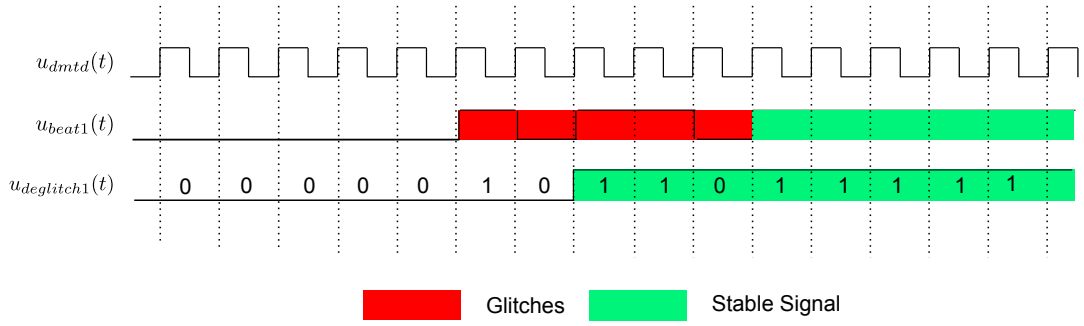


Figure 5.11: Zero Count Selection Time Diagram

This technique does not timestamp all the edge transition occurrences, instead every sample of a "0" and "1" is cancelled with each other until a position where no more zeros and ones occur. That time median position is considered most accurate edge transition.

A flowchart illustrating the implementation of the Zero Count is shown in Figure 5.12. Upon a reset signal the algorithm starts to sample the u_{beat} and counts the amount of "0" and "1". As soon as the number of stable "1"s reaches the same number of stable "0"s, the algorithm starts to calculate the median value between "0" and "1". The position calculated is defined as the u_{beat} edge transition.

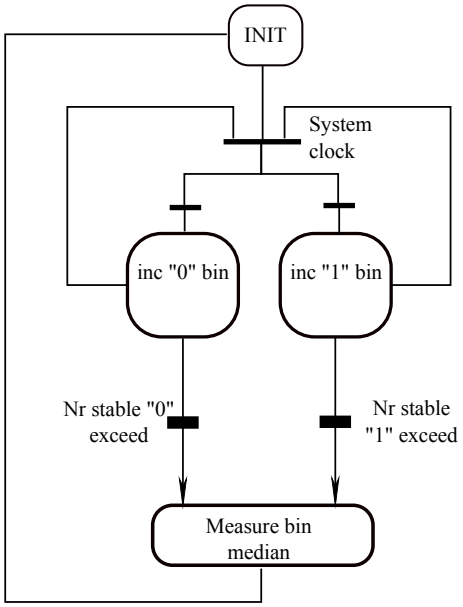


Figure 5.12: Zero Count Selection Flowchart

5.3.1 Simulation Environment

In this section, the performance of the different deglitching techniques is analysed with a MATLAB simulation environment.

The input clock signals have the same fundamental frequency of the Ethernet link recovery clock that is $v_n = 125$ MHz.

Gaussian noise is added to the phase of the input clocks to generate timing jitter. This method allows the analysis of the deglitching techniques for different ranges of jitter in the input clock signals.

The simulations aim to analyse the performance of the DDMTD circuit for the different deglitching techniques in selecting the most accurate edge transition. The performance is measured for clock signal with different v_{beat} frequencies, and quantities of timing jitter.

5.3.2 Results

MATLAB simulations are realised to investigate the performance of the different deglitching techniques of a set of scenarios. The scenarios consist in changing the beat frequency, v_{beat}, from 100 Hz to 10 MHz. In addition, added to the clock signal are different amounts of timing jitter, varying from 1 ps to 100 ps.

The phase difference between the two input clocks is set to $\pi/4$ rads, which results in a time difference of 1 ns between the two input clocks.

The simulations run until the DDMTD has a number of 20000 time difference samples. This is equivalent to run the simulation for 200 seconds when the v_{beat} is set to 100 Hz or to 2000 μs when the v_{beat} is set to 10 MHz. The results are presented as follows.

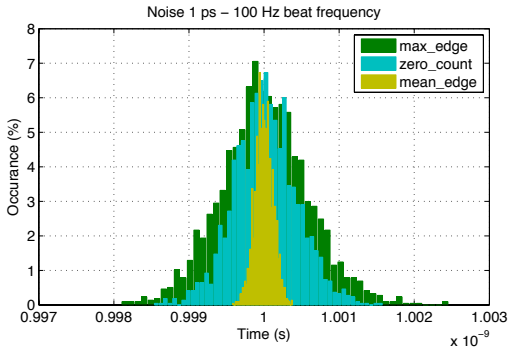
The first set of measurements are obtained for a beat frequency of 100 Hz, that is a time difference sample is measured every 10 ms. The DDMTD time resolution for a beat frequency of 100 Hz is 6.4 fs.

Figure 5.13 shows a set of histograms with the time difference measurements obtained by the three different deglitching algorithms.

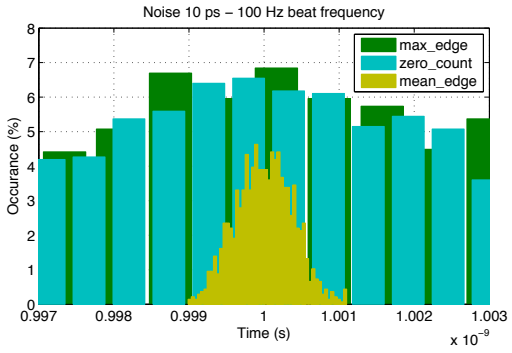
The histograms have the same x-axis scale to give a better comparative analysis among the results obtained for the various timing jitter added to the signal. The histograms show that among the deglitching techniques proposed, the one that is more accurate in the phase difference measurement is the Mean Edge. This is because its time difference measurements are closer to the reference value than the others ones. The second is the Zero Count followed by the Max Edge.

The second set of results is obtained for beat frequency of 1 kHz, which results in a time resolution for the DDMTD of 64 fs. The histograms of the time differences measurements are shown in Figure 5.14 for the different jitter levels.

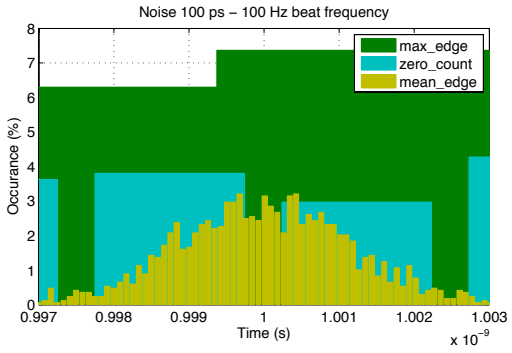
The histograms show a wider spread of the time different measurements when comparing with the measurements obtained earlier for a v_{beat} of 100 Hz. This is expected as in this configuration the DDMTD time resolution is decreased. However, increas-



(a) σ_{jitter}=1 ps v_{beat} = 100 Hz



(b) σ_{jitter}=10 ps v_{beat} = 100 Hz

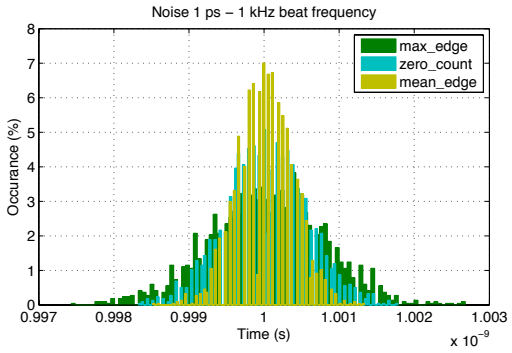


(c) σ_{jitter}=100 ps v_{beat} = 100 Hz

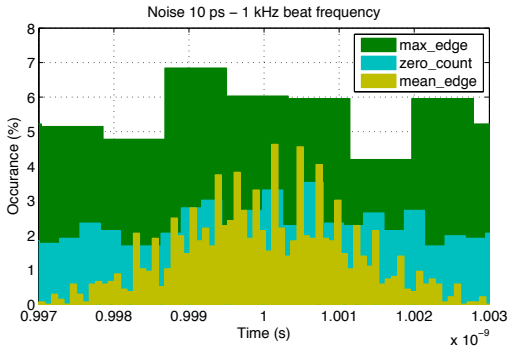
Figure 5.13: Deglitching techniques at v_{beat} = 100 Hz

ing the timing jitter in the input clocks the different deglitching techniques show an identical performance. For the time resolution shown in Figure 5.14 the Mean Edge deglitching technique still improves the accuracy of time difference measurement.

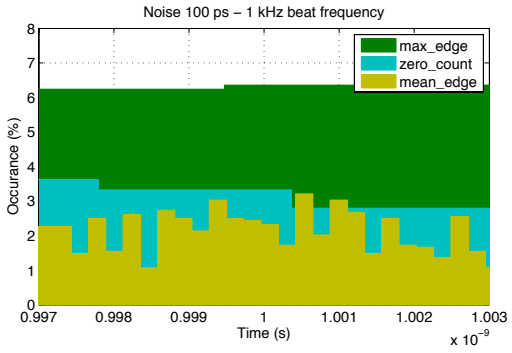
The next set of measurements, shown in Figure 5.15, are for a beat frequency of 10 kHz. The DDMTD's time difference resolution for the 10 kHz beat frequency is 640 fs.



(a) σ_{jitter}=1 ps v_{beat} = 1kHz



(b) σ_{jitter}=10 ps v_{beat} = 1kHz

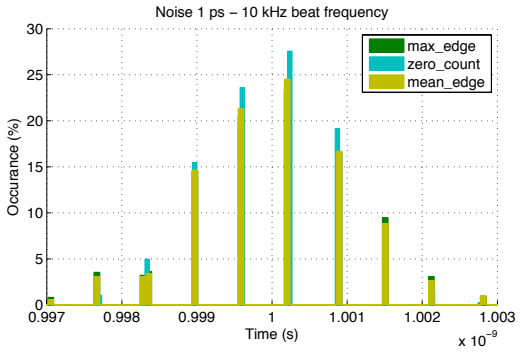


(c) σ_{jitter}=100 ps v_{beat} = 1kHz

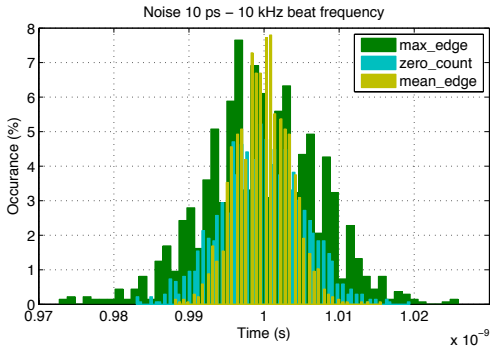
Figure 5.14: Deglitching techniques at v_{beat} = 1 kHz

With the increase of the v_{beat} frequency to 10 kHz, and for a clock jitter of 1 ps, the three deglitching techniques show an identical performance. The space visible between the measurements bins, seen in Figure 5.15(a), is exactly the current time resolution of the measuring circuit.

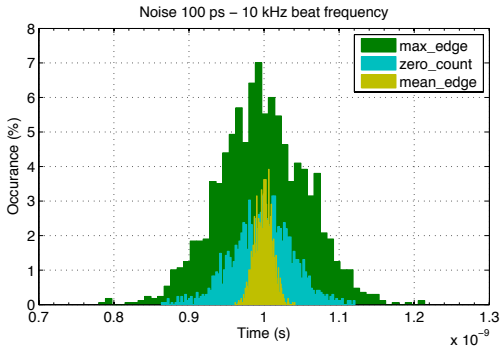
For a higher input clock's jitter the performance for the different deglitching techniques is different. In this case, Figure 5.15(b), the Mean Edge technique shows a



(a) σ_{jitter}=1 ps v_{beat} = 10 kHz



(b) σ_{jitter}=10 ps v_{beat} = 10 kHz



(c) σ_{jitter}=100 ps v_{beat} = 10 kHz

Figure 5.15: Deglitching techniques at v_{beat} = 10 kHz

narrow spread of the time difference measurements indicating a better performance.

Figure 5.16 shows the DDMTD simulations for a beat frequency of 100 kHz. The DDMTD time resolution for this beat frequency is 6.4 ps.

The histograms in Figure 5.16(a)(b) show that for a clock with jitter between 1 ps and 10 ps the tested deglitching techniques do not show an improvement in the performance of the time difference measurements.

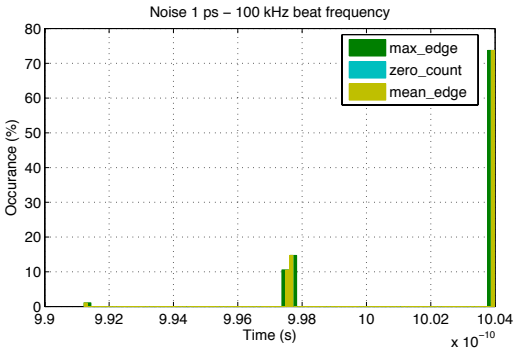
In the case of the 1 ps jitter the Gaussian shape almost vanished. This is an expected result, as for a time resolution of 6.4 ps the 1 ps second jitter is like if it does not exist from the point of view of the DDMTD circuit.

Only in the presence of a clock's jitter higher than 100 ps it reasonable to apply, to the DDMTD circuit, the Mean Edge or the Zero Count techniques.

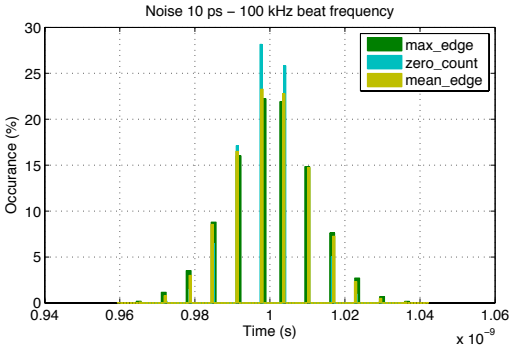
Figure 5.16(c) shows a slight improvement in the precision of the time difference measurements for the Mean Edge deglitching technique.

To conclude this set of measurements, simulations are presented for beat frequencies of 1 MHz and 10 MHz, the time resolution is 64 ps and 640 ps, respectively.

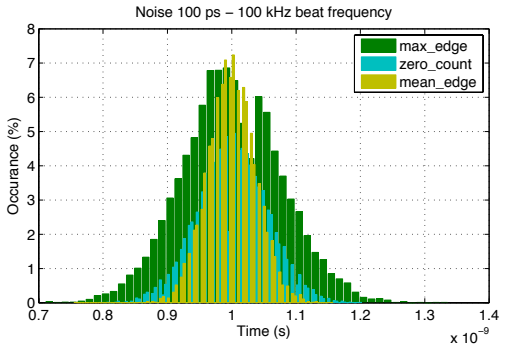
Figures 5.17 and 5.18 show that for these beat frequencies (which results in a low time difference resolution) an implementation of a deglitching technique to improve the accuracy of the measurements is unnecessary. Nonetheless, even in the cases where the DDMTD's time resolution is low the occurrence of glitches can always occur, although it probability diminishes with the increase of the beat frequency. For these cases the Max Edge technique should be used to remove the glitches, because it has the same performance of the others two deglitching technique, however, it requires less hardware resources in its implementation.



(a) σ_{jitter}=1 ps v_{beat} = 100 kHz

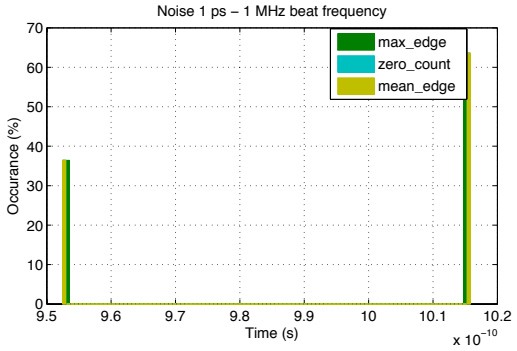


(b) σ_{jitter}=10 ps v_{beat} = 100 kHz

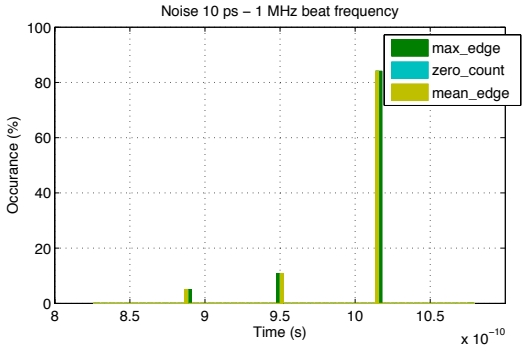


(c) σ_{jitter}=100 ps v_{beat} = 100 kHz

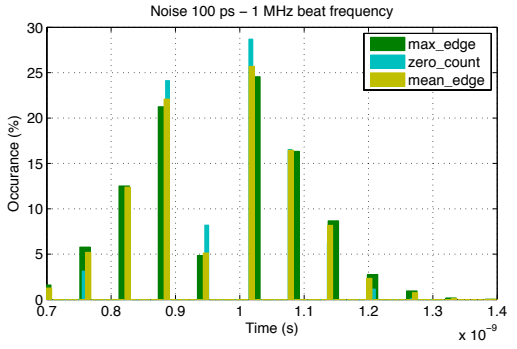
Figure 5.16: Deglitching techniques at v_{beat} = 100 kHz



(a) σ_{jitter}=1 ps v_{beat} = 1 MHz

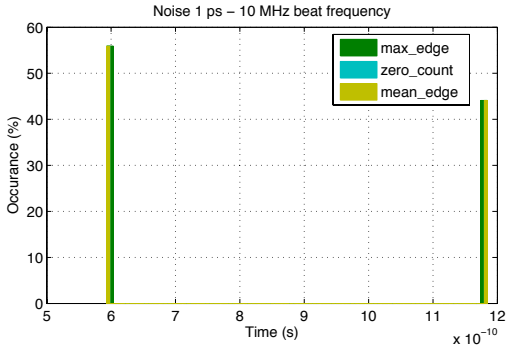


(b) σ_{jitter}=10 ps v_{beat} = 1 MHz

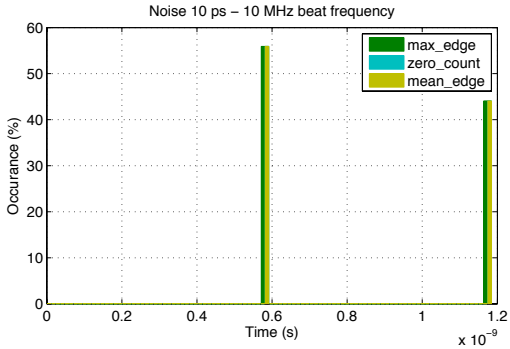


(c) σ_{jitter}=100 ps v_{beat} = 1 MHz

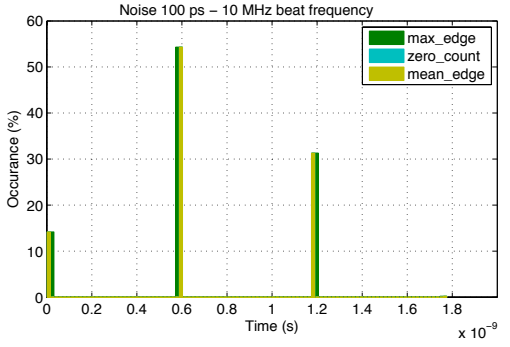
Figure 5.17: Deglitching techniques at v_{beat} = 1 MHz



(a) σ_{jitter}=1 ps v_{beat} = 10 MHz



(b) σ_{jitter}=10 ps v_{beat} = 10 MHz



(c) σ_{jitter}=100 ps v_{beat} = 10 MHz

Figure 5.18: Deglitching techniques at v_{beat} = 10 MHz

5.4 Summary

In this chapter, a novel time difference digital circuit capable of sub-picosecond time resolution is proposed. This circuit is required to measure the time difference between the transmitted and recovered Ethernet clock. In addition, the circuit is used to phase shift the system clock so that it is in phase with its master and the slave nodes are able to compensate their timing for daily phase variations in the data transmission of the link and achieve the required sub-nanosecond accuracy specification of the WR network.

The accuracy of the proposed digital time different circuit is characterised by glitches during the edge transition of the DFF. The glitches occur when the DDMTD is configured to measure the time difference between two digital clocks with high time resolution. The glitches are due to timing jitter in the input clock signals. To mitigate the occurrence of glitches, a set of deglitching techniques are proposed. They aim at filtering and removing the appearance of glitches. To differentiate the performance between the proposed deglitching techniques a computer-based simulation environment was developed in Matlab. The performance for each deglitching technique is characterised for different input parameters such as beat frequency and amount of timing jitter in the input clocks. For v_{beat} higher than 1 MHz, that is time resolution higher than 64 ps, the difference between the proposed deglitching is not noticeable. However, for v_{beat} frequencies lower that 100 kHz, that is time resolution lower than 6.5 ps, an increase in the accuracy measured by the Mean Edge can be observed. The Zero Count algorithm shows similar performance with the Mean Edge algorithm for high time resolution. Nonetheless, the Zero Count algorithm requires less hardware in its implementation when compared with the Mean Edge and should be implemented in applications that required high time resolutions. For the remaining applications, due to its simplicity the Max Edge should be used.

Next chapter discusses the FPGA's implementation of the DDMTD circuit in a PLL design with picosecond time resolution phase shifting capabilities.

Chapter 6

Implementation of the DDMTD

This chapter presents the FPGA-based implementation of the DDMTD circuit described in Chapter 5. The aim of this implementation is to verify the performance of the DDMTD in two specific tasks: One is to use the DDMTD as phase detector in a PLL configuration to lock the phase and frequency of a free running oscillator with the network reference clock. This task aims to remove the high frequency phase noise components present in the reference clock, thus providing a stabler reference clock to the system. The second task is to digitally phase shift the DDMTD PLL's output clock with a picosecond time resolution.

As part of the WR project, a custom-made board was designed to meet the project requirements in terms of timing accuracy and data transmission. The description of the hardware circuit board is given in Section 6.1. In addition, various hardware components used in the implementation of the DDMTD circuit are presented. Section 6.2 discusses the FPGA design, particularly the interface architecture employed to exchange data between the digital blocks implemented in the FPGA.

In Section 6.3, an introduction to PLL systems is given. It describes of the main components of a PLL system, such as the phase detectors and loop controller design. Most important, is shown how the loop controller order influence the phase error and the phase noise or timing jitter of the filter output clock. Most of the analysis of

PLL circuits is realised using the Laplace transform, however, the implementation of loop controller is done digitally in integrated circuits such as an FPGA. For this reason, the different mathematical transformations used to translate between the continuous-time domain to the discrete-time domain are also briefly discussed.

Section 6.4 describes the design and implementation of the DDMTD circuit. It first starts with the design of the Helper PLL that generates the v_{beat} clock. Next, the performance of various loop controllers employed in the Helper PLL is compared by analysing both their simulated and measured phase-noise spectrum. Next, in Section 6.4.2, the design of a PLL that uses proposed the DDMTD circuit as phase detector is presented. This section is concluded with a discussion of the performance, in terms of resulting RMS jitter, obtained by the various loop controllers designed for the proposed DDMTD PLL design.

In Section 6.5, the capabilities of DDMTD PLL design as digital phase shifter with picosecond time resolution are analysed and discussed. A summary of the main results is provided at the end of the chapter.

6.1 Hardware

The White Rabbit (WR) board was been designed to meet the best timing accuracy at each node of the WR network. Besides distributing high accurate timing, the designed board is also responsible for processing and distributing data packets to multiple nodes. The WR hardware works as a normal Ethernet switch for data packets, implementing all the required protocols and standards. This section describes the WR hardware giving special attention to the timing clock distribution.

The WR switch contains two redundant uplink ports that receive both timing and data packets. The physical link are connected to the board by using Small Form-Factor Pluggable transceiver (SFP), thus giving flexibility to the type of fibre used in the network. However, in the current design only single mode fibres (SMF) are used. Ethernet transceivers circuits are used encode and decode the clock and data that

is transmitted and received. It also provides a recovered clock on one of its output pins.

The WR PCB was designed as microTCA Management Carrier Hub (MCH) board, to allow it to interface with the Micro Telecommunications Computing Architecture (MicroTCA system). The MCH is the central management and data switching device in a MicroTCA system.

MicroTCA is a PICMG standard for open modular systems based on Advanced Mezzanine Cards (AMC) [86]. This was designed to address high communication bandwidth, high processing capacity and high availability requirements. MicroTCA offers a wide range of options by allowing full redundant systems, partially redundant systems and systems with no redundancy at all.

To fit into the MCH specification, which limits the size of the PCB, the PCB followed a Full Height Two Tongue MCH configuration [86]. This means that the WR hardware is divided in two PCB's interconnected by high speed connectors due to the need to exchange information between each circuit board.

The two interconnected PCBs have difference functionalities:

- The main PCB, which hosts an Altera's Cyclone III EP3C120 [87] FPGA, is in charge of hosting all the features regarding the data packets that includes the switch data routing and switch management.
- The timing PCB, which hosts an Altera's Cyclone III EP3C5 [87]FPGA, is responsible for the implementation of all tasks required to process the timing information.

A block diagram illustrating the two boards is shown in Figure 6.1.

6.1.1 Timing Board

In this section a special attention is given to the description of the timing board, as it is the main board used in the implementation of the DDMTD. The timing PCB

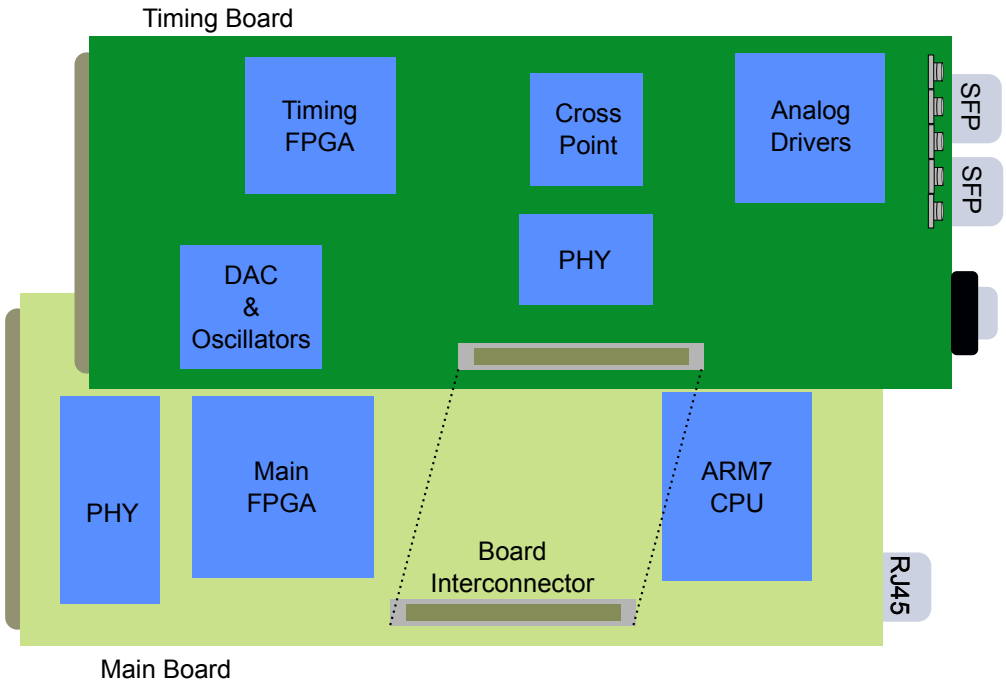


Figure 6.1: WR MCH boards

simplified schematic is shown in Figure 6.2.

The timing PCB, besides hosting an FPGA system, also carries two gigabit uplink ports with the respective SFPs [88], [89] and Ethernet transceivers [90], a crosspoint switch circuit [91] is added to the board to calibrate the transceivers recovery clock random latency.

There are two 16-bit serial DAC [92] each one is connected to a different voltage control oscillator (VCO) centred at 25 MHz. One of these, the free running clock, is responsible for generating a reference clock, or system clock, to the FPGA and the Ethernet transceivers. The free running clock is a 25 MHz Voltage and Temperature Controlled Crystal Oscillator (VCTCXO) that generates a ± 2.5 ppm centre frequency and a ± 10 ppm tuning range.

Following the reference VCO is a AD9516 [93] PLL frequency multiplier/fanout integrated circuit. This device is responsible for the generation of the 125 MHz reference clock. It is fully programmable by the FPGA through a serial link. It can lock to two distinguished reference input clocks, one is the reference clock generated by the

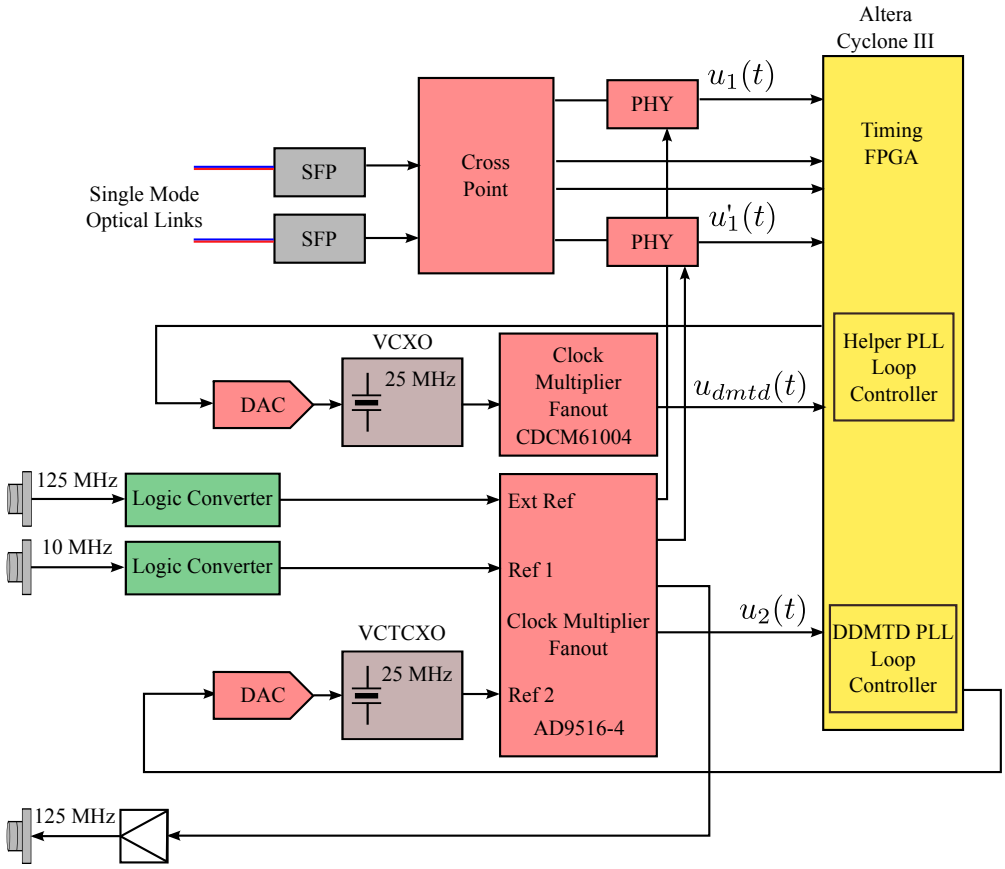


Figure 6.2: Timing Board Schematic

recovery clock and the other clock reference is a 10 MHz source generated by a high stable frequency source such as Caesium or Rubidium atomic clocks. The 10 MHz input is used when the board is configured to act as a Master node in the timing network.

To generate a 125 MHz reference frequency the AD9516 multiplies the reference frequency by five, in the case of 25 MHz reference, and by 12.5 when the reference is from a 10 MHz clock source.

The remaining VCO is used to generate the beat frequency in the DDMTD Helper PLL circuit. The Helper PLL VCO is a 25 MHz Voltage Control Crystal Oscillator (VCXO) with a minimal pulling range of ±100 ppm. The clock multiplier/fanout is implemented by a PLL chip from Texas Instruments, the CDCM61004 [94], selected due to its low jitter and low cost.

To minimise the clock noise the majority of the clock signals are fanout to various hardware modules using differential signalling such as LVDS or LVPECL.

This hardware architecture was designed to maintain the reference clocks with the lowest jitter. Others architectures could have been chosen, such as to use a 125 MHz reference oscillator plus a clock distribution chip, instead of the 25 MHz oscillators. However, this particular solution increases the cost of a single board as the 25 MHz oscillator plus frequency multiplier is today less competitive than a 125 MHz oscillator.

As already mentioned, the timing board hosts a Cyclone-III FPGA with less hardware resource than the main circuit board.

The tasks for the timing FPGA are to implement:

- The digital controllers for both the Helper PLL and the Reference PLL.
- The signal processing required for the DDMTD circuit, which include the deglitching techniques and the time difference measurements.

Other tasks include the implementation of serial links to control the PLLs multipliers, the DACs and the cross-points.

One of the reasons to use two different FPGAs is to maintain a simple hardware routing procedure so that the different delays between the timing components remain almost constant during each synthesising process. Another reason is related with the fact that the routing process can be done easier and faster in a smaller FPGA. In addition, having two different FPGAs allows hardware designers to work in parallel without overlapping or colliding.

Both the circuit schematic and board layout were designed with Altium Designer Tool. Special care was taken to isolate sensitive clock signals from the digital and power supply sections. In addition, the timing clocks are routed around the board with the smallest path lengths possible. The designed timing circuit board is shown in Figure 6.3.

Figure 6.3: Timing PCB

6.2 FPGA Design

The register-transfer level (RTL) code have been written in VHDL'93, and no proprietary IP cores has been used in the design, except for some FPGA vendor unavoidable cores instantiation.

Each hardware module is interfaced through a memory-mapped Wishbone System-on-Chip bus, which is used by the main CPU to communicate, debug and control the internal registers of the difference blocks, as shown in Figure 6.4.

Wishbone utilises "Master" and "Slave" architectures that are connected to each other through an interface called "Intercon". Master is an IP core that initiates the data transaction to the Slave IP core. Master starts a transaction by providing an address and control signal to Slave. Slave in turn responds to the data transaction with the Master with the specified address range.

The Intercon is the medium consisting of wires and logic, which helps in data transfer between Master and Slave. The interconnection can be described using hardware description languages like VHDL and Verilog, and the system integrator can modify the interconnection according to the requirement of the design. This makes Wishbone interface different from traditional microcomputer buses. Wishbone interface supports variable interconnection. Master and Slave interfaces may use four types of

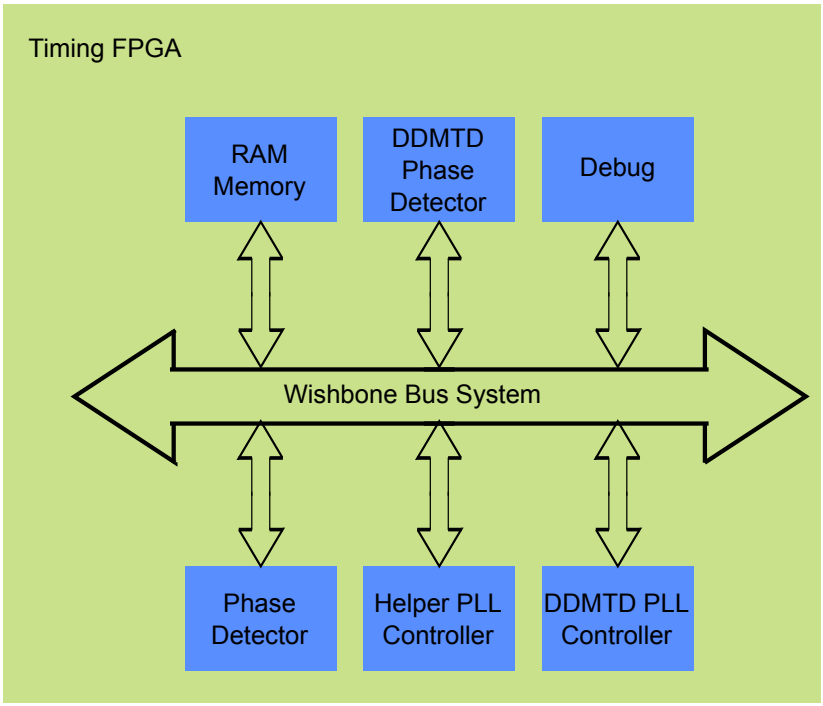


Figure 6.4: Timing FPGA Communication Interface

interconnections such as: point to point; dataflow; shared bus and crossbar switch interconnection [95].

The point-to-point interconnection is the simplest one that allows a single Master interface to connect to a slave interface. The dataflow interconnection is needed for sequential data processing.

In the shared bus interconnection, two or more Masters can be connected with one or more Slaves. An arbiter is used to allow the master to gain access to the shared bus. Crossbar switch interconnection allows two or more Wishbone masters to access two or more slaves at the same time. More than one master can use the interconnection as long as two masters do not access the same slave at the same time. Wishbone supports all the popular data transfer bus protocols such as single Read/Write, block Read/Write and Read- Modify-Write (RMW). Big-endian and little-endian data ordering schemes are also supported by Wishbone [95].

A CPU (ARM9) running a Linux Operating System has a External Bus Interface

(EBI) connection to the Main FPGA.

The Main FPGA maps the memory that should be sent to the timing FPGA and serialise it in a SPI interface.

In the Timing FPGA, the SPI data is converted to the wishbone bus interface and then sent to the appropriate hardware module.

This procedure is transparent for both the CPU users and the digital blocks in the Timing FGPA.

From the boards' CPU and via wishbone interface is it possible to control and debug the timing hardware in a faster and simpler way.

The timing FPGA contains several clock domains, the REF clock, DMTD clock and SYS clock. The SYS clock is the global system clock driving the packet processing pipeline and wishbone bus, its frequency is set to 62.5 MHz.

The design of a PLL is a quite challenging process specially for low jitter applications. In the next section, is discussed the various elements that compose a PLL. In particular, it contributes in the design of the loop controller DDMTD's PLL to select the best circuit bandwidth to achieve optimal jitter transfer from the reference clock and local clock.

6.3 Phase Lock Loop

A Phase Lock Loop (PLL) is composed by the following elements; a phase detector (PD), the loop controller (LC), the frequency dividers (DIV) and a voltage controlled oscillator (VCO). These elements are connected as represented in Figure 6.5.

The phase of the input signal is denoted by θ_i and the output phase of the VCO is represented by θ_o, the units of both signals are radians. The phase error is defined as:

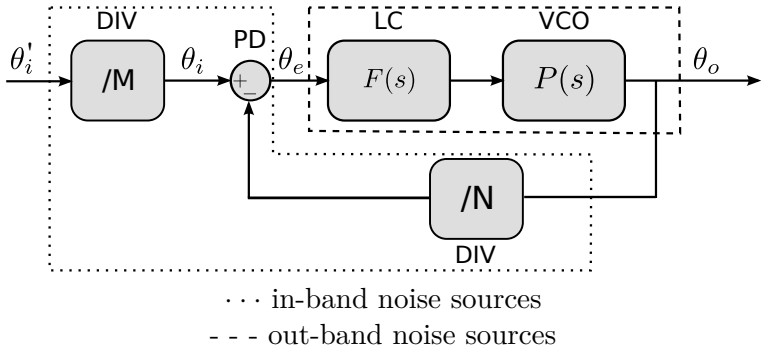


Figure 6.5: PLL Block Diagram

$$\theta_e = \theta_i - \theta_o \tag{6.1}$$

Assuming that the PLL is locked, in other words that the input signal and the output signal are very close to each other, and that the phase detector is linear then the output of the phase detector is,

$$V_d = K_d \left(\theta_i - \theta_o\right) \tag{6.2}$$

Where K_d is the phase detector gain and its units, depends on the phase detector employed but generally they are {voltage, amperes, ticks} per radians. If it is an analogue mixer then the units are voltage per radians if it is a sequential (or digital) phase detector then its units are ticks per radians.

The output signal, V_d, from the PD is processed by the loop controller $F(s)$, whose main purpose is to establish the dynamics of the feedback loop and to deliver a suitable control signal to the VCO. The loop controller output is a voltage V_c that controls the frequency of the VCO.

The VCO transfer function, P(s), represents the frequency deviation from the VCO centre frequency when a control signal is applied. The frequency deviation is $\Delta\omega = K_o \Delta V_c$ in rad/s where K_o is the VCO gain and has the units of (rad/s)/V. Since frequency is the derivative of phase, the VCO main operation is described by $d\theta_o/dt =$

$K_o V_c(t)$ or by its Laplace transform as $\theta_o = K_o V_c/s$.

Transfer functions of the individual elements can be combined to obtain the overall loop transfer functions needed for analysis and design purposes. The open loop transfer function is defined as

$$G(s) = \frac{\theta_o}{\theta_e} = \frac{K_d K_o F(s)}{s} \tag{6.3}$$

the PLL or closed loop transfer function is defined as:

$$H(s) = \frac{\theta_o}{\theta_i} = \frac{G(s)}{1 + G(s)/N} = \frac{K_d K_o F(s)}{s + K_d K_o F(s)/N} \tag{6.4}$$

and finally the error transfer function is defined as:

$$E(s) = \frac{\theta_e}{\theta_i} = \frac{1}{1 + G(s)/N} = 1 - H(s) = \frac{s}{K_d K_o F(s)/N} \tag{6.5}$$

The vast majority of practical PLLs either are second-order or are designed approximately as a second-order loop by neglecting higher-order effects, at least for initial design [96].

6.3.1 Phase Detector

In this section, different digital phase detector architectures are presented and their main advantages and disadvantages are listed.

Hogge Phase Detector

Clock recovery from a data stream, known as clock/data recovery (CDR), requires a special type of phase detector. One of the most popular is the so-called Hogge [97] detector, shown in Figure 6.6. The data signal is applied to the input of the DFF1 and to an XOR1 logic gate. The DFF1 is a positive edge clock D FF while

DFF2 is negative edge clocked D FF. The output from the DFF1, the retimed data, is connected to the DFF2, to the remaining input of the XOR1 gate and the input of the XOR2 gate. The XOR2 gets another input for the output signal of the DFF2. The output from the XOR2 is a fixed square pulse wave of half the period for each transition of the retimed data. The XOR1 output is a variable width pulse wave for each transition of the data signal. The width of the pulse depends on the position of the clock.

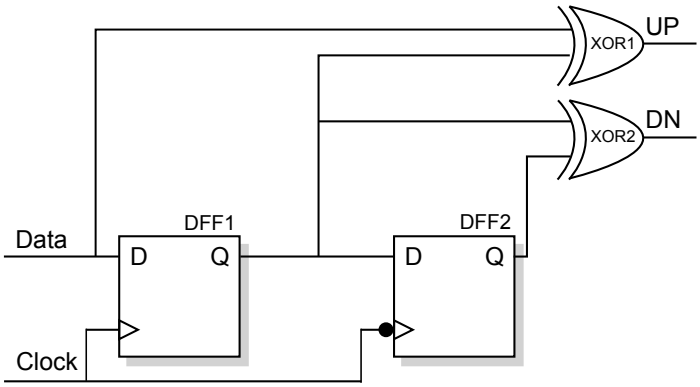
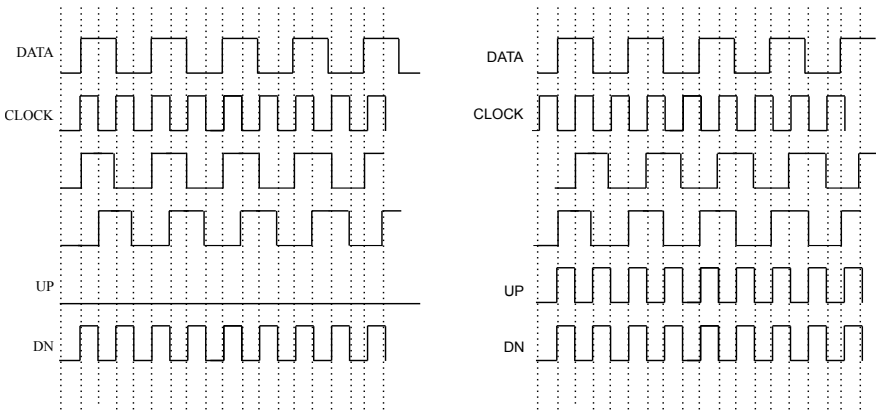


Figure 6.6: Hogge Phase Detector

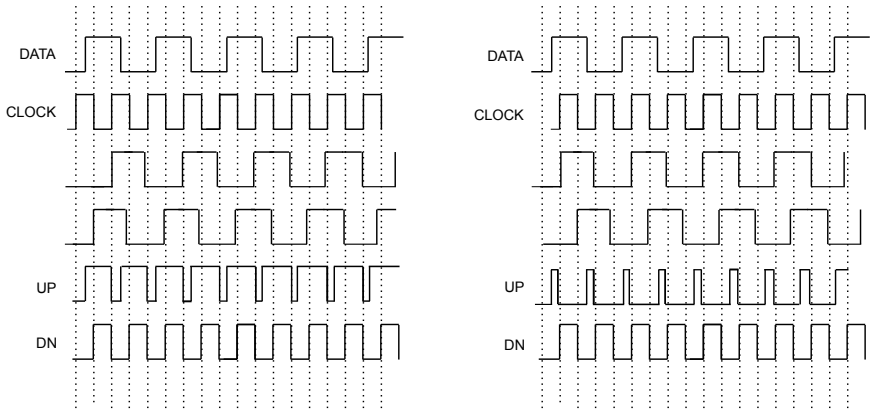
Figure 6.7 and 6.8 are timing diagrams that illustrate the behaviour of the Hogge phase detector in four situations. First, in Figure 6.7 (a), it is shown the case where the clock and data are aligned. In this situation the variable pulse signal UP does not show any pulse as the clock signal is aligned with the data. Figure 6.7 (b) shows the data and clock signal phase shift by 90°, in this situation the pulse width of the signal UP has the same characteristics as the signal DN. This situation in the ideal for the circuit as the positive edge trigger is located at the centre of the data "eye".

Figure 6.8 (a) depicted the behaviour where the clock signal is leading the data signal. The signal UP shows a higher pulse width, which indicates that the clock is leading with respect to the data signal. Similar if the clock signal is lagging the data signal then the pulse width of the signal UP is smaller, this situation is depicted in Figure 6.8 (b).



(a) Clock and Data Aligned (b) Clock and Data phase shifted 90°

Figure 6.7: Hogge Phase Detector Time Diagrams



(a) Clock Leading Data (b) Clock Lagging Data

Figure 6.8: Hogge Phase Detector Time Diagrams

Bang-Bang Detector

The Alexander or Bang-Bang phase detector [98], shown in Figure 6.9, is unique among the digital phase detectors presented here, as its behaviour is never linear. Instead, the detector acts as a relay over the region from lead to lag in respect to the reference clock. A linear analysis is not applicable in this case, and thus the bandwidth in a strict sense is not defined [99].

The Alexander circuit samples the data signal at three distinguish points and uses logic to determine whether the data is leading or lagging the reference clock signal. Figure 6.10(a)-(b) illustrate the result of sampling in the two different cases. The one

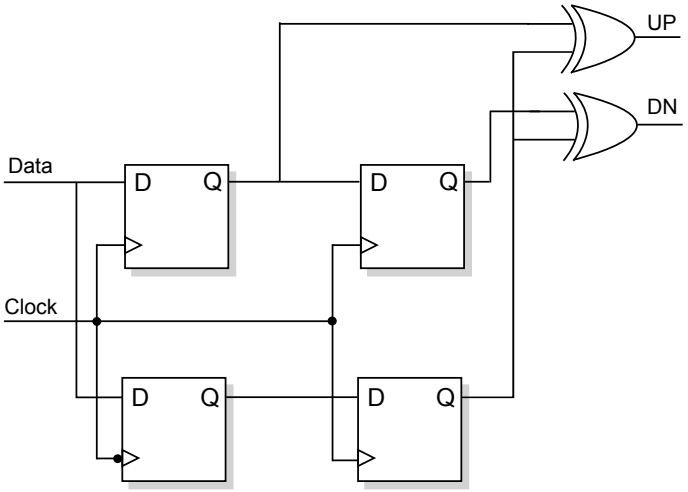
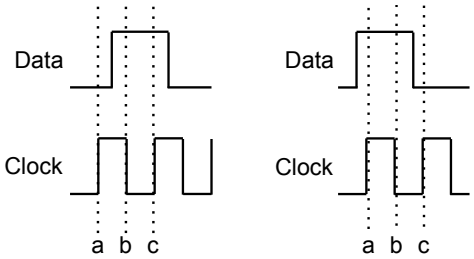


Figure 6.9: Alexander Phase Detector



(a) Leading (b) Lagging

Figure 6.10: Alexander Phase Detector Sampling

depicted in Figure 6.10(a) shows the data signal leading the clock, in this case the sampled signals shows a=0, b=1 and c=1. Shown in Figure 6.10(b) is the situation where the data signal is lagging the clock signal resulting in the sample signals a=1, b=1 and c=0. However, there are other cases where the data and clock signal result in different sampled signals, these cases are presented in Table 6.1.

The state "Invalid" results from the clock signal having a clock frequency different than the embedded clock signal. Further discussion on the modelling of bang-bang phase detectors is given in [99].

Table 6.1: Alexander Phase Detector Decision Table

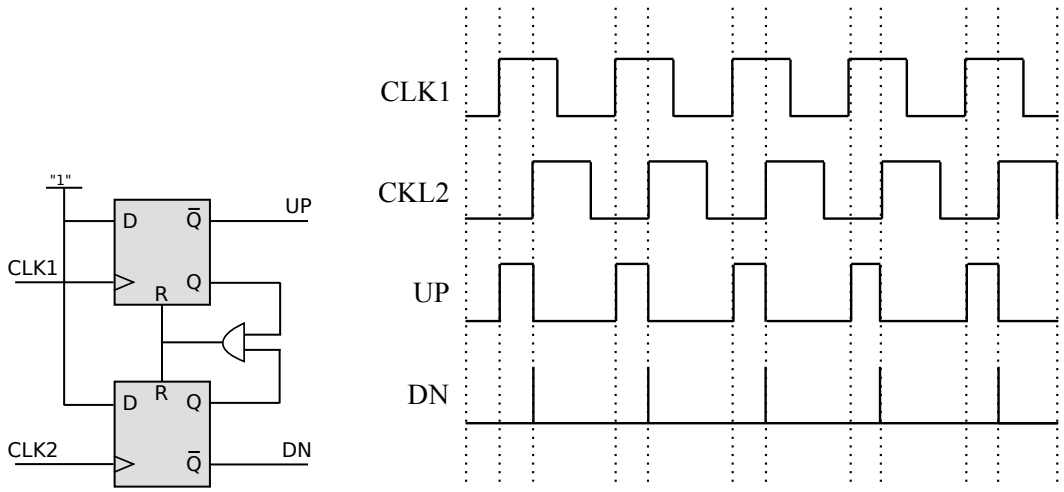
a	b	c	Result
0	0	0	Invalid
0	0	1	Leading
0	1	0	Invalid
0	1	1	Leading
1	0	0	Lagging
1	0	1	Invalid
1	1	0	Lagging
1	1	1	Invalid

Phase/Frequency detector

However, the most important and best know PD is the phase/frequency detector (PFD), it was first disclose by J.I. Brown in [100]. A basic PFD consists of a pair of D flip-flop (DFF) plus a AND gate and a delay in a feedback connection as shown in Figure 6.11. The data terminals of the DFF are held permanently to the logic level "1". Transitions from the input signal and from the feedback signal are applied to clock terminals of the DFF. The output from one of the DFFs is labelled UP and the other DN. A transition of the correct polarity turns on its associated DFF. If UP and DN are "1" simultaneously, as detected by the AND gate, feedback resets both DFF.

The output phase of the digital phase detector is generally driven by a low noise gate, charge-pump, to produce the actual DC output value. Details on charge-pumps are not discuss in this work, because no implementation of charge pumps are realised, however, the subject if found extensively in the literature [101].

The gating signal of the phase detectors can also be measured by using a fast digital counter. This output value is proportional to the time different, and the phase noise floor is given by the frequency of the counter, this method is known as time-to-digital converter (TDC) [78].



(a) Symbol (b) Waveforms

Figure 6.11: PFD Phase Detector

Noise Contribution

It is known that within the PLL loop bandwidth the phase noise is typically dominated by noise added by the frequency dividers and phase detector [102]. For frequencies well below the loop bandwidth the phase noise plot, typically flattens out resulting in the in-band phase noise floor.

Combining the $20\log_{10}(N)$ noise improvement due to the transfer function and the $10\log_{10}(N)$ degradation due to the added phase detector noise, the net effect on phase noise is $10\log_{10}(N)$. In other words, if N were increased by 10, there would be a 10 dB degradation in the phase noise. This is the reason why phase noise floor is not very meaningful without also knowing the sampling frequency [102].

The phase noise generated by the phase detector is given by:

$$\mathscr{L}_{floor} = \mathscr{L}_{1Hz} + 10\log_{10}(f_s) + 20\log_{10}(N) \tag{6.6}$$

where N is the division ratio in the loop and f_s is the phase detector sampling frequency, and $\mathscr{L}_{1Hz}$ is the noise generated by the device at 1Hz bandwidth.

By substituting $N = \frac{f_o}{f_s}$ into (6.6) results in :

$$\mathscr{L}_{floor} = \mathscr{L}_{1Hz} + 20\log_{10}(f_o) - 10\log_{10}(f_s) \tag{6.7}$$

This demonstrates that for a given PLL and desired output frequency, the phase noise floor may be decreased by 3dB by doubling the phase sampling frequency.

An analytic demonstration is given by [103] and described as follows. Additive noise, predominantly thermal, within the PFD gives rise to timing jitter on both the rising and falling edges of the output pulses. This may be considered equivalent to a certain input timing RMS jitter of Δt seconds on the reference input of a hypothetical, noise-free PFD and, in turn, may be related to an equivalent phase jitter at the PFD input, which for a phase detector operating frequency f_s, is given by

$$\Delta\theta_{in} = 2\pi f_s \Delta t \tag{6.8}$$

where is $\Delta\theta_{in}$ the RMS phase error, and Δt the RMS jitter. In practice, Δt is very small (in the picosecond range). The PFD, being a sampling device with an output pulse train of low duty cycle in a locked loop, is a good approximation to an impulse sampler thus having an equivalent noise bandwidth of half the sampling frequency and virtually uniform spectral density of translated components over this frequency range [104].

$$S_{\theta_{in}}(f) = \frac{(\Delta\theta_{in})^2}{f_s/2} = 8\pi^2 f_s \Delta t^2 \left(\ rad^2 \ / \ \text{Hz} \right) \tag{6.9}$$

This indicates a 10dB/decade increase in PFD phase noise with the phase detector operating frequency, f_s. The resulting output phase noise power spectral density is subject to a gain within the loop bandwidth equivalent to the divider ratio, N $= f_o/f_s$, where f_o is the output frequency of the synthesiser, giving the following expression:

$$S\theta_{out}(f) = 8\pi^2 f_s \Delta t^2 N^2 = \frac{8\pi^2 \Delta t^2 f_o^2}{f_s} \tag{6.10}$$

This now shows a 10dB/decade decrease in output phase noise with the phase detector operating frequency, f_s. Assuming that the overall output phase noise is smaller than 0.1 rad, then a narrowband phase modulation approximation [103] may be employed to arrive at the output phase noise. The SSB noise power density relative to carrier is given by

$$\mathscr{L}_{in-band}(f) = \frac{S\phi_{out}(f)}{2} = 10\log_{10}\left(\frac{4\pi^2 \Delta t^2 f_o^2}{f_s}\right) \text{dBc/Hz} \tag{6.11}$$

and this may be expressed more conveniently in dB terms:

$$\mathscr{L}_{in-band}(f) = \mathscr{L}_{FOM}(f) + 20\log_{10}(f_o) - 10\log_{10}(f_s) \tag{6.12}$$

where $\mathscr{L}_{FOM}(f)$ is the Figure of Merit of the phase/frequency detector, which is constant for a given device. This result indicates that the phase noise increases at a rate of 20 dB/decade with the output frequency and decreases at a rate of 10dB/decade with the PFD operating frequency. This latter -10 dB/decade dependency on f_s is significant and not widely appreciated. It explains why it is clearly advantageous, in synthesiser design, to operate phase/frequency detectors at the highest possible frequency in order to reduce the in-band phase noise floor.

The expression in Eq. 6.12 serves two purposes, it allows calculation of the PFD in-band phase noise for a given phase detector under any combination of reference and output frequencies. It also allows a normalised Figure of Merit to be associated with any phase detector in order to compare the noise performance between devices.

6.3.2 Loop Controller Design

In a PLL configuration the phase error signal, θ_e, is processed by the loop controller, whose purpose is to establish the dynamic performance of the PLL. The loop controller is designed so that the feedback loop removes the phase high frequency components of the input signal and tracks its low frequency components.

The PLL can be designed to have different orders, but the most widely used is the second-order. Next, in this section, the influence of the PLL's order on the output phase noise and on the phase tracking of the input clock signal is investigated.

First-Order PLL

The first PLL configuration discussed is the one in which the loop controller F(s) is basically a scalar gain, defined as K_p with the units of volts per radian if its an analogue frequency mixer. In this configuration the feedback loop contains a single pole, given by the VCO transfer function, for this reason it is known as a first order PLL[†].

The advantages of the first order PLL is obviously the simplicity of the design, in addition this configuration is always stable.

Nevertheless the advantages named are opposed to a couple of disadvantages: steady-state phase error and bandwidth are linked in this loop configuration. A requirement for a big range of PLL applications is that the steady-state phase error converges to zero independent of the bandwidth. The inability to fulfil this requirement make the first-order loop configuration rarely used.

The evaluation of the limitations of the first order loop is as follows. Specifically, the input-output phase transfer function is derived as:

$$\frac{\theta_{out}(s)}{\theta_{in}(s)} = H(s) = \frac{K_o K_d K_p}{s + K_o K_d K_p} \tag{6.13}$$

[†]In classic control theory the order of the controller is defined by the number of poles (or integrators) within the closed loop [105]

the closed-loop bandwidth is given as:

$$\omega_n = K_o K_d K_p \tag{6.14}$$

To show that the closed-loop bandwidth and the phase error are coupled, the input-to-error transfer function is evaluated:

$$\frac{\theta_e(s)}{\theta_{in}(s)} = 1 - H(s) = \frac{s}{s + K_o K_d K_p} \tag{6.15}$$

the input signal is a constant-frequency clock signal with a frequency ω_i, then the phase input ramps linearly with the time at a rate of ω_i rad/s. The Laplace-domain representation of the input signal is thus given as :

$$\theta_{in}(s) = \frac{\omega_i}{s^2} \tag{6.16}$$

so that,

$$\theta_e(s) = \frac{\omega_i}{s\,(s + K_o K_d K_p)} \tag{6.17}$$

The final value theorem [105] states that steady-state error of a transfer function G(s) is equal to

$$\lim_{t\to\infty} g(t) = \lim_{s\to 0} s\, G(s) \tag{6.18}$$

if all poles of sG(s) are in the left half-plane.

Therefore, the steady state error of θ(s) is

$$\lim_{s\to 0} s\phi_e(s) = \frac{\omega_i}{K_o K_d K_p} = \frac{\omega_i}{\omega_n} \tag{6.19}$$

Eq. 6.19 shows that the steady-state phase error is the ratio between the input

frequency and the PLL close-loop bandwidth; one radian of phase noise results in the PLL close-loop bandwidth being equal to the input frequency. To reduce the steady-state error it is required that the PLL close-loop bandwidth is bigger than the input frequency.

Since the VCO control voltage derives for the output of a phase detector, there must be a nonzero phase error. To produce a given control voltage with a smaller phase error it is required an increase in the gain that relates the VCO control voltage to phase detector. As shown in Eq. 6.14 an increase in gain raises the closed-loop bandwidth, so a bandwidth increase results in a reduction in the steady-state phase error.

Since a first-order controller has other undesirable properties, such as frequency jump, it is and should be avoided in most applications.

Second-Order PLL

The second-order PLL is the most widely used order in PLL design for the main following reason. To produce zero phase error, it is required an element that can generate and arbitrary VCO control voltage from zero phase detector output, implying the need for an infinite gain. To decouple the steady-state error from the bandwidth, however, this element needs to have infinite gain only at DC, rather than at all frequencies. An integrator has the prescribed characteristics, and its use leads to a second-order loop.

A typical controller, with the required characteristics, is the well known proportional-plus-integral (PI) loop controller. Its transfer-function equation is given as:

$$F(s) = K_p + \frac{K_i}{s} \tag{6.20}$$

where K_p is the gain coefficient of the proportional path through the controller and K_i is the coefficient of the integral path through the controller. The coefficient K_p

is dimensionless but K_i must have dimensions of $(sec)^{-1}$ to make F(s) dimensionless overall.

Additional configurations for loop controllers for the second-order PLL exist, this one is by no means the only one that might be used [106].

Now substituting the loop-controller transfer functions given by Eq. 6.20 into the basic system transfer function of Eq. 6.4 its obtained,

$$H(s) = \frac{K_d K_o (K_p s + K_i)}{s^2 + s K_d K_o K_p + K_d K_o K_2} \tag{6.21}$$

The denominator polynomial (characteristic polynomial) of this transfer function is of second degree, so that the PLL is said to be second-order. The two roots of the denominator are the poles of the transfer function, and the root of the numerator is a zero located at s = - $K_i / (K_p K_o K_d)$.

The best known set of parameters for a second-order PLL consists of the undamped natural frequency ω_n rad/sec (usually just natural frequency) and the dimensionless damping factor ζ. Natural frequency and damping are an attractive set of parameters because of their intuitive physical description and because of their widespread occurrence in the PLL literature. These parameters are defined for a second-order type 2 PLL as

$$\frac{\phi_{out}}{\phi_{in}} = \frac{2\zeta\omega_n s + \omega_n^2}{s^2 + 2\zeta\omega_n s + \omega_n^2} \tag{6.22}$$

where :

$$\begin{aligned} \omega_n &= \sqrt{K_d K_o K_i} \\ \zeta &= \frac{K_p}{2}\sqrt{\frac{K_d K_o}{K_i}} \end{aligned} \tag{6.23}$$

An illustration of | H(s) | is given in Figure 6.12 for different values of the damping

factor, and a natural frequency of 9 rad/sec.

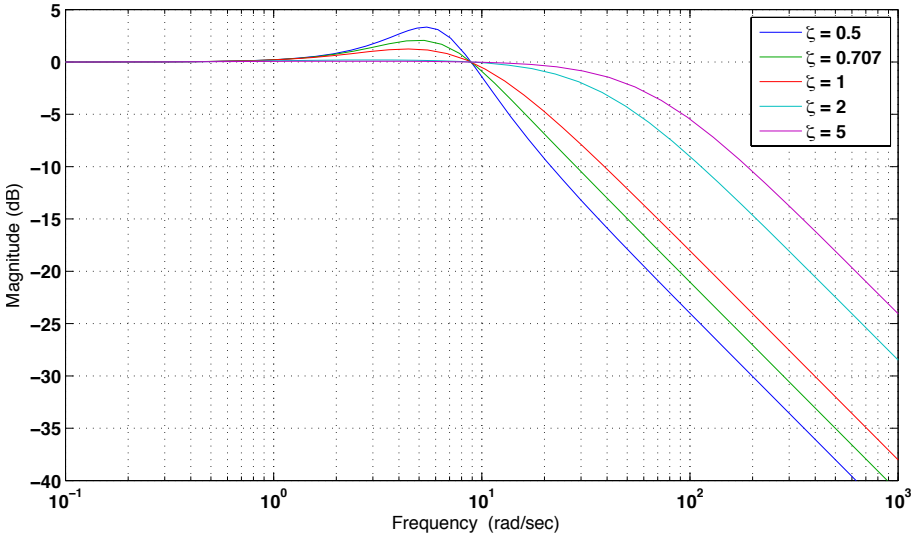


Figure 6.12: Frequency Response | H(s) | for a second-order type 2 PLL

The bandwidth of the controller loop is defined by the ω_{-3dB}, although there are other definitions for such term the one mentioned is the most widely used.

The -3 dB bandwidth of the loop is obtained by setting the magnitude of Eq. 6.22 to $1/\sqrt{2}$ and is given by [107],

$$\omega^2_{-3dB} = \omega^2_n \left[2\zeta^2 + 1 + \sqrt{(2\zeta^2+1)^2 + 1}\right] \tag{6.24}$$

The definition of such bandwidth is the major rule in designing a PLL, a bandwidth too big results in following the high frequency noise of the input phase signal and a bandwidth to small results in an offset phase error.

Values of ζ typically lie between 0.5 and 2, with 0.707 often a preferred value, but much larger values - up to 20 or 30 - are needed in application such as synchronous network or cascade PLL. Loops with damping smaller than ~0.5 have excessive overshoot in their transient responses and so are dynamically unsatisfactory. Damping factors much larger than ~1 are ordinarily needed only in special circumstances such

us applications that require cascading PLLs.

The loop bandwidth, ω_c, and phase margin, ϕ_c, are defined as [105]:

$$\mid H\left(j\omega_c\right) \mid = 1 \tag{6.25}$$

$$\pi - \angle\left(H\left(j\omega_c\right)\right) = \phi_c \tag{6.26}$$

The ϕ_c is defined as the change in open loop phase shift required to make a closed-loop system unstable. It also measures the system's tolerance to time delay [105].

A PLL is stable if its phase margin is positive and unstable if its phase margin is negative. Phase margin not only tells whether a loop is stable but also gives a qualitative indication of the loop damping.

Phase margin can be related to the damping factor ζ as [105]:

$$\phi_M = tan^{-1}\left(2\zeta\omega/\omega_n\right) = tan^{-1}\left(2\zeta\sqrt{2\zeta^2 + \sqrt{4\zeta^4 + 1}}\right) \tag{6.27}$$

The loop is stable, in principle, for all damping factor values - although at very low damping factors stability is marginal, whilst at high damping factors the loop response is a bit slow. For a typical damping factor lying between 0.5 and 1, the phase margin stays between 51^o and 76^o .

Gain Peaking

Gain peaking of a PLL plays an important role especially in synchronous networks. Gain peaking behaviour is observed in every second-order loop. There is a band of frequencies, bound roughly by the zero localisation and the second pole, where the magnitude of the transfer function exceeds unity.

The implication of this peaking is that if there is any modulation (intended or oth-

erwise) on the input with spectral components within that certain frequency band, the output modulation will have a phase excursion that exceeds the excursion on the input.

The zero causes | H(s) | to rise with frequency; the rise is terminated by the rolloff of the poles. Spacing between the zero and the closest pole decreases as K is increased (damping increases), but the pole never coincides with the zero for any finite K. Therefore, a second-order type 2 PLL always exhibits some gain peaking in | H(s) |. If this peaking is to be kept to a minimum, is require a large loop transmissions to keep the first pole as close to zero as possible.

The peaking problem is increased if there are cascading PLLs . The overall peaking of cascading PLLs can actually be large enough to cause downstream PLLs to loose lock. This can be a significant problem in some digital networks. Consider a situation in which a large number of PLLs are connected in cascade, such as in a chain of repeaters of a telecommunications system. If each repeater has only 1 dB of peaking (corresponding to $\zeta \approx 1$, a damping not ordinarily considered to be small), if the chain contains 100 repeaters (a not unreasonable number), and if no protective measures are taken, the chain will have peaking of 100 dB, which results in an unstable chain. Common standards for repeaters in the telecommunications plant specify maximum peaking of only 0.1 dB.

From analysis of the transfer function, given by Eq. 6.22, for H(s), gain peaking of a second-order type 2 PLL is found to be [96]:

$$\text{gain peaking} = 10 \log_{10} \left(\frac{8\zeta^4}{8\zeta^4 - 4\zeta^2 - 1 + \sqrt{8\zeta + 1}} \right) \tag{6.28}$$

A first or second-order loop is unconditionally stable for all values of loop gain [96]. But if the low-pass filter already has two poles, the extra pole from the VCO can make the PLL a third-order loop, which is only conditionally stable. Many additional high-frequency poles and zeros may exist because of stray capacitance and resistance. The locations of these poles and zeros must be carefully analysed to ensure the stability

of the loop.

This can be hazardous unless sufficient phase and gain margins are specifically included in the design.

Third and Higher Order PLL

Third and higher order PLL are useful in applications where high frequency components (above the PFD sampling frequency) occurs. This occurs mainly in analogue PLL where the charge pumps circuit generate a ripple signal that drives the VCO. This ripple appears as an oscillation in the VCO output frequency [108]. Additionally, the sensitivity to drifts in the master clock limits the network quality figures [109]. Another important point is that the frequency jumps inherent to the second-order loop usually cannot be accepted and additional filtering is necessary [101]. To suppress the ripple that induces jitter, an extra pole is added to the PLL's loop controller F(s), thus a degradation in the phase margin must be account to maintain the stability of the closed-loop. Most actual PLLs contain additional poles at higher frequency deliberately inserted to suppress higher-frequency disturbances resulting from the charge pump phase detector [96]. The analysis of a third-order PLL is described as follows:

$$F(s) = \frac{s + \dfrac{1}{RC_1}}{C_2 s \left(s + \dfrac{1}{R(\dfrac{C_1 C_2}{C_1 + C_2})} \right)} \tag{6.29}$$

As mentioned before, the loop controller has a zero and three poles, the zero is located at 1/RC and the poles are located at "0" and at $1/\left(R\left(C_1C_2/C_1 + C_2\right)\right)$. The phase margin degradation is easy to verify and it depicted in Figure 6.13.

As a natural extension to the third-order PLL, the fourth-order loop is often chosen

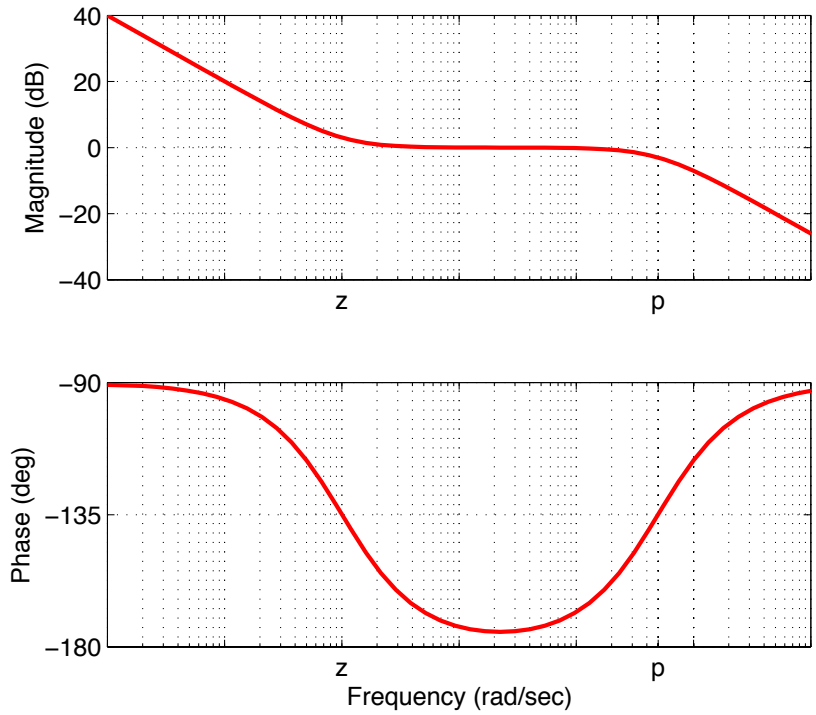


Figure 6.13: Third-Order Bode Plot

because the extra pole in the loop controller offers the designer the freedom of adding additional attenuation beyond the loop bandwidth.

In addition, because the extra attenuation this controller offers helps buffer the design from effects due to charge pump imbalance [101].

However, the magnitude and phase of the open-loop gain can become degraded by adding this additional pole to a passive charge-pump-driven loop controller. The difficulty is that this extra pole added into the controller to provide additional attenuation provides this attenuation at the expense of both a reduced loop bandwidth and phase margin, thereby increasing the lock time and compromising stability [103].

The fourth-order PLL, is characterised by one zero and three poles one of that is located in the origin, thus:

$$F(s) = \frac{R_2C_3}{C_2}\frac{s + 1/RC_1}{s(s + \dfrac{1}{R\left(\dfrac{C_1C_2}{C_1 + C_2}\right)})(s + R_2C_3)} \tag{6.30}$$

Figure 6.14 depicts a possible implementation of a passive third-order and fourth-order F(s). Its frequency response is shown in Figure 6.15.

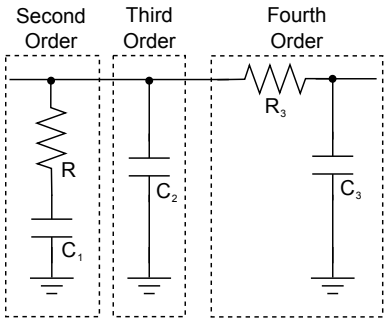


Figure 6.14: Fourth-Order PLL implementation

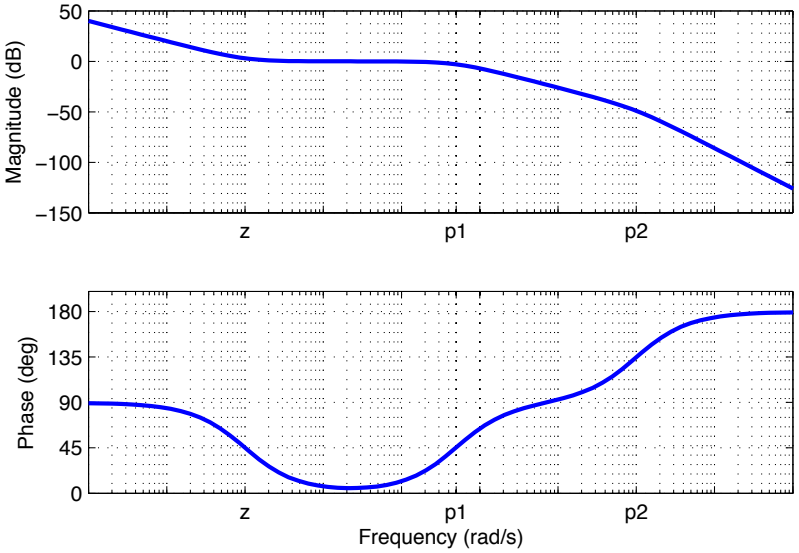


Figure 6.15: Fourth-Order Bode Plot

6.3.3 PLL Design for Optimal Bandwidth

It should be noted that in-band sources dominate within the loop bandwidth, that is for $f \ll f_c$ and the VCO noise dominates outside of the loop bandwidth, that is for $f \gg f_c$. The phase noise measured at an offset that is close to the carrier is basically independent of loop bandwidth, provided that the loop bandwidth is sufficiently wide to eliminate the VCO noise. However, the RMS phase error is more dependent on the loop bandwidth. To theoretically design for the lowest RMS jitter, this means that one needs to design such that the VCO noise contribution at $f = f_c$ is equal to the total noise contribution from the other sources at $f = f_c$, as shown in Figure 6.16. If the VCO is noisy relatively to the PLL, then this number would be smaller, and if the PLL is noisy relatively to the VCO, then this number would be larger.

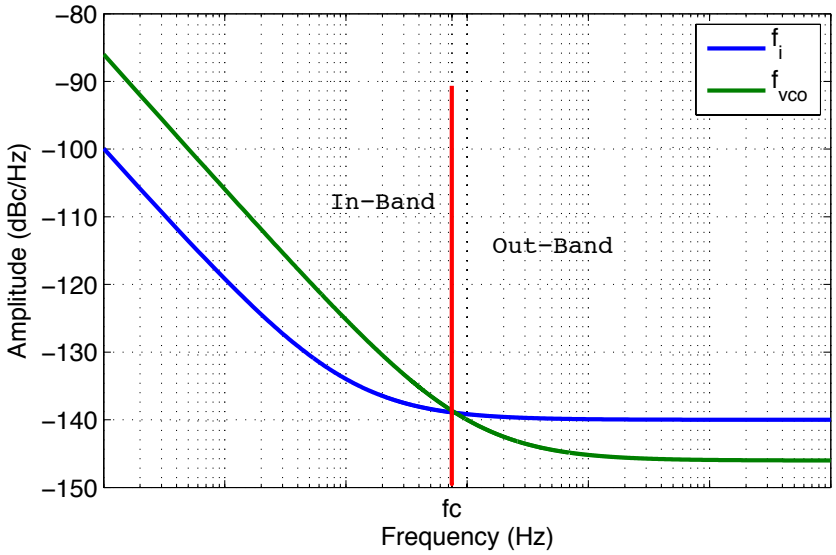


Figure 6.16: Loop Controller Optimal Bandwidth Criteria

6.3.4 Digital Design

An analogue circuit is described in the time domain by a differential equation, a digital circuit is described in the discrete domain by a difference equation. Just as a linear time-invariant differential equation is converted to the transform domain by

means of Laplace transforms, a linear shift-invariant difference equation is converted to the transform domain by means of z-transforms.

There are several methods to convert from the Laplace domain to the z-domain, such as the forward and backward difference approximations. The bilinear transformation is another method that maps the entire left half of the s-plane into the unit circle in the z-plane. In Table 6.2, the approximations of each continuous to discrete transformation are listed.

Table 6.2: Digital Approximations

Method	Approximation
Forward Rule	$s = \frac{z-1}{T_s}$
Backward Rule	$s = \frac{z-1}{T_s z}$
Trapezoid Rule	$s = \frac{2}{T_s}\frac{(z-1)}{(z+1)}$

where T_s is the sampling time of a discrete-time system.

The goal in the transformation is to preserve the frequency response and stability of the system; thus the bilinear transformation is the recommended choice. The only disadvantage of this method is the frequency wrapping, which affects the frequency response close to the sampling frequency. Since the bandwidth of the PLL is in most cases at least ten times smaller than the sampling rate the frequency warping will have a negligible effect.

Every digital loop controller suffers from quantisation effects. Quantisation is a non-linear operation whose consequences are most significant at small phase errors. To avoid the severe complications of non-linear analysis, common practice assumes that quantisation is fine enough to be ignored to first-order and that the digital controller can be analysed by a linear approximation.

6.4 Design and Implementation

This section describes the design and implementation of a PLL that employs the DDMTD circuit, as shown in Figure 5.2, as phase detector. This section starts with the design of the Helper PLL, followed with a description of the hardware architecture implemented. The main components of the Helper PLL are detailed and approximated to a numerical model to simplify the calculation of the digital loop controllers. The phase-noise spectrum for the different controllers is predicted by computer-based simulations. Phase-noise spectrum and RMS jitter are measured to validate the performance of the Helper PLL and compared with the results obtained by the simulation environment.

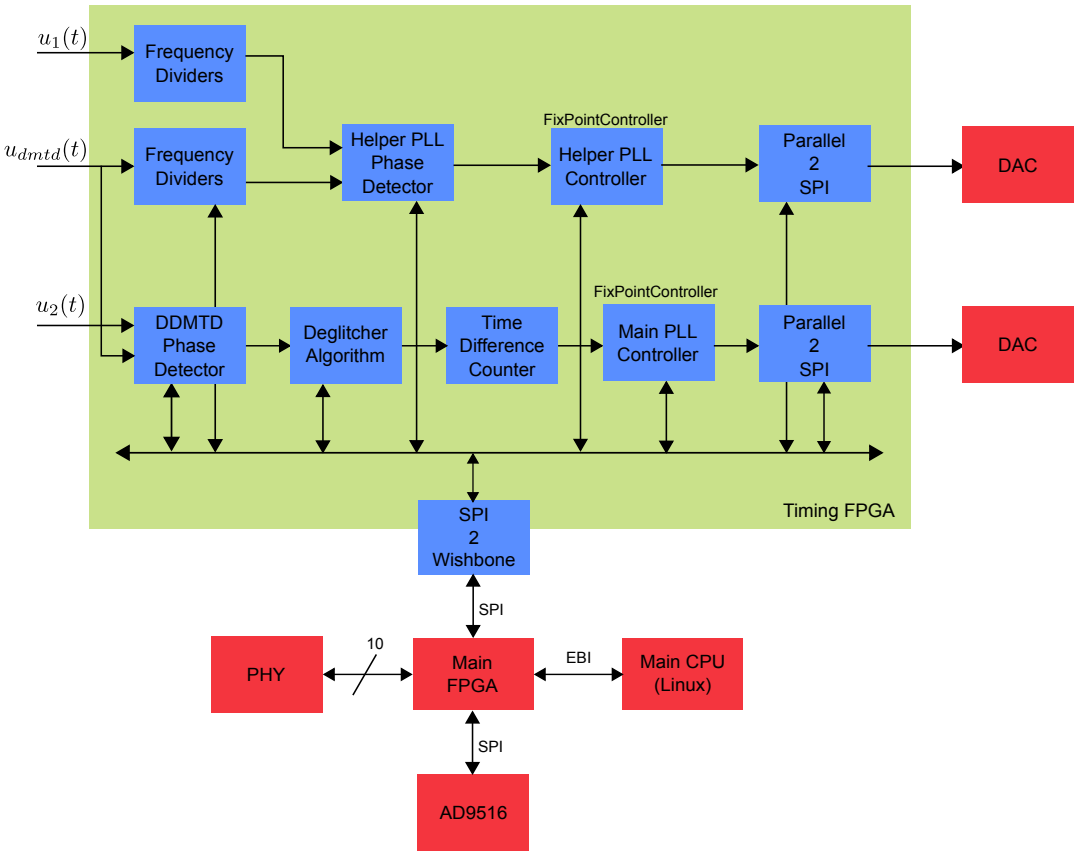


Figure 6.17: DDMTD PLL Design Block Diagram

Figure 6.17 shows a block diagram of various blocks designed in the Timing FPGA to implement the DDMTD's PLL and phase shifter.

6.4.1 Helper PLL

One of the most important building blocks of the DDMTD circuit is the Helper PLL. It is the heart of the DDMTD circuit and it defines the DDMTD time resolution as described in Section 5.2.

The hardware architecture employed for the implementation of the Helper PLL is divided in two blocks, as shown in Figure 6.18. One of the digital blocks is implemented in an FPGA , the other block is an analogue block composed by a DAC and frequency multiplier.

In the timing FPGA are implemented the digital phase detector, the frequency dividers and the digital controller.

The analogue block consists of a 16-bit serial DAC, an analogue first-order low-pass filter and VXCO and a frequency multiplier/fanout integrated circuit. There are several reasons to select this design approach of having a digital block and an external analogue block and they are described as follows: The PLL embedded in the FPGA only synthesises frequencies with an offset of 1 MHz from the reference clock, while in the Helper PLL requires clock reference with a frequency offset less than 100 kHz. This design approach allows flexibility in the selection of the Helper PLL parameters, such as bandwidth and damping factor, by means of the design of the digital loop controller. The high frequency jitter generated by the Altera's embedded PLL is very high (>200 ps) for the WR timing application, so an external VXCO with lower high frequency jitter is required to reduce the RMS jitter.

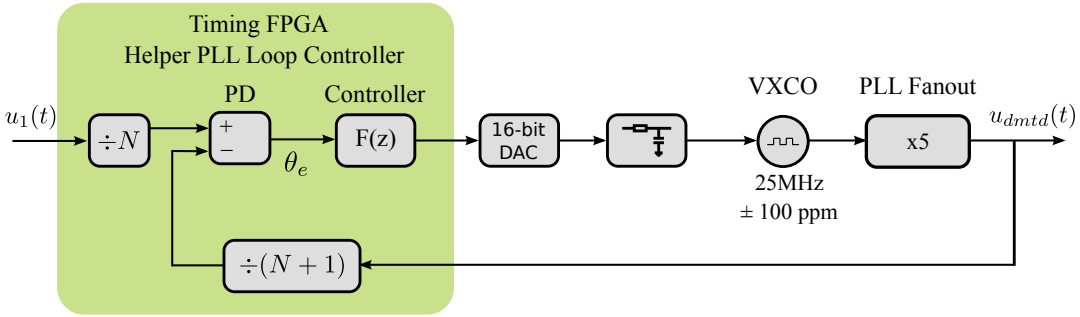


Figure 6.18: Block Diagram of the Helper PLL Implementation

The input clocks of the Helper PLL are the 125 MHz recovery clock generated by the Ethernet transceiver and the DDMTD clock input that is generated by the VCXO plus the frequency multiplier/fanout chip.

The Helper PLL inputs are frequency divided and afterwards connected to a digital phase detector that yields a 16-bit result proportional to the phase difference between the two inputs.

The output of the phase detector, which is detailed later in this section, is connected to a digital controller that adapts to the dynamics of the VXCO model and tunes its to make the clock stabler and with a predefined frequency response.

The output of the digital loop controller is converted to a voltage signal by a 16-bit serial DAC. The DAC's output then passes through a first-order RC analogue low-pass filter before acting in the VXCO. The analogue low-pass filter is added to the circuit to minimise the DAC's quantisation noise that generates spurs on the output of the VXCO [96]. The low-pass filter is designed to have a cut-off frequency of ~3.4 kHz, with R = 470Ω and C = 100nF. The low-pass filter cut-off frequency was chosen to be far way in frequency from the digital controller bandwidth so that the influence of the filtering process in the designed digital controller can be neglected.

The output of the low-pass filter is connected to a VCXO centred at 25 MHz with ± 100 ppm pulling range. This pulling range is the frequency range that the VCXO can operate, this means that the VCXO can be tuned from 24.9975 MHZ to 25.0025 MHz. The 125 MHz clock frequency reference is distributed among several hardware modules around the board, such as the main FPGA, by a frequency multiplier/fanout referenced to the VCXO. The frequency multiplier/fanout is a low-jitter clock generator with a bandwidth set to 400 kHz by design. This clock frequency multiplier fanouts the clock generated by 4 outputs, it is configured through its inputs socket pads to act as a five times clock multiplier [94].

Frequency Divider

The DDMTD time difference resolution is defined by the Helper PLL output frequency, the v_{beat} frequency.

The relationship between the frequency division, defined by the parameter M, and the resolution of the DDMTD circuit is described by Eq. 5.14 and illustrated in Figure 6.19, for an input frequency of 125 MHz. The parameter M is represented in Figure 6.19 in the form of 2^M.

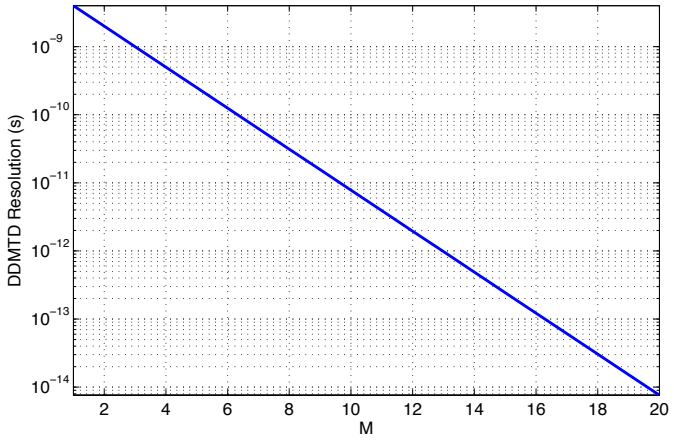


Figure 6.19: DDMTD Accuracy vs 2^M

As shown in Figure 6.19, to achieve sub-picosecond time resolution, the value of the exponent M should be higher than 13. For a division factor equal to 2^{13} the output frequency is centred at 124.9847 MHz that results in a beat frequency, $v_{beat} = 15.2588$ kHz. The DDMTD time difference resolution for this beat frequency is calculated, from Eq. 5.13, to be ~976 fs.

Phase Detector

The phase detector is implemented in the FPGA as a PFD, with a digital counter to measure the phase difference between the two pulses of the PFD and some minor additional logic, as depicted in Figure 6.20. The phase noise locking floor of the Helper PLL depends mainly on the Time-to-Digital Converter (TDC) resolution, the

higher the TDC resolution the lower is the phase noise transferred to the VXCO. In this implementation, the TDC resolution is limited 8 ns, which is originated by the 125 MHz system clock.

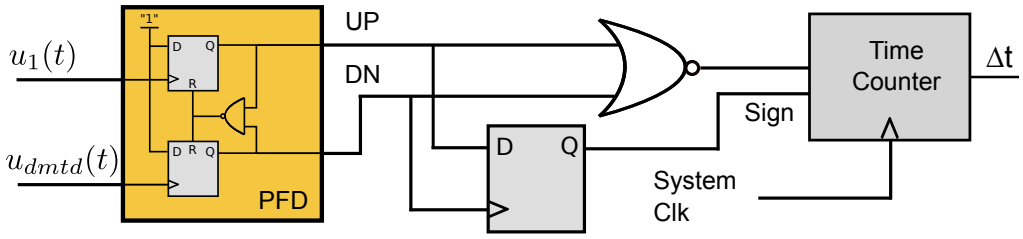


Figure 6.20: Digital Phase Detector

The relationship between $\Delta\theta = \theta_a - \theta_b$ (where θ_a and θ_b are the phases of the clock signals $u_1(t)$ and $u_{dmtd}(t)$, respectively), and the time resolution of the TDC determines the applicability of linear analysis for the Helper PLL. If the input phase error is smaller than the resolution of the TDC converter, the behaviour of the digital PD is no different from a bang-bang phase detector [99]. On the other hand, if the input phase error $\Delta\theta$ is much larger than the resolution of the TDC, then the input phase error is digitised in a cumulative linear manner with 8 ns resolution. The TDC can be modelled as a gain plus quantisation noise. Having a linear phase detector allows us to use linear techniques for the analysis of Helper PLL dynamics.

Figure 6.21 illustrates the operation of the phase detector when the phase detector is in its locking state and when the phase error is larger than the TDC resolution.

The output of the phase detector is a periodic signal with a frequency equals to v_{beat}, and a duty cycle ratio that is proportional to the phase difference, $\Delta\theta$, its output phase characteristics is shown in Figure 6.22.

Following the analysis given in section 6.3.1 for different phase detectors, the noise contribution for the above phase detector is given as:

$$S_{\varphi_{out}}(f)_{tdc} = 10\log_{10}\left(2\,\pi\,\frac{f_o}{N}\frac{1}{f_s}|G(f)|\right)^2 \tag{6.31}$$

where f_o is the input frequency, N the division factor, f_s the sampling frequency and

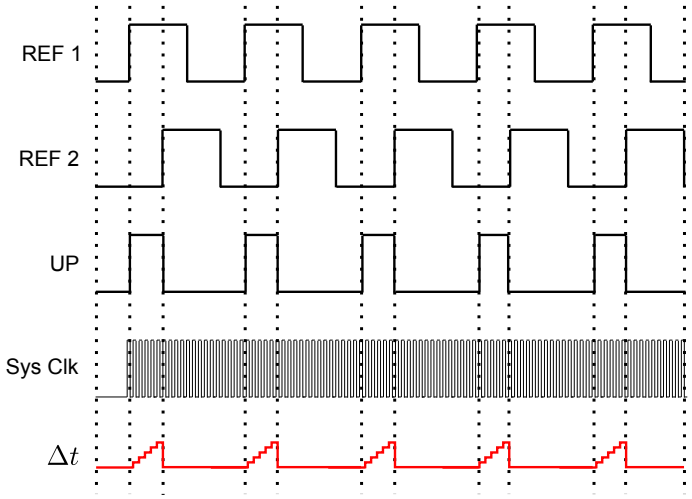


Figure 6.21: Output in locking state

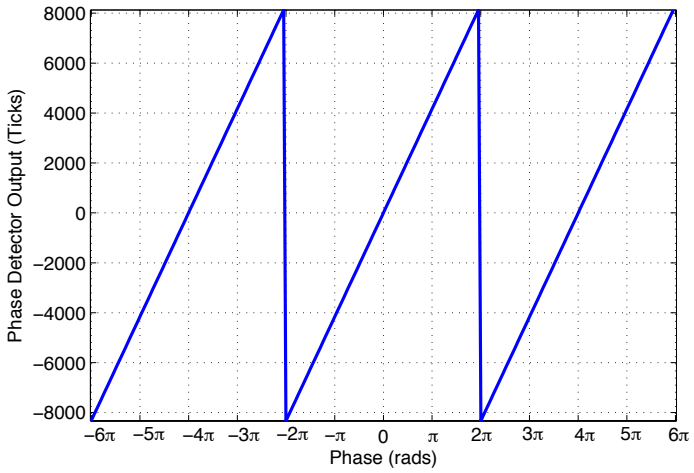


Figure 6.22: Phase Detector Characteristics

G(f) is the frequency response of the phase detector. At low frequencies, G(f)=1, and for a f_o = 125 MHz, N = 2^{13}, f_s = 125 MHz the estimated $S_{\varphi_{out}}(f)_{tdc}$ noise floor is ~(-62) dBc/Hz. Doubling the sampling frequency decreases the phase noise floor by -6 dBc/Hz.

To design the Helper PLL digital controller it is necessary to calculate the phase detector gain, K_p. The phase detector gain is equal to the ratio between the output of the phase detector and $\Delta\theta$.

$$K_p = \frac{\frac{v_{sys}}{v_{beat}}}{2\pi} \approx 1304\ ticks/rad \tag{6.32}$$

VCXO

As shown in Figure 6.18, the output of the digital controller tunes indirectly a VXCO. The VCXO incorporates a voltage gain from $[0, 3.3]$V, which is able to tune the oscillator from 25 MHz $\pm$ 100 ppm. The VCXO frequency output is then multiplied five times by a frequency multiplier/fanout chip to obtained the specified output frequency centred at 125 MHz.

The VCXO input voltage is generated by a 16-bit DAC, controlled by the Timing FPGA via a serial line SPI interface. The gain of the VXCO, K_i is defined as the ratio between the change in frequency of the VCXO with the change in the DAC input 16-bit word, that is :

$$K_i = \frac{\mathrm{d}\omega}{\mathrm{dW}_{dac}} \tag{6.33}$$

The VXCO gain was estimated by measurements by feeding the DAC's digital input with random data and reading the output frequency, from the frequency multiplier, using a time counter. The frequency measurements and respective estimation, where the y-axis represent the frequency measured in Hz and the x-axis represents the 16-bit word DAC's input, are shown in Figure 6.23.

With the measured set of data a first-order model that best fits the frequency measurements was estimated. The estimated model, also shown in Figure 6.23, is given as:

$$y = 0.66274x + 124976628 \tag{6.34}$$

where x represents the DAC input and y the output frequency measured in Hz. By

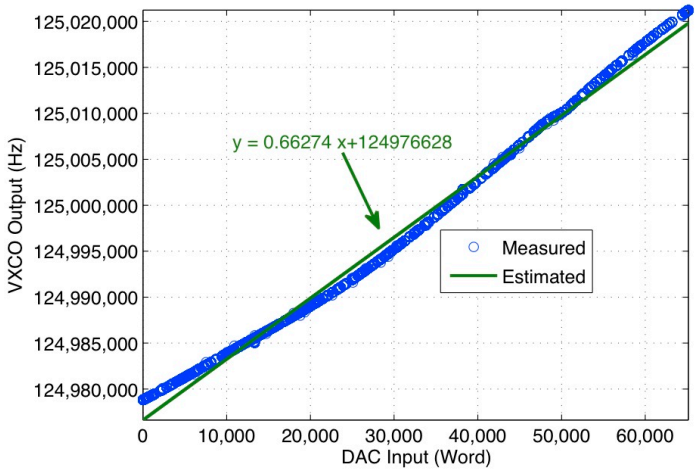


Figure 6.23: Helper PLL VXCO Characteristic

multiplying the estimated model by 2π it easy to obtain the VXCO gain, K_i, as:

$$K_i = 0.66274 \cdot 2\pi = 4.1641 \text{ (rads/sec)/tick} \tag{6.35}$$

Digital Loop Controller

The PLL parameters N, K_p and K_i calculated earlier are necessary to design the digital controller, F(z).

The purpose of the PLL design is to behave as a low pass filter, that is to follow the low frequency phase components and to filter out the high frequency phase components. By changing the PLL's bandwidth the amount of jitter transfer from the input reference clock to the output of the VCXO also changes. However, as described in Section 6.3.3, reducing the PLL's bandwidth too much may increase the jitter in the output clock, as the PLL starts to track the low frequency jitter of the VCXO.

Verification of the digital controller performance, by means of simulations and measurements allows the estimation the effect of changing the different parameters of the controller such as K_p and K_i on the jitter generated by the VXCO's clock output to be. For this purpose a set of loop controllers having difference parameters, such as

ω_n and ζ was calculated. They are shown in Table 6.3.

Table 6.3: PLL Response and Controller Parameters

Controller	PLL Response H(s)		Loop Controller F(s)	
	$\omega_n(rad/s)$	ζ	K_p	K_i
Loop 1	204	33	2.4	7.5
Loop 2	702	290	73	89
Loop 3	759	118	32	104
Loop 4	7595	59	163	10445
Loop 5	11395	59	245	23514
Loop 6	37975	59	817	261127
Loop 7	75950	59	1634	1044496

The controller F(s) was transformed to the z-domain using the bilinear transform, with a sampling frequency of 15.2588 kHz (the v_{beat} frequency). The controllers were designed and simulated in MATLAB and implemented in the FPGA using VHDL.

The performance of the loop controller is evaluated by analysing the amount of RMS jitter generated in the Helper PLL output clock. The phase-noise spectrum was measured with an Agilent signal source analyser (SSA) 5052B [70]. The applied measurement technique of the Agilent SSA is based on analysing the phase noise power spectral density of the input clock signal. The SSA locks two ideally similar and low phase noise electronic oscillators to the input clock signal with a very low loop bandwidth (≈1 Hz). The mixed signals from each oscillator with the input clock signal are then compared.

Correlated phase noise terms correspond to the phase noise of the input clock signal, uncorrelated phase noise terms correspond to the phase noise of the electronic oscillators and cancel each other. With this measurement method a gain increase of about 20 dB is achieved (assuming a sufficiently long correlation time) in system noise floor when compared to the noise floor of each individual oscillator [110].

Figure 6.24 shows the phase-noise spectra of two clocks. One is the reference clock, which is the recovery clock obtained by the Ethernet transceiver. The other is the

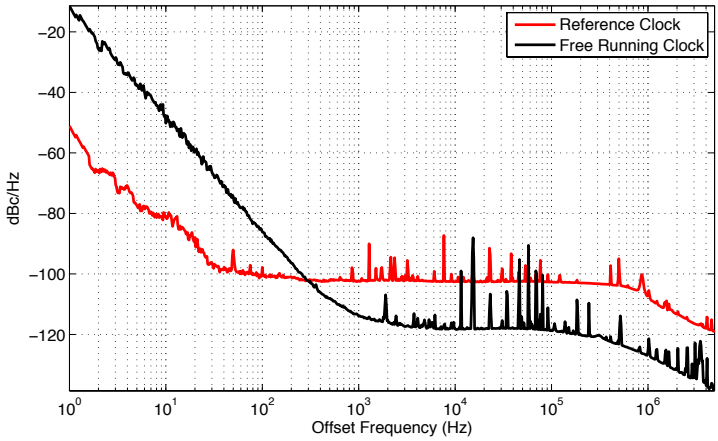


Figure 6.24: Phase-Noise Spectrum of the Reference and Free-Running Clocks

feedback clock, which is the free-running VXCO. With the phase-noise spectra from the reference clock and the feedback clock, the Helper PLL output phase-noise spectrum, for the difference controllers, is estimated.

Figures 6.25 to 6.31 show the estimated and the measured phase-noise spectrum on the output of the PLL plotted from 1 Hz to 1 MHz frequency offset from the centre frequency. The output clock is measured with a centre frequency of 124.984746 MHz, which results in the required offset of ~15.2588 kHz.

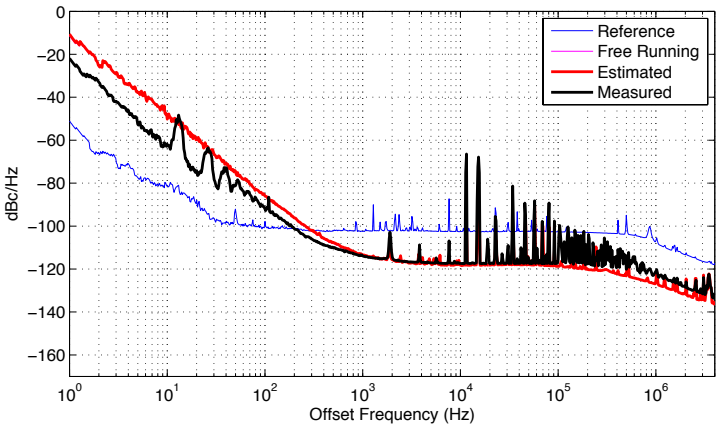


Figure 6.25: Phase-Noise Spectrum - Loop 1

Figures 6.25 - 6.27 are phase-noise spectra of the PLL with very low bandwidth, as shown in Table 6.3. The loop bandwidth of the digital controller is so low that the

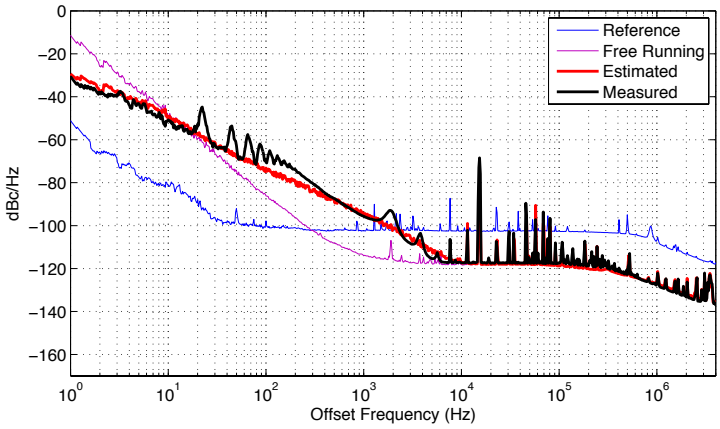


Figure 6.26: Phase-Noise Spectrum - Loop 2

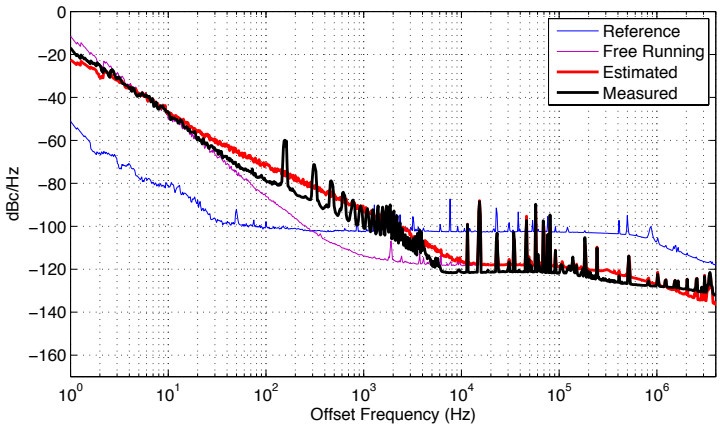


Figure 6.27: Phase-Noise Spectrum - Loop 3

feedback clock shows a phase-noise spectrum identical to the free-running clock.

The phase oscillations seen in Figure 6.25 and Figure 6.26 between the 10 Hz and the 100 Hz offset frequency are attributed to the fixed-point implementation of the controller. This is due to the loop controllers low bandwidth and the fixed-point implementation. The implemented digital controllers do not have enough precision in the arithmetic to output the proper control signal. These issues result in phase jumps in the loop controller's output.

The oscillations are also visible in Figure 6.27 but in this case between the 10 Hz and the 5 kHz offset frequency.

The "Loop 3" controller's bandwidth is higher than the previous two designs described. For this reason the phase oscillations are moved to a higher offset frequency where they are filtered by the VCXO high frequency response.

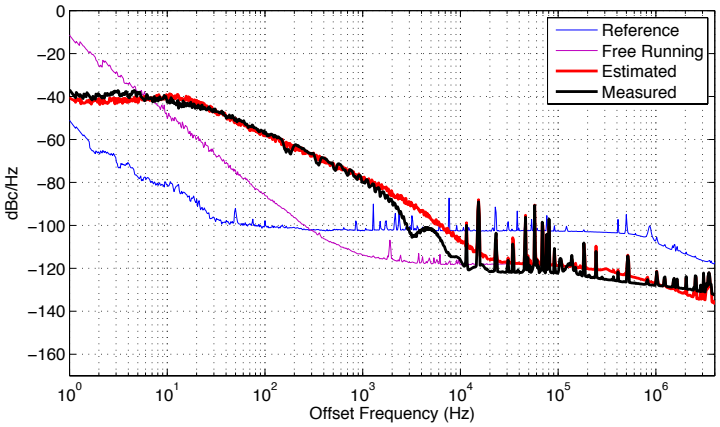


Figure 6.28: Phase-Noise Spectrum - Loop 4

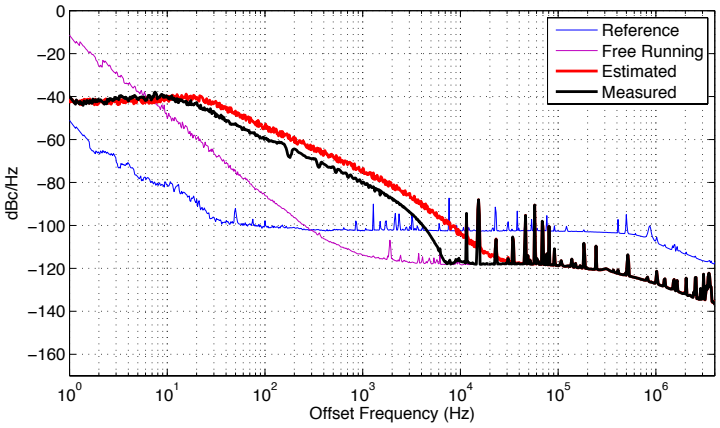


Figure 6.29: Phase-Noise Spectrum - Loop 5

From Figure 6.28 to Figure 6.31 it is noticeable that the phase detector noise floor is approximatively -40 dBc/Hz. After reaching the PLL cut-off frequency the phase-noise spectrum drops -20dB/decade until it reaches the phase-noise floor generated by the 125 MHz feedback clock.

The phase-noise spectrum measurements of the different controllers have an identical frequency response as estimated.

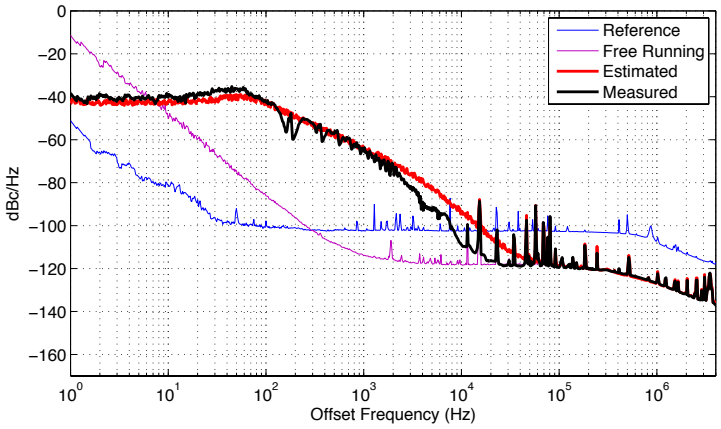


Figure 6.30: Phase-Noise Spectrum - Loop 6

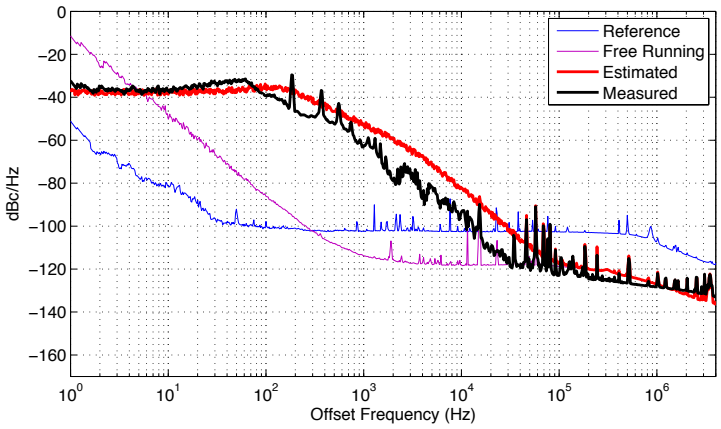


Figure 6.31: Phase-Noise Spectrum - Loop 7

From Figures 6.25 to 6.31 the RMS jitter for each one of the PLL's designs is calculated . The RMS jitter is obtained by integrating the phase-noise power spectrum over the frequency range of interest.

The low frequency integration should be as low as the instrument can measure to get the true RMS jitter that is 1 Hz[†], the high frequency integration value is set to 4 MHz.

Table 6.4 shows the integrated RMS jitter calculated from the phase-noise spectrum measured for different Helper PLL controllers. Table 6.4 shows that the controller that generates the lowest RMS jitter is the one labelled "Loop 2" with 53.9 ps.

[†]However, oscillator specification generally are given for offset frequencies above 10 Hz.

Table 6.4: Helper PLL RMS Jitter

Clock	**Jitter (ps)** [1 Hz – 4 MHz]
Feedback	14.7
Reference	275.5
Loop 1	79.5
Loop 2	53.9
Loop 3	169.5
Loop 4	87.6
Loop 5	82.1
Loop 6	240.4
Loop 7	455.9

A lower bandwidth frequency loop filter, "Loop 1", shows a higher jitter value of 79 ps. The reason for this is that a decrease in the controller's bandwidth can in fact increase the output jitter as the controller follows the VCO low frequency jitter. By increasing the controller's bandwidth the RMS jitter also increases due to the jitter generated in phase detector noise floor.

Summary

The design and implementation of the Helper PLL circuit that generates the DDMTD beat frequency clock was presented.

The Helper PLL output frequency was calculated to provide to the DDMTD circuit a picosecond time difference resolution. With this purpose, several controllers were designed and their phase-noise spectrum was estimated and measured.

The phase-noise spectra measurements showed an identical frequency response as estimated by the simulations. The RMS jitter was also estimated by integrating the phase-noise spectrum.

The measurements classified the controller labelled "Loop 2", with $\omega_n = 702$ rad/s and $\zeta = 290$, was the one that showed the lowest RMS jitter, thus is the clock with higher stability.

6.4.2 DDMTD PLL

In this section, the implementation the PLL that uses the DDMTD circuit as phase detector is described.

The DDMTD PLL uses the Helper PLL presented in the previous section that provides the DDMTD circuit with a time resolution of 976 fs. The PLL is divided in two blocks, one implemented in the FPGA, and the other one is implemented using analogue hardware components external to the FPGA. The DDMTD PLL follows a similar circuit architecture employed for the Helper PLL. However, this circuit has more strict requirements in terms of clock specification.

In the FPGA system is implemented the digital components of the DDMTD circuit such as the DFFs, the deglitching circuits, the time counter and the digital loop controller. The DDMTD PLL external block is composed by the 16-bit serial DAC, an analogue first-order low-pass filter, a VCTCXO and a frequency multiplier/fanout integrated circuit.

Figure 6.32 illustrates the arrangement between various hardware modules composing the DDMTD PLL.

The DDMTD PLL has two input clock signals, $u_1(t)$ and $u_{out}(t)$. The clock signal $u_1(t)$ is the reference clock, to be more specific, it is the recovery clock generated by the Ethernet transceiver. The $u_{out}(t)$ is the feedback clock, generated by the frequency multiplier/fanout locked to the 25 MHz VCTCXO. Both clock signals are centred at 125 MHz.

The output of the DDMTD circuit, which works as a digital phase detector in this PLL configuration, is a digital signal with 16-bit data width that measures the time difference between the two inputs clock signals. The Helper PLL output frequency sets the time difference resolution measured by the DDMTD circuit to be 976 fs. That is, each bit increment done by the time difference counter is equivalent to a time increment of 976 fs.

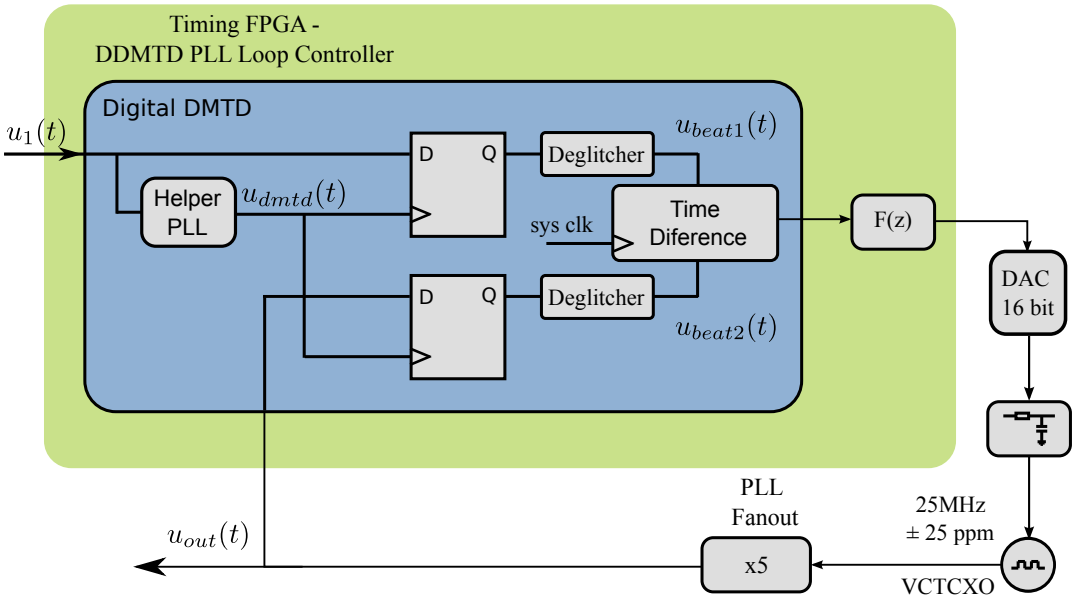


Figure 6.32: DMTD PLL Block Diagram

The result from DDMTD circuit is feed to the digital controller that tunes the VCTCXO to be aligned with reference clock. The low pass characteristic of the VCTCXO filters the high frequency phase noise of the input signal.

However, before being feed to the VCTCXO the loop controller output is converted to an analogue voltage signal through a 16-bit serial bit DAC and filtered by an analogue low-pass filter. The analogue low-pass filter was designed with a cut-off frequency set to ~3.4 kHz, following the same methodology realised for the Helper PLL circuit.

The phase detector gain, K_p, defined by the DDMTD circuit time resolution is calculated as follows:

$$K_p = \frac{\frac{125 \text{ MHz}}{15.2588 \text{ kHz}}}{2\,\pi} \approx 1304 \text{ ticks/rad} \tag{6.36}$$

The DDMTD beat frequency, calculated earlier, is $v_{beat} = 15.2588$ kHz. The time difference counter has a system clock of 125 MHz.

VCTCXO

The DDMTD PLL contains a voltage controlled, temperature compensated oscillator (VCTCXO) with a tuning voltage of $[0, 3.3]$ V and a pull range of 25 MHz $\pm$ 25 ppm.

The 25 MHz VCTCXO clock signal is up-converted five times by the frequency multiplier (Analog AD9516-4), to generate the required 125 MHz reference. As mentioned earlier, the frequency multiplier/fanout chip has a set of internal registers, accessed through a serial link, that sets the phase detector sampling frequency and the frequency dividers. However, the PLL's frequency bandwidth is configured externally using passive components.

A user interface software supports in the configuration of the chip and generates a text file with all the chip internal registers setup. The configuration file is sent to the frequency multiplier via the SPI serial interface.

The most important aspects to disclose in the configuration of the AD9516-4 are the phase detector sampling frequency, set to 25 MHz, and the frequency dividers, which are configured to output a clock signal with a frequency equal to fives times the reference clock. The loop filter of the AD9516-4 PLL is a fourth-order PLL, as defined in Section 6.3.2, configured using external passive components, as illustrated in Figure 6.33.

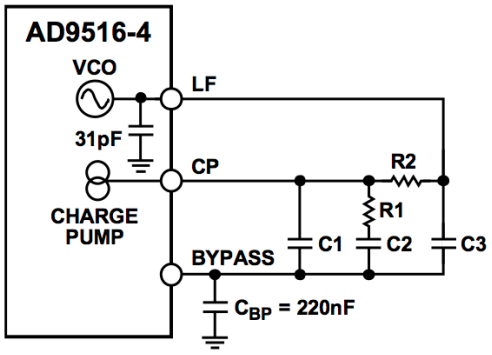


Figure 6.33: AD9516-4 External Loop Filter [93]

The AD9515 is configured to have a bandwidth of 40 kHz with a damping factor of 2. This results in attributing to the passive components the following parameters :

R1= 820 Ω, R2 = 390Ω, C1 = 68 pF, C2= 220 nF and C3 = 33 nF [93].

The bandwidth of the multiplier was been selected to be far way from DDMTD PLL cut-off frequency so that its contribution to the frequency response can be mitigated.

The combined gain of the VCTCXO plus DAC and the frequency multiplier/fanout chip, defined as K_i, is estimated experimentally using the same methodology used for the VXCO of the Helper PLL .

The measurements and their first-order fitting vector are illustrated in Figure 6.34.

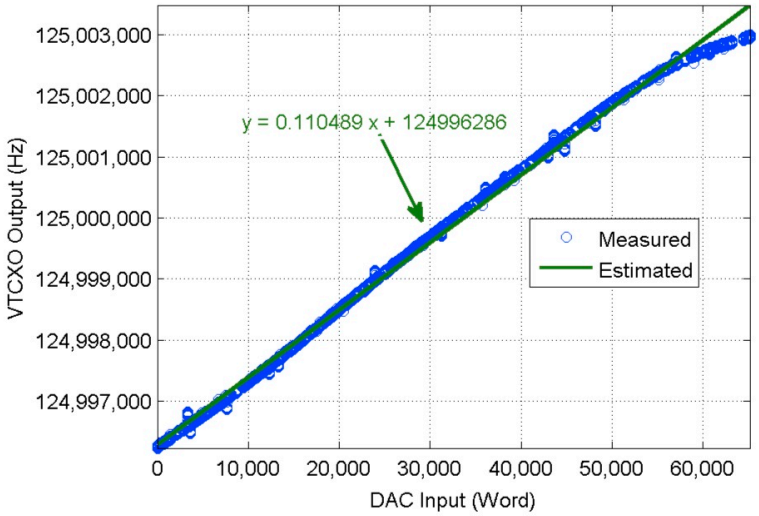


Figure 6.34: VCTCXO DAC/Frequency Characteristic

The estimated first-order curve that better approximates the DAC/Frequency characteristics of the VCTCXO is:

$$y = 0.1105x + 124996240 \tag{6.37}$$

Where **x** represents the DAC's input WORD and **y** the measured output frequency in Hz.

The VCTCXO gain, K_i, is given by the slope of Eq. 6.37. The K_i after converting to radians results in:

$$K_i = 0.1105 \cdot 2\,\pi = 0.6939 \text{ rads/sec/tick} \tag{6.38}$$

Digital Loop Controller

With the PLL parameters, K_p and K_i, estimated the loop controller can now be designed to give to the PLL different frequency dynamics.

As mentioned, the loop controller is responsible for tuning the VCTCXO and to aligned the VCTCXO with the reference clock while filtering the high frequency jitter components of the reference clock signal. In other words, the controller's bandwidth defines if the phase-noise contribution is due the reference clock or from the free-running clock.

Several loop controllers were designed with different parameters such as natural frequency, ω_n, and loop damping ζ. The loop controllers and their frequency response characteristics are shown in Table 6.5.

Table 6.5: DDMTD PLL - Digital Loop Controllers

Controller	**PLL Response H(s)**			**Loop Filter F(s)**	
	ω_n (rad/s)	ζ	ω_{-3dB}	K_p	K_i
Loop 1	144	2.9	136	0.927	22.67
Loop 2	653	0.52	191	0.74	464.4
Loop 3	1144	0.52	334	1.29	1422
Loop 4	1634	0.52	477	1.85	2902
Loop 5	2311	1.23	1051	6.21	5805

In addition, the phase-noise spectrum of the DDMTD PLL output clock was measured using the Agilent SSA 5052B.

Figure 6.35 shows the phase-noise spectra of the reference clock and the feedback clock.

The optimal cut-off frequency for the DDMTD PLL can be obtained, from Figure 6.35, to be around 20 Hz.

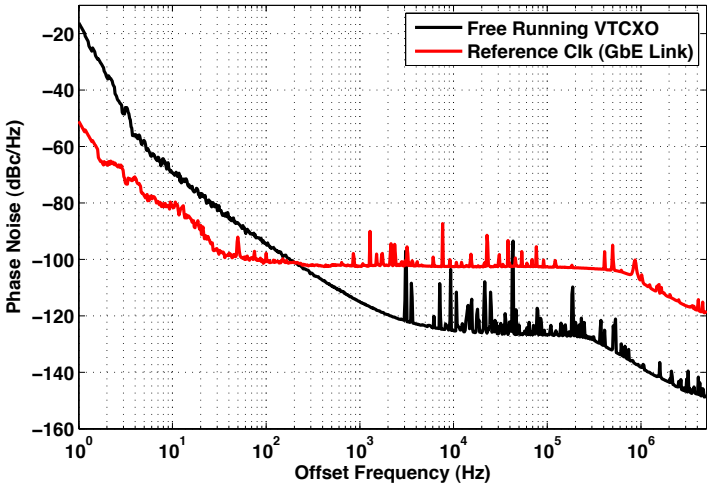


Figure 6.35: Phase-Noise Spectrum of the Free Running and Reference Clocks

So, for offset frequencies below the 20 Hz mark the contribution to the output phase noise comes from the reference clock, for upper frequencies the phase noise contribution is set by VCTCXO phase noise response.

Nonetheless, to achieve this optimal frequency selection the phase-noise spectrum floor of the phase detector must be less than -100 dBc/Hz. However, the DDMTD PLL has a noise floor of -80 dBc/Hz, as shown in Figure 6.37, thus, the optimal bandwidth for the closed-loop needs to be selected to minimise the contribution among the various sources of phase-noise.

The phase-noise spectra of the different PLL's loop controllers, described in Table 6.5, are illustrated from Figures 6.36 to 6.40.

Figure 6.36 shows the phase-noise spectrum of the "Loop 1" controller, this controller has the lowest bandwidth from the designed controllers.

The estimated phase-noise spectrum has similar frequency response as the measured spectrum. Some amplitude impairments are observed at the high frequency that are justified by a change in the free running oscillator conditions, due for instance to temperature variations.

Figure 6.37 shows the phase-noise spectrum of the "Loop 2" controller. In this

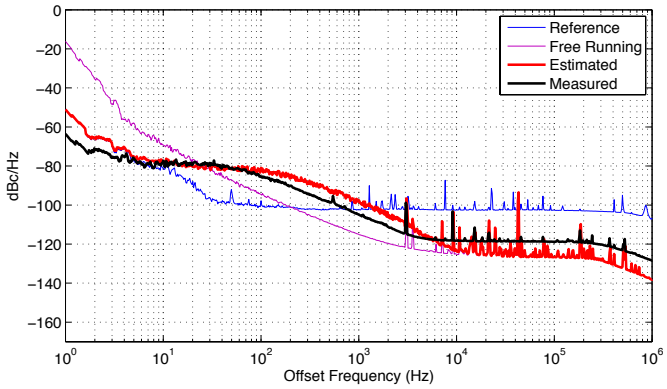


Figure 6.36: Phase-Noise Spectrum DDMTD PLL - Loop 1

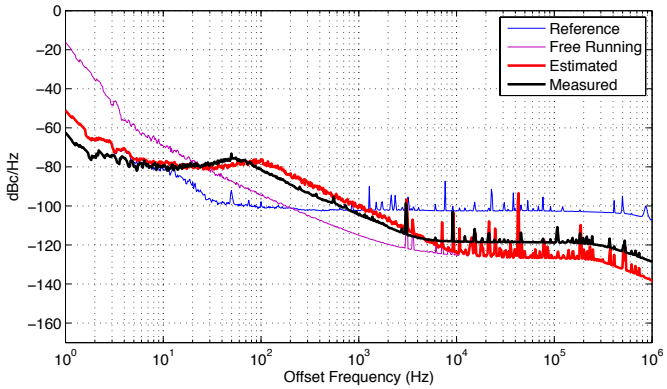


Figure 6.37: Phase-Noise Spectrum DDMTD PLL - Loop 2

measurement, the phase's detector locking noise floor starts to be observable around 10 to 100 Hz and is set at the -80 dBc/Hz y-axis. The estimated and measured phase-noise spectra have an identical frequency response.

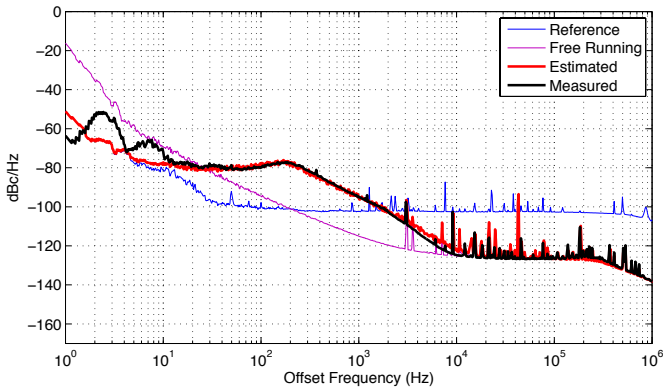


Figure 6.38: Phase-Noise Spectrum DDMTD PLL - Loop 3

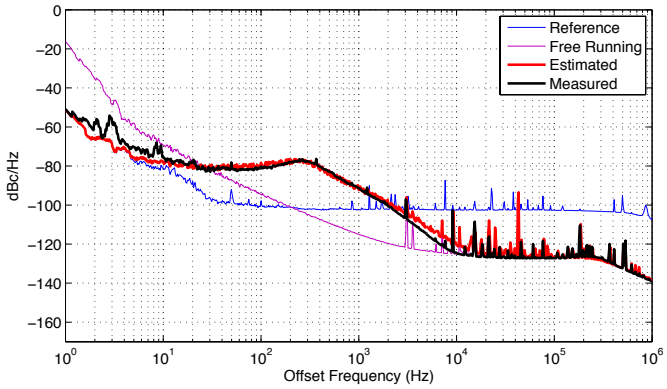


Figure 6.39: Phase-Noise Spectrum DDMTD PLL - Loop 4

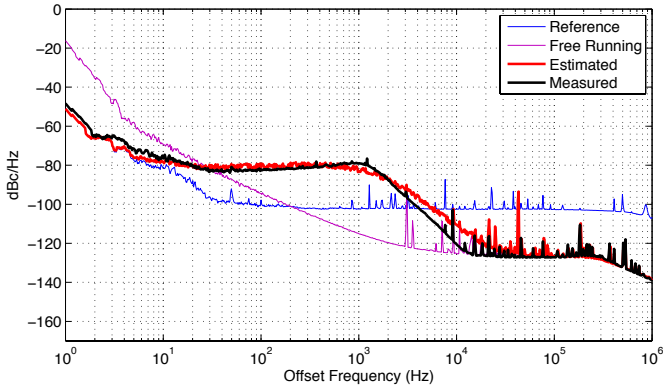


Figure 6.40: Phase-Noise Spectrum DDMTD PLL - Loop 5

From Figures 6.37 to 6.40, the loop controllers have a higher bandwidth resulting in a wider phase noise locking floor. From all measurements, the estimated phase-noise spectra show similar shape as the measured phase-noise spectra.

From the phase-noise spectra measurements, the RMS jitter is estimated by integrating the phase-noise amplitude around the offset frequencies of interest. This value quantifies the output's clock stability. The estimated RMS jitter the different loop controllers is shown in Table 6.6.

Table 6.6 shows the controller that generates the minimal RMS jitter, which is the one with less bandwidth label as "Loop 1". All the remaining loop controllers generate a higher RMS jitter values, this is due to the higher loop cut-off frequency. In the particular case of loop controller "Loop 5", the RMS jitter is nearly four times higher

Table 6.6: DDMTD PLL - RMS Jitter

Clock	Jitter (ps) [1 Hz − 4 MHz]
Reference	14
Feedback	114
Loop 1	2.6
Loop 2	3.1
Loop 3	6.7
Loop 4	6.3
Loop 5	8.3

than “Loop 1” .

Summary

The implementation of the DDMTD PLL was presented in this section. The DDMTD circuit showed very good performance as phase detector, with a phase-noise spectrum noise floor measured to be -80 dBc/Hz. This is a good result and when compared with others digital phase detectors, such as the one implemented for the Helper PLL design where the noise floor was measured to be ~(-40) dBc/Hz, the noise floor obtained by the DDMTD circuit doubles that value.

The controller design for the DDMTD PLL is able to output a clock signal with a RMS jitter of 2.6 ps meeting the initial RMS jitter specification defined for the WR project to be less than 4 ps.

6.5 DDMTD Phase shifter

In the application described earlier, the purpose of the DDMTD circuit was to measure the time difference between the reference and the feedback clock. The time difference measurement is used by the digital controller as the phase error reference that needs to be minimised.

However, some applications, such as in the WR project, require to phase shift the

feedback clock with picosecond time resolution. In these applications the DDMTD circuit can also play a very important role. The phase measurement of the DDMTD is done digitally, so it is very simple to effectively offset the measured phase error between the input and the feedback clock.

The schematic of the DDMTD phase shifter circuit is shown in Figure 6.41. With this circuit configuration, the PLL is able to phase shift the output clock with the time resolution in the DDMTD circuit, which is defined by the Helper PLL beat frequency, v_{beat}.

In the Helper PLL design presented earlier, the DDMTD circuit is able to phase shift the feedback clock with ~976 fs time resolution.

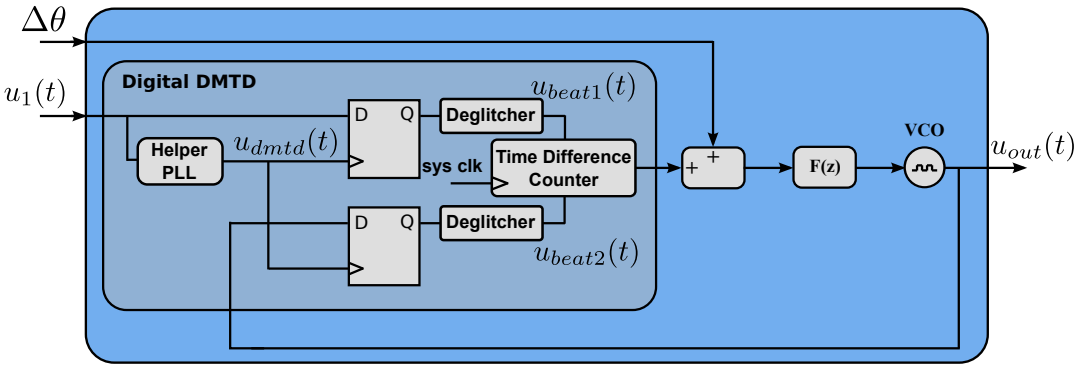


Figure 6.41: DDMTD Phase shifter circuit

The phase of the output clock signal $u_{out}(t)$ is given by:

$$\theta_{out}(t) = \theta_1(t) + \Delta\theta + \theta_{delay} \tag{6.39}$$

Where $\theta_{out}(t)$ is the phase of the output clock, $\theta_1(t)$ is the phase of the input clock, $\Delta\theta$ is the amount of phase shift in the output clock and θ_{delay} is the phase produced by the circuit's delay.

The θ_{delay} value is obtained by setting $\Delta\theta$ to zero and then measuring the time different between the edge transition of the input and output clock. This phase difference is subtracted from the final DDMTD phase value to obtain the actual

phase difference between the two clock signals.

The time difference between the recovery clock and the phase shifted clock are measured using the LeCroy WavePro 7300 series oscilloscope, the results are shown in Figure 6.42.

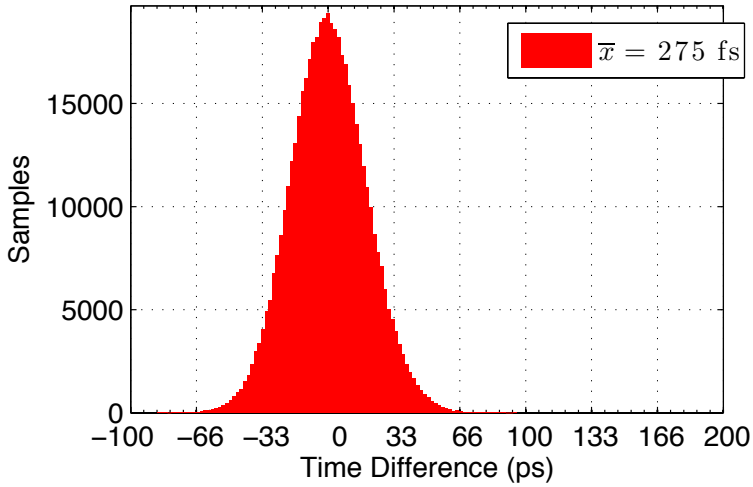


Figure 6.42: Time Shift Calibration

Figures 6.43-6.44 show in a histogram plot the time difference measurements between the recovered and feedback clocks time shifted by ~54 ps and ~108 ps, respectively.

These are equivalent to add an offset error to the time difference counters by 55 ticks and 110 ticks, respectively.

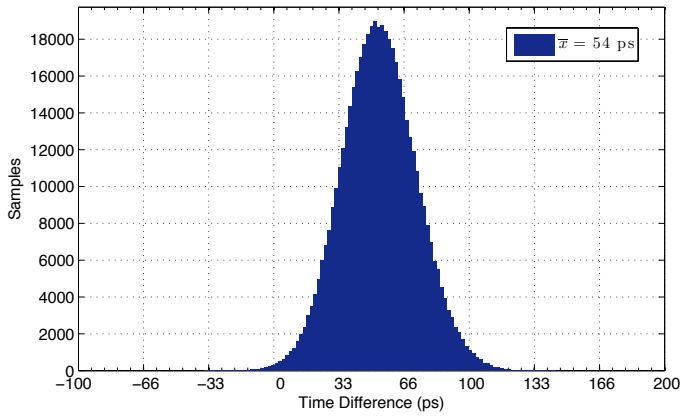


Figure 6.43: Time Shift by 54 ps

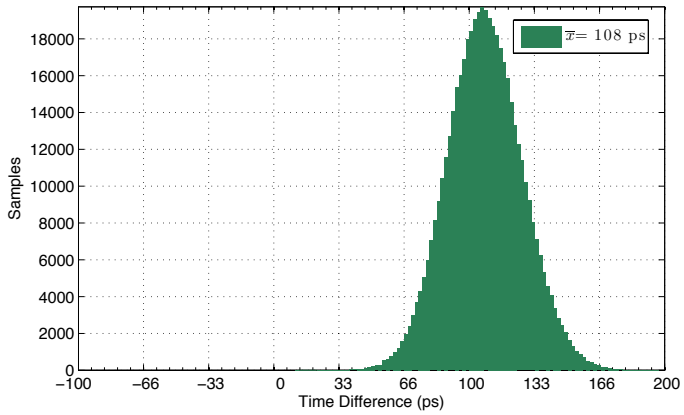


Figure 6.44: Time Shift by 108 ps

The standard deviation measured for each of each set of time shifted data is ~18 ps. The standard deviation is due to the clock stability (or timig jitter) in the input and output clocks, and also the time counter used in the time difference measurements. In practice, the amount of phase shift that the DDMTD PLL is able to perform is also dependent on the stability of these clocks.

6.6 Summary

In this chapter, the description of the hardware circuit board designed to allow the usage of Ethernet as data and timing network capable of achieving sub-nanosecond accuracy was given. In particular, a detailed description of the hardware circuits responsible for the treatment of the timing clocks was given. The timing FPGA board is responsible for the recovery of the Ethernet link carrier and to filter the jitter. The jitter in the recovery clock is filtered by PLL design that uses a proposed circuit, the DDMTD, as phase detector.

The DDMTD design requires a Helper clock, whose frequency is just slightly different from the reference clock. The frequency of the Helper clock sets the time resolution measured by the DDMTD circuit. The Helper clock was generated by the Helper PLL to have an offset frequency of 15.258 kHz with respect to the reference clock,

which is 125 MHz. This gives to the DDMTD's circuit a time difference measurement with a resolution of 976 fs .

The Helper PLL phase detector has an architecture that comprises a PFD circuit plus a time counter circuit to measure the PFD output duty cycle. The phase detector noise floor was measured to be approximately -40 dBc/Hz, a rather high level, but the Helper PLL can still output a reference clock to fulfil the required functionalities. The Helper PLL design section was concluded with the design of several digital controllers followed by their phase-noise spectra and RMS jitter measurements.

Next, the design of the PLL that uses as phase detector the DDMTD circuit was presented. The DDMTD PLL was able to filter the high frequency jitter from in the recovery clock. The recovery clock RMS jitter was initial measured as around 14 ps, after the DDMTD PLL the RMS jitter decreased to be less than 3 ps.

In addition, this chapter showed the DDMTD PLL functionality to operate as phase shifter with picosecond time resolution. This is capable of phase shifting the feedback clock with respect to the reference clock with a time resolution of 976 fs.

To conclude, the circuit design and implementation discussed in the previous chapters is used in the WR network to monitor and compensate for phase variations in the transmission link, thus, enabling the WR network to achieve the required sub-nanosecond timing accuracy. However, the discussion and demonstration of the timing achieved by the WR network is not reported in this work, since this topic is part of a group work. Nonetheless, the first installation of the WR network in a HEP application and the timing measurements that demonstrate the sub-nanosecond accuracy capability of the WR network is discussed in detail by [160].

Chapter 7

Distributing Radio Frequency Signals over the WR network

This chapter investigates the application of the White Rabbit network to distribute Radio Frequency (RF) signals over local area networks. Previous chapters have described and demonstrated how the White Rabbit network synchronises its nodes to a time-based measure to achieve sub-nanosecond timing accuracy. However, some equipment, such as the experiments detectors in the LHC, require synchronisation to the beam frequency [111]. The current RF synchronisation scheme is realised by a dedicated network that distributes the LHC clock generated by the RF system to the various access points and equipment. Such dedicated network increases the deployment and maintenance costs. A network that distributes the beam RF reference in the LHC is the Trigger, Time Control network (TTC) briefly discussed in Chapter 2.

This chapter proposes a new distribution RF system that takes benefit of the timing characteristics of the White Rabbit network. The main advantages of the proposed distribution system are the reduction of deployment and maintenance costs (due to the usage of non dedicated network) and its ability for frequency flexibility.

The major requirement in the proposed distributed RF system is to replicate an RF

signal image from a source node to different destination nodes using the minimal network bandwidth required.

Frequency flexibility is an effective characteristic of the proposed system architecture. Within this work scope, frequency flexibility means that the system can operate at a wide range of frequency bands without modifying analogue hardware of the system.

The distributed RF system is a simple and flexible architecture that results in a significant reduction in deployment and operation costs when compared with systems with similar requirements.

The first section of Chapter 7 discusses the various signal processing blocks of the proposed system. These processes are implemented in the proposed system architecture to reduce the network bandwidth necessary to distribute the bandpass signal.

Section 7.2 starts with a brief description of the Nyquist-Shannon sampling theorem followed by the bandpass sampling where their main contributors and applications are highlighted. In Section 7.3 the various analogue-to-digital (ADC) and the digital-to-analogue (DAC) converter architectures are reviewed. In addition, it describes how the quantisation process and clock jitter degrades the signal-to-noise ratio (SNR) of the bandpass signal.

Next, Section 7.4 discusses the quadrature modulation techniques and how they are applied in the proposed system to frequency shift the bandpass signal to baseband. This section also examines the CORDIC algorithm, an optimised method to generate a free running frequency reference in hardware. The reference generated by the CORDIC algorithm is used by the modulation processes. In Section 7.5, the filtering, decimation and interpolation digital processes are discussed.

Finally, Section 7.6, describes a simulation environment developed in MATLAB to analyse the performance of the distribution RF system. The performance is measured in terms of SNR, phase-noise spectrum and jitter.

The work presented in Chapter 7 has been published in:

- P. Moreira, P. Alvarez, J. Serrano, I. Darwazeh **Distributing RF Signals In An Ethernet Network**. IEEE International Frequency Control Symposium, June 2012.

- P. Moreira, P. Alvarez, J. Serrano, I. Darwazeh, M. Lipinski **Distributed DDS in a White Rabbit Network: An IEEE 1588 Application**. IEEE International Symposium on Precision Clock Synchronization for Measurement, Control and Communication, June 2012

- P. Moreira, I. Darwazeh **An FPGA implementation of the Distributed RF over White Rabbit**. IEEE International Frequency Control Symposium, June 2013.

The paper published at IFCS 2012 has been awarded the best student paper for the Group 5 - Timekeeping, Time and Frequency Transfer, GNSS Applications.

7.1 Proposed Distributed RF System Architecture

To distribute an RF signal to various receivers, the RF signal is converted to the digital domain and then it undergoes a set of digital processing manipulations before being transmitted to the nodes in an Ethernet packet. This section proposes a system architecture that aims to distribute an RF signal over the White Rabbit, an Ethernet network with sub-nanosecond timing accuracy [82]. The proposed system utilise an efficient methodology that allows network traffic from other sources without affecting the distribution of the RF signal.

The processing blocks in the transmitter node are illustrated in Figure 7.1. The transmitter node is defined as the network node where the analogue RF signal is sampled and converted to the digital domain. Details on the WR synchronisation engine have been already presented and discussed and are not presented in Figure 7.1 for the sake of clarity.

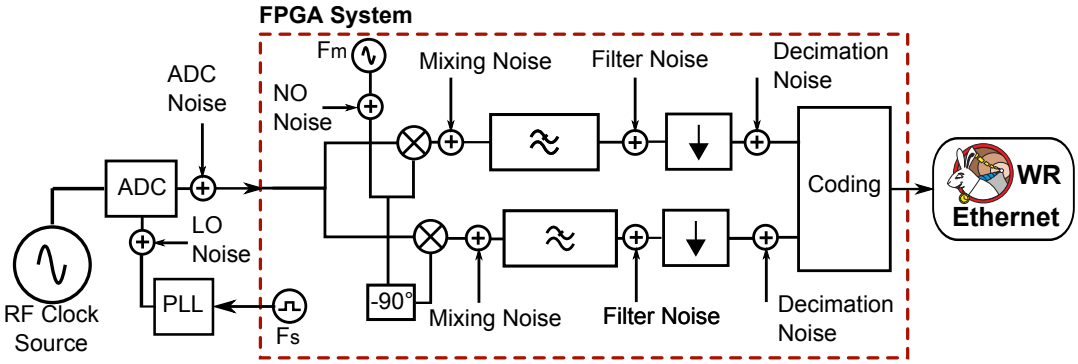


Figure 7.1: Distributed RF Architecture - Transmitter

The reference analogue RF signal is translated into the digital domain using an ADC. The ADC employs the bandpass sampling technique, discussed in Section 7.2.3, to digitise the RF signal and to down-convert the RF signal from its carrier frequency to the first Nyquist region of the ADC. This technique has the advantage of down-converting the RF signal without the need for analogue mixers and local oscillators, hence, it increases the input frequency range over which the system is able to operate. However, the bandpass sampling requires a bandpass filter prior to the sampling converter to reduce the presence of undesired frequency components, which without the filter, would be aliased to the ADC first Nyquist zone and corrupt the digitised RF signal [112]. The bandpass filter must have a high suppression in the stop-bands to reduce the aliased noise. The system is designed to distribute efficiently RF signals with a signal bandwidth below 2 MHz. This signal characteristic covers a wide range of applications, such as the distribution of the LHC Bunch clock or the distribution of atomic clock references such as Rubidium or Caesium clocks.

The digitised RF signal is split into its in-phase and quadrature (IQ) baseband components by mixing the digitized signal with its centre frequency, F_m. The mixing clock reference with a frequency, F_m, is synthesised by the FPGA using the CORDIC algorithm [113], as illustrated in Figure 7.1.

Following the mixing process, a digital low-pass filter removes the high frequency harmonics generated by the IQ mixing process. The cut-off frequency of the low-pass filter is chosen to be, typically, two times higher than the bandwidth of the baseband

signal [112].

The sampling frequency of the IQ signals can be further reduced to the frequency rate necessary to retain the phase and frequency information carried by the IQ signals.

Generally, the new frequency rate is chosen to be two times higher than the cut-off frequency of the adjacent low-pass filter. The reduction of the sampling frequency is implemented in the decimation block.

The decimation operation is realised by keeping a certain amount of samples and removing the remaining ones so that the desired sampling frequency is reached. Following the decimation operation the IQ data is packed and sent to the receivers in an Ethernet packet containing a (Distributed RF) DRF frame.

At the receiver node, the IQ data is received from the transmitter and the inverse process described earlier is applied to reconstruct the analogue RF signal.

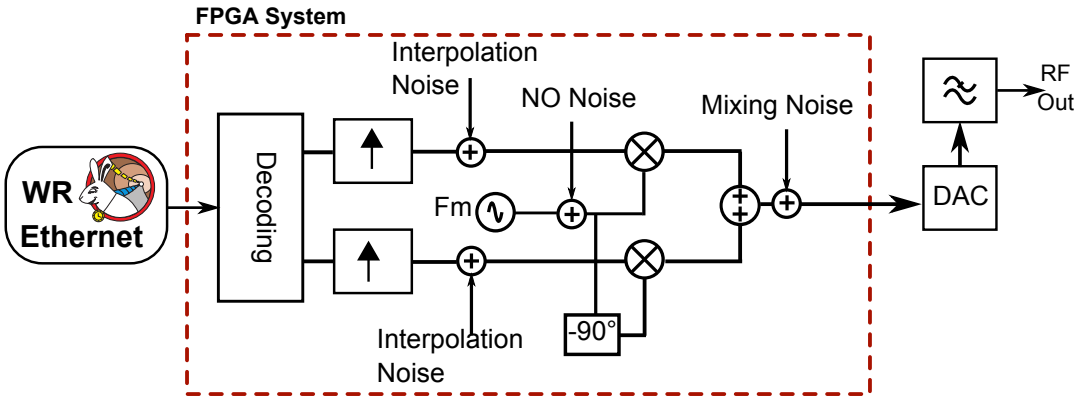


Figure 7.2: Distributed RF Architecture - Receiver

The received data is fed to an interpolation filter designed to reconstruct the data to the original sampling frequency of the transmitter's ADC. After the interpolation filter, the data is once again digitally mixed with the same clock frequency reference employed at the transmitter node. The resulting interpolated signals are then summed with each other, resulting in a signal identical in phase, frequency and amplitude with the digital signal obtained after the conversion process in the transmitter node.

A PLL-based frequency synthesizer is used at the receiver to up-convert the output signal to the passband frequency of the input RF signal. Implementing PLL-based frequency synthesizer has the main advantage of broad spectral RF spectral quality.

The following sections discuss in detail the conversion and processing blocks that formulate the proposed system. They aim to serve as a ground to develop an accurate model of the proposed system, which is analysed in a computer-based simulations presented in Section 7.6.

7.2 The Sampling Theorem

Sampling is the process of converting a continuous variable, such as amplitude, into a discrete variable that is quantised to a digital number and can be stored in a memory bank. In a sampling process, the sampling time, that is the time step in which a continuous variable is converted into a digital number, is an independent variable that defines the memory requirements. If a continuous variable is sampled with a very small sampling time the processing requirements in terms of storage and processing would be too high, as more memory is required to store the sampled signal.

However, if low sampling rate is chosen a phenomenon known as aliasing, described in Section 7.2.1, occurs that introduces frequency errors in the sampled stream.

The sampling theorem, first presented by Shannon and Nyquist [114], plays a fundamental role in current signal processing and communications applications. The sampling theorem defines the minimum rate between samples required to convert a continuous bandpass signal to the digital domain without loss of information. The theorem also defines that the signal reconstruction consists of filtering the impulse train of samples by an ideal low pass filter [114].

First, let's consider a band-limited signal, x(t), with spectrum as shown in Figure 7.3.

Now, let's define s(t) as an impulse function occurring at every time instant kT.

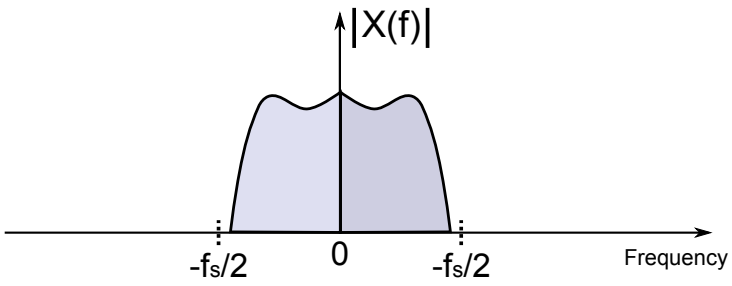


Figure 7.3: The spectrum of the band-limited signal x(t)

$$s(t) = \sum_{k=-\infty}^{+\infty} \delta(t - kT_s) \tag{7.1}$$

Where $\delta(t)$ is a Dirac impulse described as:

$$\delta(t) = \begin{cases} 1 \quad , \quad t = 0 \\ \\ 0 \quad , \quad t \neq 0 \end{cases} \tag{7.2}$$

The sampled function, which results from multiplication of x(t) by a Dirac impulse train, is obtained as:

$$\begin{aligned} x_s(t) &= x(t)s(t) \\ &= \sum_{k=-\infty}^{+\infty} x(kT_s)\delta(t - kT_s) \end{aligned} \tag{7.3}$$

The Fourier transform of $x_s(t)$ is given as:

$$X_s(f) = \int_{-\infty}^{\infty} x_s(t)e^{-j2\pi ft}dt \tag{7.4}$$

The frequency domain of signal $x_s(t)$ is:

$$X_s(f) = \sum_{k=-\infty}^{\infty} x(kT_s)e^{-j2\pi kfT_s} \tag{7.5}$$

which leads, by definition, to:

$$X_s(f) = \frac{1}{T_s} \sum_{k=-\infty}^{\infty} [X(f - kf_s)] \tag{7.6}$$

Spectrum $X_s(f)$ shows spectra copies of the input signal at multiples of the sampling frequency, as illustrated in Figure 7.4.

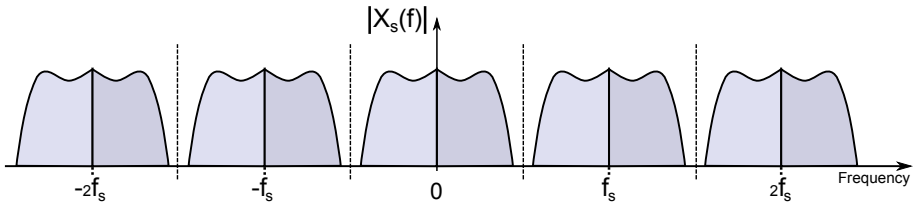


Figure 7.4: The spectrum of the band-limited signals of x(t) after sampling

The sampling theorem for band-limited signals is stated in two parts [115]:

- A band-limited signal of finite energy, which has no frequency components higher than $f_s/2$ Hz, is completely described by specifying the values of the signals at instants of time separated by $1/T_s$ seconds.
- A band-limited signal of finite energy, which has no frequency components higher than $f_s/2$ Hz, may be completely recovered from knowledge of its samples taken at rate of $1/T_s$ samples per second.

The sampling rate of f_s samples per second, for a signal bandwidth of $f_s/2$ Hz, is called the Nyquist rate.

7.2.1 Aliasing

If the frequency components of the continuous signal disobey the constraints stated by the sampling frequency, then the frequency components located higher than the Nyquist frequency will appear at frequencies between DC and the Nyquist frequency. This phenomenon is known as aliasing, meaning that frequencies above the Nyquist

rate take on the alias of the frequency range below the Nyquist frequency, as shown in Figure 7.5.

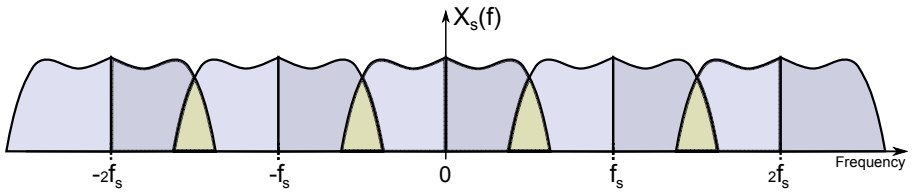


Figure 7.5: The spectrum of the band-limited signals of x(t) after sampling with Aliasing

Due to this aliasing behaviour, the bandpass sampling was developed that is further discussed in section 7.2.3.

7.2.2 Reconstruction

To reproduce a continuous signal from the samples an interpolation process is necessary.

The signal in the time-domain, $x_s(t)$, is reconstructed using the Inverse Fourier Transform of $X(f)$, by definition:

$$x_s(t) = \frac{1}{2\pi} \int_{-\infty}^{+\infty} X(f) e^{j2\pi ft} df \tag{7.7}$$

For a signal defined between $-1/(2T_s) \leq X(f) \leq 1/(2T_s)$, Eq.7.7 results in:

$$x_s(t) = \frac{1}{2\pi} \int_{-1/(2T_s)}^{+1/(2T_s)} \left[\frac{1}{T_s} \sum_{k=-\infty}^{\infty} X(f - kf_s) \right] e^{j2\pi ft} df \tag{7.8}$$

that is,

$$x_s(t) = \frac{1}{2\pi T_s} \sum_{k=-\infty}^{\infty} x(kT_s) \left[\int_{-1/(2T_s)}^{+1/(2T_s)} e^{j2\pi ft} df \right] \tag{7.9}$$

The reconstruction to x(t) is:

$$x_s(t) = \sum_{k=-\infty}^{\infty} x(kT_s) \frac{sin(\pi/T_s(t - kT_s))}{\pi/T_s(t - kT_s)} \tag{7.10}$$

and

$$\frac{sin(\pi/T_s(t - kT_s))}{\pi/T_s(t - kT_s)} = sinc(\pi/T_s(t - kT_s)) \tag{7.11}$$

Unfortunately, the ideal reconstruction cannot be implemented since it is non-causal and has an infinite delay. It implies that each sample contributes to the reconstructed signal at almost all instants of time, requiring summing an infinite number of terms. Instead, some sort of approximation of the sinc function has to be used but that leads to an interpolation error.

7.2.3 Bandpass Sampling

Bandpass sampling, also known as under-sampling, is the process of sampling an analogue signal at a lower rate than that stated by the Nyquist sampling theorem. For bandpass signals, the sampling rate can be as low as two times the bandwidth of the signal rather than twice the highest component of the low-pass signals, as comprehended by the Sampling Theorem.

The bandpass sampling technique effectively performs a frequency mixing on the analogue signal, to down-convert the signal to the frequency region defined between DC and the Nyquist frequency, also referred as the first Nyquist zone [112].

The bandpass sampling procedure is due to the Aliasing phenomena described in Section 7.2.1. The images of the sampled signal are duplicated at the different sampling frequency regions, the Nyquist zones. By choosing a proper sampling frequency, frequency translation can be implemented without the typically usage of analogue mixers to down-convert the carrier frequency, as illustrated in Figure 7.6. This process of down-converting the passband signal down to the first Nyquist zone can also

cause spectral reversal.

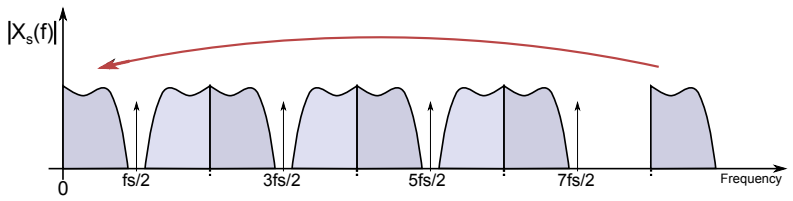


Figure 7.6: Bandpass Sampling - Aliasing

There are additional restrictions on the sampling frequency to avoid aliasing in the case of bandpass sampling as described by Vaughan [112].

From Figure 7.6 is noticeable that spectral placement is realised when down-converting a bandpass signal. The result can be normal spectral placement where the upper frequency of the signal is placed at the highest part of the Nyquist region, or it can be inverse spectral placement where the highest frequency of the signal is placed at the lower part of the Nyquist region [116].

7.2.4 Selection of the sampling frequency

To ensure that the spectrum does not overlap and corrupts the desired signal, the bandpass sampling requires that the ADC sampling rate is at least twice the bandwidth of the bandpass signal and twice the highest frequency of the down-converted bandpass signal. That is, the sampling rate, F_s, must obey [112]:

$$\frac{2f_U}{k} \leq Fs \leq \frac{2f_L}{(k-1)} \tag{7.12}$$

where k is an integer given by

$$2 \leq k \leq \left\lfloor \frac{f_U}{(f_U - f_L)} \right\rfloor \tag{7.13}$$

The bandpass signal is defined between lower signal frequency component, f_L, and its upper frequency component, f_U. The operation $\lfloor x \rfloor$ represents the largest integer

not greater than x.

In addition, bandpass sampling requires that the ADC is able to operate effectively at the highest frequency component of the bandpass signal. Typically, high attenuation requirements are specified for the analogue bandpass filters to prevent distortion of the bandpass signal due to the presence of nearby frequency components. However, in this application the bandpass filters can be disregarded because the reference signal is obtained from a very pure and well defined (frequency-wise) clock signal.

7.3 Data Conversion

The correct selection of analogue-to-digital converters (ADC) and digital-to-analogue converters (DAC) is essential to achieve good performance in the proposed distributed RF system architecture.

7.3.1 ADC Architectures

The analogue signal is converted to the digital domain by means of an ADC. This is the first stage of the Distributed RF System. The ADC plays an important role in the proposed system, since it is responsible for the first injection of distortion into the sampled signal. Its performance dictates the noise transfer between the analogue domain and the digital domain. In particular, characteristics such as bandwidth, resolution and the linearity of the ADC are of major importance.

ADCs can be implemented using different architectures with trade-offs between the conversion resolution, sampling rate and power as depicted in Figure 7.7.

Figure 7.7 shows that the flash architecture is the one that can reach higher sampling frequencies, however, with the lower bit resolution. The ADC flash architecture employs 2^n comparators that sample the analogue input, at the same time instant, to achieve a n-bit resolution. In addition, the flash architecture also requires additional logic to perform encoding. The output digital code from the comparators is typically

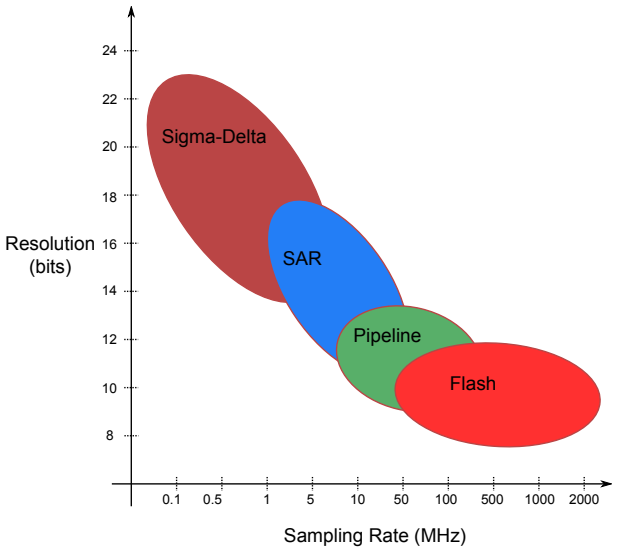


Figure 7.7: ADC Resolution vs Speed [117]

described as a thermometer code. The comparators and the decoder logic can work at very high speeds awarding the flash architecture the preferred ADC architecture when high resolutions and high conversion rates are required.

However, the chip area and the power consumption increases exponentially as the resolution increases, which are the main disadvantages of this architecture.

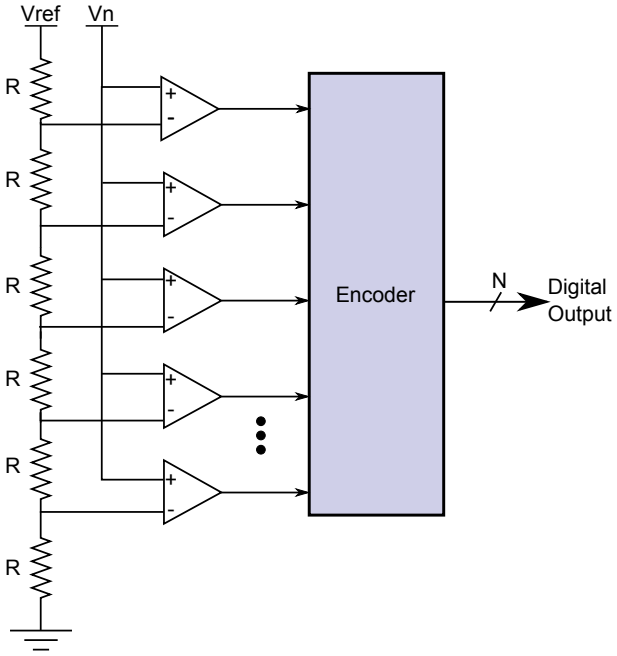


Figure 7.8: Flash ADC Architecture

The successive-approximation register (SAR) employs a feedback technique that uses a DAC in a feedback loop as shown in Figure 7.9. This technique recursively compares the analogue input with the output signal generated by the DAC and it is based on a binary search algorithm. The binary search algorithm first starts by comparing the most significant bit of the DAC and weights it against the input signal. This process repeats until the least significant bit is weighted. After the last bit is compared the ADC latches the generated digital code to the ADC's output.

This architecture requires N iterations to output a digital code. The overall accuracy and linearity of the SAR architecture is dictated mainly by the internal DAC.

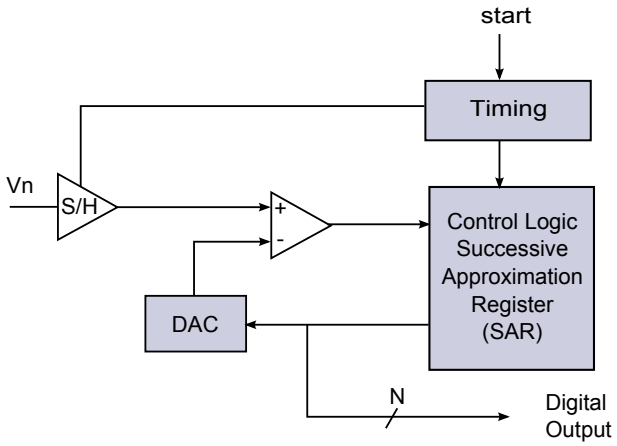


Figure 7.9: SAR ADC Architecture

The sub-ranging architecture, illustrated in Figure 7.10, provides an alternative when the sampling rate can be low and a high bit resolution is desired.

The operation scheme of the sub-ranging architecture is described as follows. In the first cycle, the input signal is sampled and held and the N1-MSBs are determined. The MSB conversion is then converted back to an analogue signal that is subtracted from the input signal. Subsequently, the resulting difference is applied to a second N2-bit resolution ADC, which results in the LSB of the input signal. Both codes are then latched to obtain the final digital code.

The pipeline architecture is constructed by a set of serialised sub-ranging ADC stages, as shown by the block diagram in Figure 7.11.

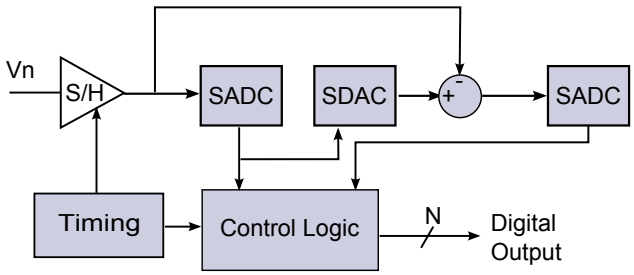


Figure 7.10: ADC Sub-Ranging Architecture

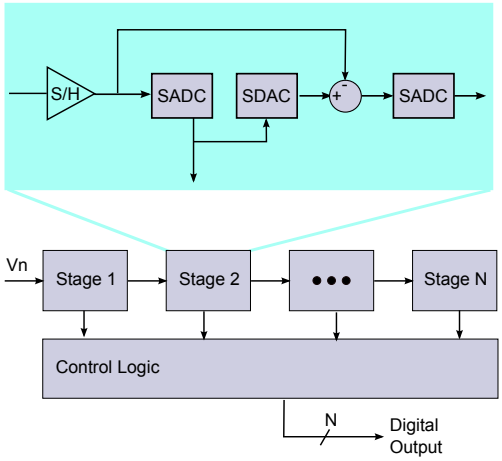


Figure 7.11: Pipelined ADC Architecture

Pipeline ADCs have become an appealing solution in many applications due to their balanced requirements in terms of size, speed, resolution, power dissipation and ultimately the cost.

Sigma Delta ($\Sigma\Delta$) ADCs are typically used in applications that require low cost, low power, low bandwidth and high resolution requirements. The sigma delta ADC architecture consists of a 1-bit ADC, 1-bit DAC, one integrator and digital filtering blocks as depicted in Figure 7.12.

The $\Sigma\Delta$ architecture over-samples the analogue signal, that is it samples the input signal at a rate much higher than the rate imposed by the sampling theorem.

An advantage of oversampling is the improvement in SNR of the digitised signal, as the quantisation noise is spread over a wider frequency range [118]. A key element of this ADC is the integrator located before the 1-bit ADC. This integrator works as a

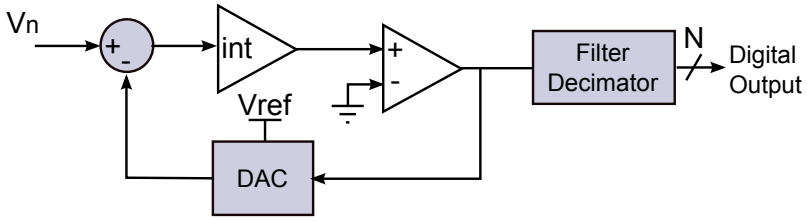


Figure 7.12: ΣΔ ADC Architecture

low-pass filter for the bandwidth of interest that is to the Nyquist frequency and as a high pass filter to the quantisation noise generated by the 1-bit ADC. This effect called noise shaping is illustrated in Figure 7.13.

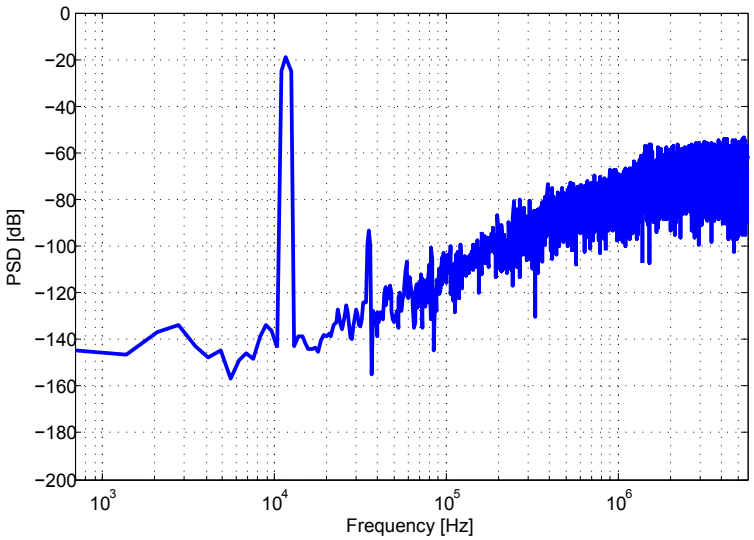


Figure 7.13: Noise Shaped Spectrum of a ΣΔ Modulator

A digital processing block is used in the ΣΔ ADCs to accomplish two things. First, it removes the noise generated by the sampling process that is located outside the frequency band of interest. Second, it reduces the output data rate to the minimal data rate required to represent the frequency range of interest.

For high resolution applications, the ΣΔ ADCs may not provide sufficient noise shaping. However, increasing the number of integrators in the modulator adds the required noise shaping but at a cost of more complex design. A second-order ΣΔ ADC schematic is shown in Figure 7.14.

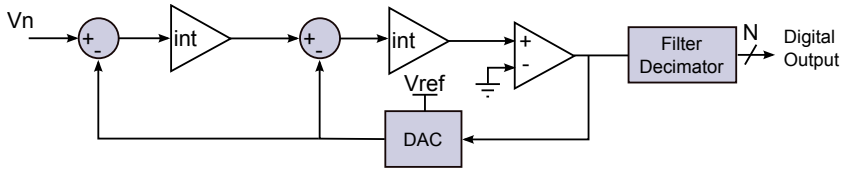


Figure 7.14: Second-order ΣΔ ADC Architecture

ADC Noise Specifications

The ADC is a source of both static and dynamic errors that degrades the quality of the analogue signal being sampled. These static errors are specified by diverse variables such as the offset, gain errors and the non-linearity of the converter. The non-linearities in the converters are typically quantified as the differential non-linearity (DNL) or integral non-linearity (INL).

The offset error is specified as constant difference between the output and the input signal. The gain error is defined as the difference between the input and the output signal after the offset error has been corrected [119].

The DNL is measured in relation to the least significant bit (LSB) and it describes the worst-case difference between the widths of actual codes with the ideal one. A DNL specification higher than 1 LSB does not guarantee a monotonic behaviour of the converter.

The INL is the integral of the DNL errors. The IEEE Standard 1241 [119] states that "the integral non-linearity is the difference between the ideal and measured code transition levels after correcting for static gain and offset". The DNL and INL errors can be observed as frequency spurs in the frequency response of the digital signal.

Nonetheless, other specifications exist that characterise the dynamic behaviour of a converter. Among them are the signal-to-noise ratio (SNR), signal-to-noise and distortion ratio (SINAD), total harmonic distortion (THD) and spurious-free dynamic range (SFNR).

The SNR defines where the noise floor of the converter is, and it is commonly referred to as quantisation noise. The SNR of an ADC is calculated next for a single tone

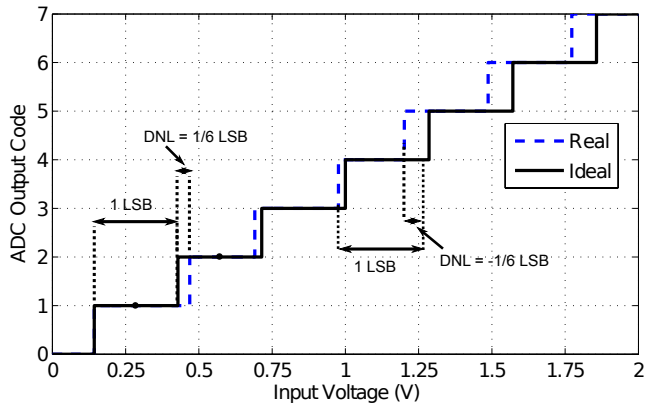


Figure 7.15: DNL of an ADC

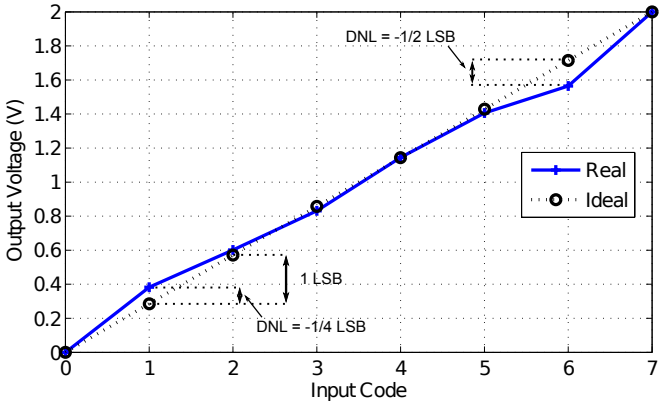


Figure 7.16: INL of an ADC

signal.

Considering a sine wave as the input signal where the only noise source present in the process is the quantisation noise, the maximum SNR for an N-bit ADC can be calculated. The maximum peak-to-peak value of the input signal is $2^N V_{LSB}$, which has an RMS value of $2^N V_{LSB}/2\sqrt{2}$. The quantisation noise has been calculated as being $V_{LSB}/\sqrt{12}$, thus the maximum SNR may be expressed as:

$$\text{SNR} = 20\log\left(\frac{2^N V_{LSB}/2\sqrt{2}}{V_{LSB}/\sqrt{12}}\right) \tag{7.14}$$

resulting in:

$$\text{SNR} = 6.02N + 1.76\ dB \tag{7.15}$$

Eq. 7.15 can be solved to calculate the effective number of bits (ENOB) required to obtain a maximum SNR of the converter.

$$\text{ENOB} = \frac{\text{SNR}_{max} - 1.76}{6.02} \tag{7.16}$$

However, it is not just due to quantisation noise that the SNR decreases. The variation in time of the exact sampling instant also degrades SNR. This variation is named aperture jitter. Aperture jitter can be caused by several reasons such as jitter in the sampling clock or due to the sampling switching process that does not occur at the precise time.

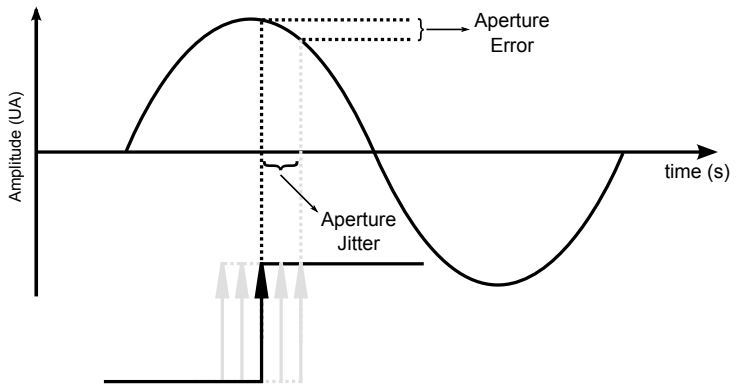


Figure 7.17: Aperture Jitter in the AD process

Aperture jitter inevitably causes a phase modulation on the digitized signal therefore increasing the amount of noise in the sampled signal. If there is sample-to-sample time variation, aperture jitter, the instant the sample and hold components acquire the signal corresponds to an amplitude error, which is defined as aperture error, this behaviour is illustrated in Figure 7.17 for clarity. The aperture jitter is usually measured in RMS picoseconds. The amplitude of the associated output error is related to the rate-of-change of the analogue input. For any given value of aperture jitter, the aperture jitter error increases as the input dv/dt increases. The effects of

phase jitter on the external sampling clock produce exactly the same type of error.

The aperture error ultimately reduces the accuracy of an ADC, and it limits the maximum frequency of the input signal. The maximum frequency of a single tone signal is defined as $x(t) = V_{in}sin(2\pi f_o t)$ and is calculated next.

The input signal maximum slew rate is:

$$\frac{dx(t)}{dt}_{max} = 2\pi f_o V_{in} \tag{7.17}$$

In order to mitigate the effect of aperture error in the conversion process its value must be less than 1/2 LSB of the converter:

$$\begin{aligned} e_{ap} &\leq \frac{1}{2}LSB & (7.18)\\ &\leq \frac{1}{2}\frac{2V_{in}}{2^n - 1} & (7.19) \end{aligned}$$

So the maximum frequency of the input signal is given as:

$$f_{max} = \frac{1}{2\pi t_a(2^n - 1)} \tag{7.20}$$

The SNR due to aperture jitter is calculated as:

$$\text{SNR} = 20\log_{10}\left(\frac{1}{2}\pi f_s t_a\right) \tag{7.21}$$

Where f_s is the sampling frequency while t_a is the aperture jitter of the ADC. Both SNRs due to quantisation error and aperture jitter can be summed to obtain the overall ADC SNR.

The spurious free dynamic range (SFRD) is calculated as the ratio of the RMS value of *tan* input sine wave to the RMS value of the largest spur observed in the frequency domain using an FFT plot, as shown in Figure 7.18. SFDR is important in communication applications that require maximizing the dynamic range of the

converter.

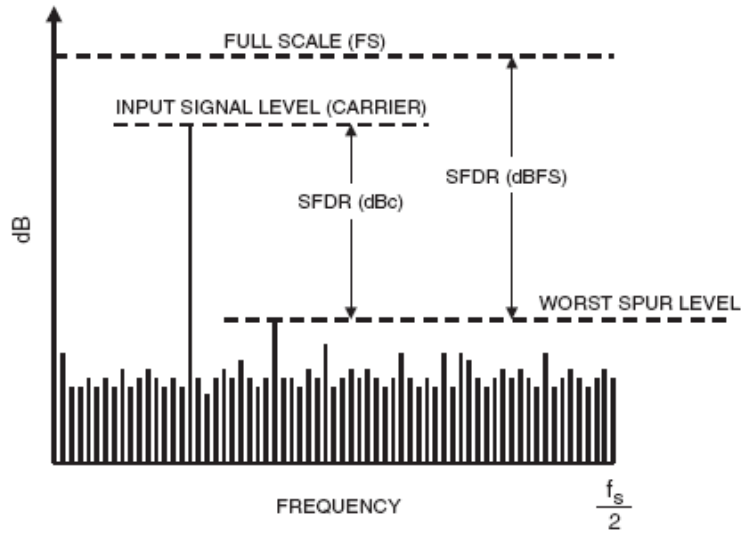


Figure 7.18: ADC SFRD Measurement [120]

7.3.2 DAC Architectures

As opposed to the ADC, the digital signal is converted to the analogue domain, either a voltage or current quantity, by means of a DAC. The DAC dictates the amount of noise transferred from the digital to the analogue domain. The analogue signal generated by the DAC suffers from various erratic sources such as offset error, scale factor error, and non-linearity.

The String DAC architecture has the simplest structure from all DAC architectures. It consists of a set of resistors connected in series and a set of switches. Each switch is connected between the end of each resistor and the analogue output, as shown in Figure 7.19. The analogue output is taken by actuating at the control of the switch, which is realised by a block of digital logic. This operation results in a monotonic output. It shows a linear behaviour when all the resistors have the same characteristics. However, for a N-bit resolution it requires 2^N resistors and the same amount in switches, which makes the String DAC architecture almost unused for medium to high-resolution applications.

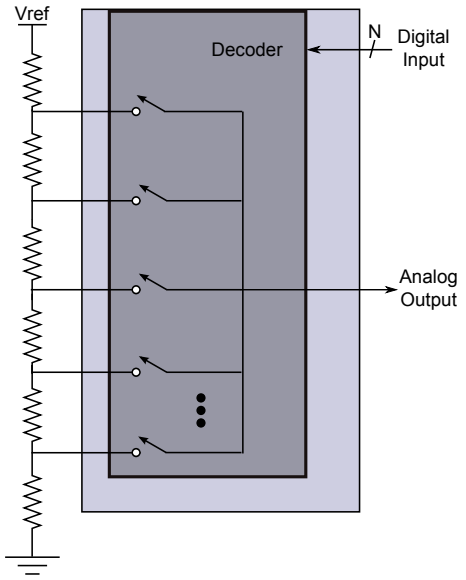


Figure 7.19: DAC String Architecture

The Segmented DAC is a variation of the string architecture that aims to reduce the number of resistors needed to achieve a N-bit resolution. However, it requires a complex digital block to decode and actuate in the switching stage.

There are many variations of the segmented architecture but typically this architecture is composed by two stages each one actuating in the segments that go from a bit range of the MSB to the remaining range belonging to the LSB, as shown in Figure 7.20. For example, in a 16-bit segmented DAC divided in two 8-bit stages each stage requires 2^8 resistors and switches for a total in the DAC of 512 resisters and switches. As for a string DAC, a 16-bit resolution requires 2^{16} resisters and switches. This clearly shows the advantage of the segmented DAC against the string DAC in terms of required chip area.

The $\Sigma\Delta$ DAC consists of various digital blocks as illustrated in Figure 7.21. The $\Sigma\Delta$ DAC has a very similar architecture to the $\Sigma\Delta$ ADC, however, the subtraction and integration operations are done digitally, while the remaining operations are realized in the analogue domain. The comparator is a digital block that extracts the most significant bit from the integrator's output. It is followed by an analogue low-pass filter on the output to obtain the average value of the comparator and remove the

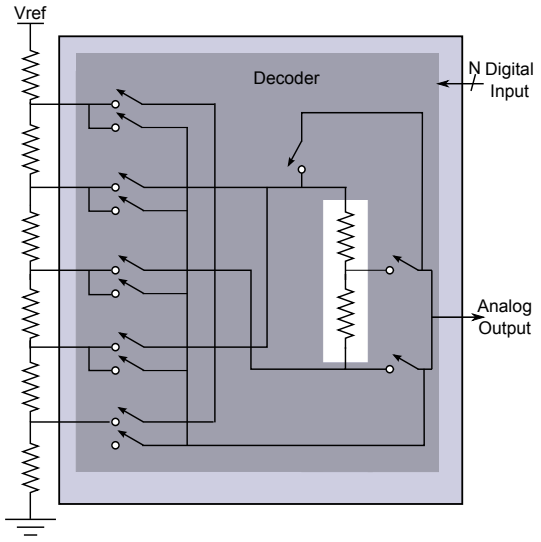


Figure 7.20: DAC Segmented Architecture

high frequencies due to the quantisation noise.

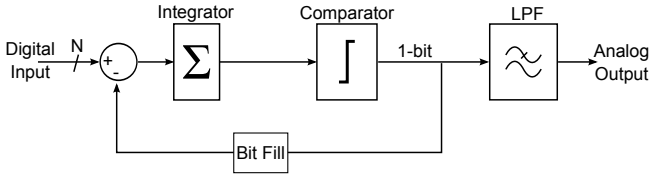


Figure 7.21: DAC $\Sigma\Delta$ Architecture

Most DAC voltage source architectures are used in relatively low speed applications while current source architectures are mainly used in high-speed applications, such as such as video, and telecommunications. The main reason for this is that current source architectures enable an inherited fast operation without a need for a voltage buffer. However, its performance is usually limited by linearity problems that are more evident when the output frequency increases.

Mismatch between the transistors in the current sources causes errors in the bit weights and limits the static linearity of the DAC.

DAC Noise Specifications

For high data rates and high resolution, the generality of the frequency spurs are due to the analogue errors in the digital to analogue conversion and not to errors due to the digital source. Like for the ADCs, DACs have also specifications that characterise the erroneous quantities generated by the converters. Some are similar to the DAC like the offset error, gain error, DNL and INL.

The offset error and gain error have equal definitions as for the ADC described earlier. However, both DNL and INL have slight differences. The DNL is the worst-case deviation from the ideal LSB step between the lower and upper codes. For a higher DNL specification than 1 LSB the DAC does not guarantee a monotonic behaviour.

The INL is specified as the worst error from a straight line that represents the ideal converter transfer function. Some of the most important dynamic specifications are the settling time and the output slew rate. The first one is measured as the time interval from its initial value until its output reaches its final value within a defined error band, as illustrated in Figure 7.22.

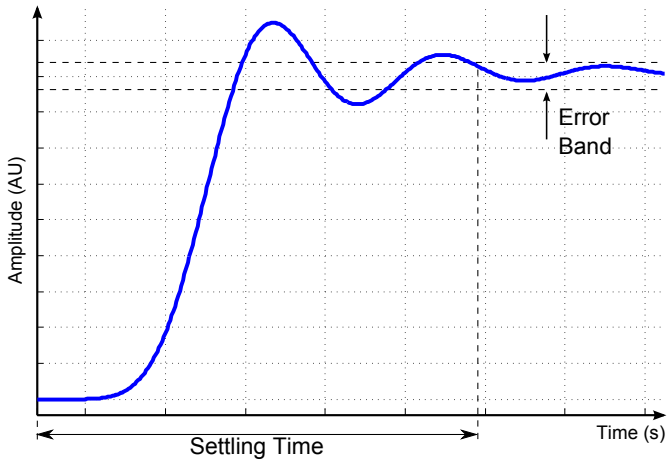


Figure 7.22: DAC Settling Time

The slew rate is the rate of change of output voltage/current. Another DAC specification mostly important for RF synthesis applications is the glitch impulse. The glitch impulse is the measure that describes the uncontrolled movement of the DAC

output during a code transition. This uncontrolled movement is typically described as overshoot or undershoot. The ADC specifications have more relevance when the signal output frequency increases.

7.4 Digital Quadrature Demodulation/Modulation

In the proposed distributed RF architecture, following the sampling of band-limited RF input signal, a digital IF signal is generated. This digitized signal is mixed with a reference signal to form the baseband in-phase and quadrature-phase (IQ) components. This mixing block is referred to as the digital quadrature demodulator.

Digital quadrature demodulation was first discussed in a paper dating back to 1983 [121]. Quadrature demodulation is a necessary signal processing process in almost every communication system. The demodulator creates a complex signal, with amplitude and phase information, of the input real signal, which facilitates the subsequent signal processing stages and extraction of information such as spectral analysis. However, in conventional quadrature demodulation the circuit is implemented using RF components that inevitably affects the performance of the demodulator. Some of the errors found in conventional RF components are gain balance, quadrature-phase balance, DC offsets between others [83]. These inherent issues in the RF components make it often problematic to match the frequency characteristics of the two mixing stages that create the two complex components.

Using digital quadrature demodulation the issues presented above are mitigated. In digital quadrature demodulation the input RF signal is sampled with an ADC. In many of the applications the input signal is bandpass sampled to down convert the RF signal to the Nyquist frequency range, as show in Figure 7.23. The digital signal is then mixed by a cosine and a sine signal with a frequency equal to the fundamental frequency of the input digital signal.

The two resulting signals are a representation of the in-phase and quadrature components of the input real signals. Each signal contains a double frequency component

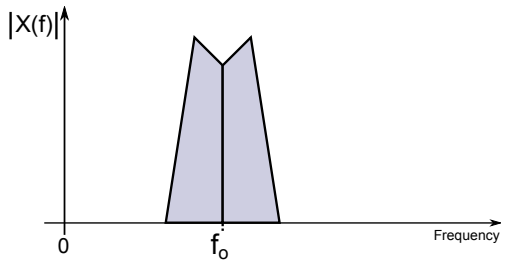


Figure 7.23: Input Signal - Frequency response

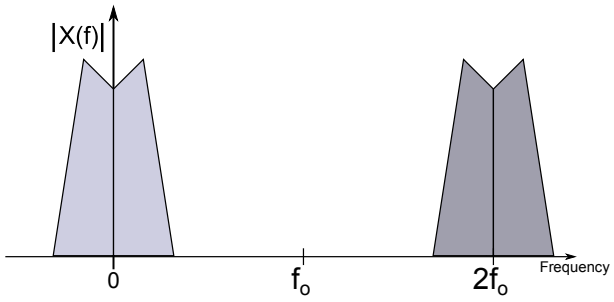


Figure 7.24: IQ demodulation - Frequency Response

generated by the mixing process. Amplitude and phase balance are guaranteed by using this digital demodulation methodology. The process is described as follows [121]:

Let's define a bandpass signal as r(t) with centre frequency f_o. The signal can be written in terms of its in-phase and quadrature components as:

$$r(t) = x_i cos(2\pi f_o t) + x_q sin(2\pi f_o t) \tag{7.22}$$

Thus, the baseband in phase component, $x_i(t)$, can be recovered by multiplying $r(t)$ by $cos(2\pi f_o t)$ and low-pass filtering the result to remove the sum-frequency terms that result from the multiplication. Hence, the baseband quadrature component, $x_q(t)$, can be recovered by multiplying r(t) by $sin(2\pi f_o t)$. Low-pass filters are then used to eliminate the unwanted side-bands. The resulting signals are the in-phase and quadrature components of the input signal.

Advantages of this digital demodulator are that the demodulator requires little tuning and maintenance and can be easily reproduced on different digital platforms.

Following the filtering process, the sampling rate of the complex signals can be reduced to the required sampling rate defined by the Nyquist sampling theorem.

Nonetheless, a component of great concern in digital quadrature demodulation is the numerical oscillator required to down-convert the intermediate frequency to baseband.

One solution to digitally implement a complex demodulator is the Mixer-Free Quadrature demodulation [122]. In digital mixer-free quadrature demodulation the signal is sampled at four times the centre frequency f_0. The digital signal is then input to the digital mixing stage that is operating at carrier centre frequency f_0. So the mixing frequency is at a quarter of the converter's sampling frequency, resulting in exactly four samples in the mixing waveform per cycle. This makes the implementation of the mixing stage very straight forward and optimised for hardware designs. The mixing stage is implemented by multiplying the digital signals with $\{-1, 1\}$. The in-phase mixing results in the sequence of multiplications $\{+1, +1, -1, -1\}$ while the quadrature sequence is $\{+1, -1, -1, +1\}$, as illustrated in Figure 7.25.

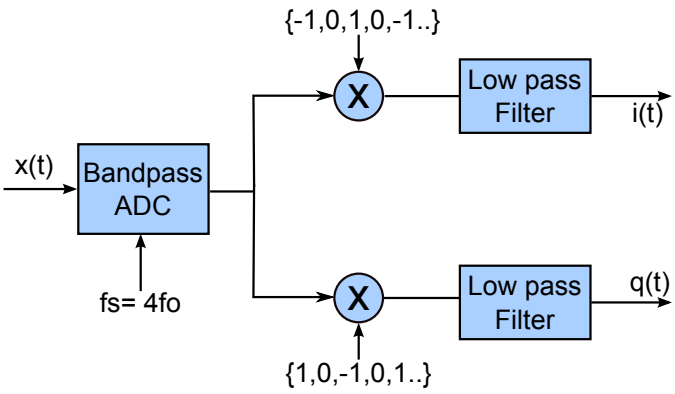


Figure 7.25: Digital Mixer Free I/Q Demodulation

The main disadvantage of this method is that the sampling converter has to operate at four times the centre frequency, which requires a corresponding increase in power dissipation, an on going concern in digital applications.

In addition, the mixer-free demodulator requires changing the sampling frequency to maintain the four times operating frequency criterion, which removes to a certain

extent a degree of flexibility in the operating frequency range of the input signal, but an ideal solution for the contrary. Nonetheless, to increase the design flexibility the CORDIC-based digital quadrature mixer is a promising architecture. This architecture digitally generates the cosine and sine functions of the numerical oscillator but optimises the operating speed and digital hardware requirements.

7.4.1 CORDIC Algorithm

The numerical oscillator used in both the demodulator and modulator can be implemented efficiently in digital hardware platforms, such as FPGAs, using the Coordinate Rotation Digital Computer (CORDIC) Algorithm, increasing the flexibility and the input frequency range of the input signals.

The CORDIC algorithm was first presented by Volder [123], in 1959, as a hardware efficient way to compute trigonometric functions to be used in real-time navigation. Later in 1971, Walther [124] extended the original CORDIC theory to compute elementary functions.

The CORDIC algorithm is derived from the general rotation transformation:

$$x' = xcos(\varphi) - ysin(\varphi) \tag{7.23}$$

$$y' = ycos(\varphi) - xsin(\varphi) \tag{7.24}$$

Which rotates a vector $[x, y]^T$ in a Cartesian plane by an angle φ. The equations above can be rewritten as:

$$x' = cos(\varphi)[x - ytan(\varphi)] \tag{7.25}$$

$$y' = cos(\varphi)[y - xtan(\varphi)] \tag{7.26}$$

if the rotation angles are restricted so that $tan(\varphi) = \pm 2^{-i}$, the multiplication by the

tangent term is simply implemented by a shift register operation. A shift register operation is an efficient and fast operation in most digital hardware platforms.

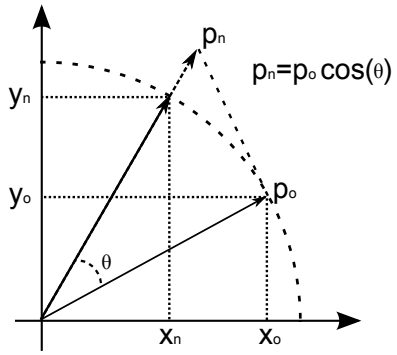


Figure 7.26: CORDIC rotation [113]

An arbitrary angle can be obtained by a series of successive micro rotations. To optimise the equation above, Volder [125] stated that the decision would be to decide the direction of rotation so that the term cos(φ) remains constant for each iteration, that is cos(φ) = cos(-φ). This results in:

$$x_{i+1} = K_i[x_i - y_i d_i 2^{-i}] \tag{7.27}$$

$$y_{i+1} = K_i[y_i + x_i d_i 2^{-i}] \tag{7.28}$$

Where $d_i \in \{-1, +1\}$ and

$$K_i = cos\left(tan^{-1}(2^{-i})\right) = \frac{1}{\sqrt{(1 + 2^{-2i})}} \tag{7.29}$$

This constant gain K_i can be removed from the iteration process, yielding to a pair of equations with simple shift-add operations. The gain can be inserted at the end of the iteration process and is calculated as follows:

$$K_n = \prod_{i=0}^{n-1} = \frac{1}{\sqrt{(1 + 2^{-2i})}} \tag{7.30}$$

The gain K_n is approximately 1.647 as the number of recurrences reach infinity [126].

The decision of the angle's rotation direction is based on the following equation:

$$z_{i1} = z_i - d_i tan^{-1}(2^{-i}) \tag{7.31}$$

The angles $tan^{-1}(2^{-i})$ are precomputed and stored in the form factor.

The rotated angle is given as:

$$\theta = \sum_{i=0}^{n-1} d_i tan^{-1}(2^{-i}) \tag{7.32}$$

The CORDIC algorithm can be implemented in two modes, the rotation mode or the vectoring mode. In the rotation mode, the Cartesian coordinates of an initial vector and an angle of rotation are given and the cosine and sine components of the vector after the initial vector is rotated the predefined angle are calculated.

During each iteration, a rotation decision is made to minimise the computed residual angle. This decision is done based on the sign of the residual angle after the iteration.

For the rotation mode, the CORDIC equations are:

$$x_{i+1} = x_i - y_i d_i 2^{-i} \tag{7.33}$$

$$y_{i+1} = y_i + x_i d_i 2^{-i} \tag{7.34}$$

$$z_{i1} = z_i - d_i tan^{-1}\left(2^{-i}\right) \tag{7.35}$$

With the condition that,

$$d_i = \begin{cases} -1 & , \quad z_i \leq 0 \\ 1 & , \quad z_i > 0 \end{cases} \tag{7.36}$$

In the vectoring mode, the Cartesian coordinates of an initial vector are given and its magnitude and angular components are computed by rotating the input vector to

the x-axis. For the vector mode, the condition for the direction of d_i is:

$$d_i = \begin{cases} -1 & , \quad y_i \leq 0 \\ 1 & , \quad y_i > 0 \end{cases} \tag{7.37}$$

The CORDIC algorithm as initially defined is limited to angles lying between $\frac{\pi}{2}$ and $-\frac{\pi}{2}$. This limitation is due to the fact that it is required that the tangent in the first rotation is 20. However, using the symmetry properties of the sine and cosine functions in the different quadrants, different angles can be calculated by a simple rotation to settle the vector in the required angle by the algorithm.

CORDIC Accuracy

The accuracy of the CORDIC algorithm can be degraded due to diverse sources of errors [113]. The first source of errors are due to the limited resolution and the finite number of iterations, as a result of the angle approximation error in the linear calculation of the rotation vectors. Second, is the rounding or truncation of the output rotation angle that increases successively after each iteration of the micro-rotations. Finally, the scaling of each iteration also adds errors to the final result. The scaling is due to the fixed-point implementation that results in a rounding error [84].

The floating point implementation of the CORDIC algorithm can mitigate the rounding problems and increase the accuracy of the algorithm. However, floating point implementation has increased complexity when compared with fixed point implementation with impact on the chip area and power consumption. Hence, there is a trade-off between the cost of the implementation and the numerical accuracy required for the application. Walther [124] described in his paper that the degradation of accuracy in the CORDIC algorithm is bounded, and that the effect of the magnitude of errors depends on the function being computed. In addition, Walther [124] stated that for an n-bit word length of the input vector and input angle, then n+1 iterations

leads to results with n-bit accuracy.

The work from Hu [84] brought an analysis of the error constraints due to the calculation of the rotation vectors for both fixed and floating point implementations. In a later work, Kota and Cavallaro [127] analysed the resulting error bound for fixed-point implementation of the CORDIC algorithm in vectoring mode. In addition, they also discussed practical implementation in hardware platforms to increase accuracy in the algorithm without further increase in area cost. In fixed-point implementation to achieve n-bit accuracy for rotation mode, the representation of x and y is $(n + 2 + log_2(n))$ [128]. For the vectoring mode the computation of the angle θ requires a representation of $(n + log_2(n))$ [127] .

Work from Park et al [129] compared the accuracy of ROM-based phase to amplitude converters with the CORDIC algorithm. The work estimated that in terms of hardware costs the CORDIC algorithm becomes more effective when the required accuracy is higher than 9-bit. The CORDIC algorithm is very effective for solutions where quadrature mixing is performed [130]. However, the problem of this algorithm is the long latency time due to its iterative nature. In Chapter 8 hardware implementation strategies to mitigate the latency issue in the CORDIC algorithm are discussed and their performance evaluated.

7.5 Filtering

Digital filters are implemented in the proposed system to eliminate the double frequency components created by the demodulation stages. However, it needs to retain the phase and frequency characteristics of the low frequency signal bandwidth.

The filtering process typically requires the implementation of multiple multiplications that increase chip area and rapidly deplete the available resources in the FPGA hardware. This also reflects in an increase of the power consumption of the design. For this reason filtering is usually the process that becomes the resources bottleneck of the design.

Many types of filters are presented in the literature for quadrature modulation applications. The infinite-duration impulse response (IIR) filters can be used to suppress the double frequency signal. It has the advantage of requiring low computational resources. However, as Raders points out in his paper [131], there is some phase distortion with IIR filters that can be negligible for many applications but others would be harmfully affected by this distortion.

In the case of employing finite-duration impulse response (FIR) filters the phase distortion would be insignificant at the expense of an increase in the hardware resources required for their direct implementation.

In many systems, as the distributed RF proposed in this work, the complex components after the filtering process occupy a small fraction (depending on the application) of the available bandwidth. Therefore, the sampling rate can be reduced to the minimum required by the sampling theorem. Decimation is the process that reduces the effective data rate. A straightforward way to implement a decimation process is to discard intermediate values in between samples.

Interpolation, is the process that effectively increases the sampling rate of a data stream. There are many mathematical functions that can perform the interpolation operation such as sinc-based or polynomial-based interpolations [132]. Such functions show different advantages and disadvantages especially due to the computational resources required in their implementation in hardware. For instance, the polynomial based interpolation minimises hardware complexity while yielding satisfactory results [133].

Filtering, decimation and interpolation processes can be implemented using multirate FIR structures to optimise the hardware resources in implementing both processes.

7.5.1 Polyphase Structures

Polyphase structures are efficient architecture for implementing multi-rate systems where sampling rate conversion and filtering operations are aggregated reducing the

hardware resources required to implement both operations. For a decimation by a factor M, a N-tap filter running at the sampling rate, F_s, is the same to M M/N-tap sub-filters running at F_s/M. Conversely, an interpolation of M, a N-tap filter running at the sampling rate, F_s, is equivalent to M N/M-tap sub-filters running at F_s/M.

A block diagram illustrating the filtering and decimating processes by a factor M of the signal $x(n)$ is shown in Figure 7.27.

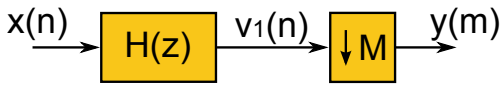


Figure 7.27: Polyphase Filter - Decimator

The block diagram, shown in Figure 7.27, consists of a filter, H(z), running at a F_s sampling rate, followed by a decimation block that reduces the sampling rate from F_s to F_s/M. The filter H(z) is a low-pass filter that prevents aliasing by limiting the frequency of the input signal to be at least less than $F_s/2M$. The decimation is commonly implemented by discarding M-1 samples from every M samples.

Formally, the process of down-sampling can be expressed as [134]:

$$y[n] = v[nM] \tag{7.38}$$

$$y(n) = \sum_{k=0}^{K} h(k)x(nM - k) \tag{7.39}$$

where h(k) are the coefficients of a K-order low-pass filter. Nonetheless, the filtering and decimation processes described earlier are highly inefficient because the filter is computing values at the sampling rate that are being discarded by the decimation process.

An optimised structure is to first decimate the data stream and then filter while maintaining the relevant frequency information of the initial structure. The Noble identities [134] describe when it is possible to reverse the order of the filtering and down-sampling.

The Noble identities proves the equivalence of each pair of block diagrams, as illustrated in Figure 7.28.

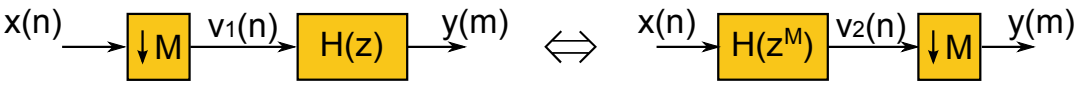


Figure 7.28: Nobel Identity Decimator

From the left side of the Figure 7.28 it is obtained:

$$Y(z) = H(z)V_1(z) \tag{7.40}$$

$$= H(z)\frac{1}{M}\sum_{k=0}^{M-1} X(e^{-j\frac{2\pi}{M}k\frac{1}{z^M}}) \tag{7.41}$$

from the right side:

$$Y(z) = \frac{1}{M}\sum_{k=0}^{M-1} V_2(e^{-j\frac{2\pi}{M}k\frac{1}{z^M}}) \tag{7.42}$$

$$V_2(z) = X(z)H(z^M) \tag{7.43}$$

that is,

$$Y(z) = \frac{1}{M}\sum_{k=0}^{M-1} X(e^{-j\frac{2\pi}{M}k\frac{1}{z^M}})H(e^{-j\frac{2\pi}{M}Mk\frac{1}{z^{\frac{M}{M}}}}) \tag{7.44}$$

$$= H(z)\frac{1}{M}\sum_{k=0}^{M-1} X(e^{-j\frac{2\pi}{M}k\frac{1}{z^M}}) \tag{7.45}$$

Hence, this demonstrates the Noble identity for decimation.

In Figure 7.29 a possible Noble identity implementation of the filter $H(z)$ is illustrated.

The polyphase components of $H(z)$, $H_p(z)$ where $p \in [0, M-1]$, operate at a lower rate and the output of the filters are kept to calculate the output of the filter and

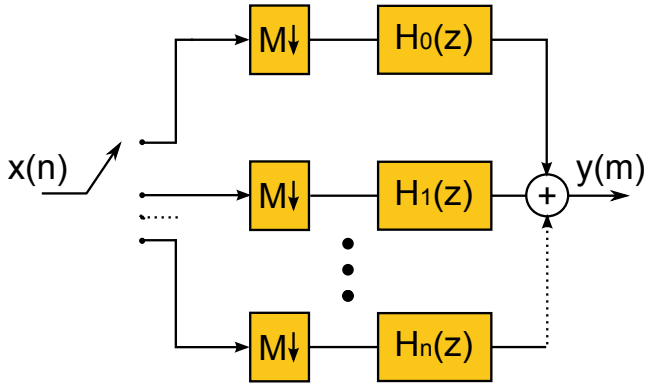


Figure 7.29: Polyphase Filter - Decimator Implementation

decimation process. This configuration yields an optimised structure with advantages particularly in terms of hardware implementations as it requires less power consumption and resources.

The interpolation is the process of up-sampling and low-pass filtering the signal to effectively increase its sampling rate without changing its spectral components.

The up-sampling process is realised by the insertion of zeros between the samples of the input signal. Effectively, if it is desired to up-sample a signal by a factor L, then its is required to add $L-1$ zeros between every sample of the signal.

Formally, the process of up-sampling is expressed as:

$$y[n] = \begin{cases} x[\frac{n}{L}] & \text{, when } \quad \frac{n}{L} \\ 0 & \text{, other cases} \end{cases} \tag{7.46}$$

A block diagram of the process of up-sampling and filtering the signal $x(n)$ by a factor L is shown in Figure 7.30.

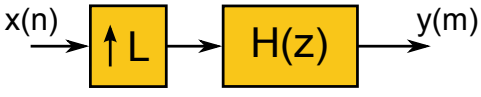


Figure 7.30: Polyphase Filter - Interpolator

The interpolation process, depicted in Figure 7.30, is also very inefficient because

filter $H(z)$ operates in a signal composed mostly by trails of zeros. However, as in the decimation process, the interpolation process can be rearranged to a more efficient structure described by its Nobel identity.

The Noble identity for the interpolation is illustrated as follows:

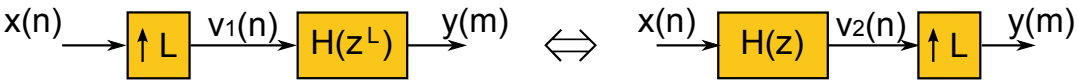


Figure 7.31: Interpolator Nobel identities

From the left side of the Figure 7.31,

$$Y(z) = H(z^L)V_1(z) \text{ where } V_1 = X(z^L) \quad (7.47)$$

$$= H(z^L)X(z^L) \quad (7.48)$$

From the right side,

$$Y(z) = V_2(z^L) \text{ where } V_2 = H(z)X(z) \quad (7.49)$$

$$= H(z^L)X(z^L) \quad (7.50)$$

So the Noble identity is established for the interpolation process.

Figure 7.32 illustrates a possible polyphase implementation of the filter $H(z)$ for an interpolation process. The polyphase components of $H(z)$, $H_p(z)$ where $p \in [0, L-1]$, operate at a lower rate. They do not operate when the input signal is up-sampled with the trails of zeros.

In the direct implementation of a N-order filter, a number N operations of multiplications in both decimation and interpolation processes are required to produce an output sample. However, in the case of polyphase implementation, the filtering process is decomposed into sub-filters of order proportional to the down-sampling or up-sampling factors operating at the lower sampling rate, thus allowing sharing of

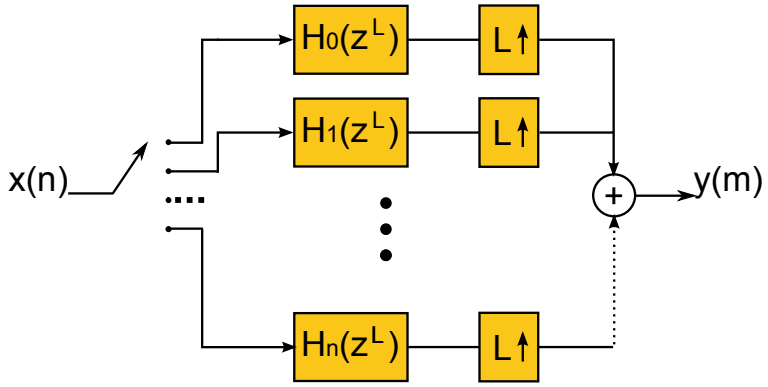


Figure 7.32: Polyphase Filter - Implementation Interpolator

computational resources such as the multipliers.

7.5.2 Multiplierless Filters

A direct hardware implementation of a filter typically requires a number of multipliers proportional to the filter's order that results in significant resource utilisation and increased power consumption.

In addition, using a direct implementation the maximum operation rate of the filter is limited due to the critical timings paths † created to route the multiplication operations and subsequent summing operations.

In fixed-coefficient filter applications that require a high operation rate, a multiplierless filter architecture can be implemented where each filter coefficient is coded by a Canonic Signed Digit (CSD) representation [136].

CSD is a fixed-point representation that has been widely explored in digital signal processing applications thanks to its improved performance in terms of area and power consumption [137]. The technique was first introduced by Reitwiesner [138] in 1960. With the CSD representation, non-zero digits can be represented by a number of sums and differences of power-of-two which is formally known as a radix-2 signed-digit code [139]. A coefficient h_i can be expressed using the radix-2 signed-digit code

†The critical path is the maximum delay of any combinational logic in the design. The maximum frequency at which the design can be clocked has an inverse relationship with the critical path delay [135].

form as:

$$h_i = \sum_{k=0}^{M_i-1} d_k 2^{-p_k} \tag{7.51}$$

where $d_k \in \{-1, 0, 1\}$ and $p_k \in [0, L]$ and L+1 is the word length of the coefficient. For $h_i \neq 0$ then M_i is the number of non-zero digits in h_i. A CSD representation is defined as the minimal representation for which two non-zero digits d_k are adjacent. The CSD representation of a coefficient is unique and can be realized by a simple algorithm that converts from the binary representation to the CSD representation [140] [141] [142]. As the power-of-2 multiplication can be obtained using simple shifts, the multiplication by each filter coefficient may be realised efficiently in hardware using structures employing adders and hard-wired shifts.

CSD representation for filters can increase the speed and decrease the complexity of the hardware implementation by a significant amount, Shyh-Jye et al [143] reported an area reduction by 70% when comparing with the direct approach. It reduces the necessity of designing full multipliers while still maintaining the required response characteristics of the filter.

7.6 Simulation Environment

This section presents a simulation model of the distributed RF system developed to evaluate the performance of the proposed architecture through the implementation of different combinations of transmitter's and receiver's model parameters as discussed in the previous section. This includes the effect of various filters architectures, with various orders and fixed-point realizations.

The filter must determine the dynamic range and the desired precision of input, intermediate, and output signals in a design implementation to ensure that the algorithm fidelity criteria are met. Typically in VLSI systems, such as FPGAs, the filtering processes are implemented using fixed-point arithmetic due to fewer hardware resources

need when compared to the floating-point arithmetic.

The simulation model, designed using MATLAB simulation environment, is also used as a model to analyse the hardware requirements in the implementation of the proposed system architecture. The distributed RF system is to be implemented in an FPGA, which is further described in Chapter 8.

To evaluate the amount of noise introduced due to the several signal processing stages, the SNR is calculated between the input digital signal and the reconstructed signal on the receiver. Furthermore, the evaluation of the clock frequency stability of the reconstructed RF signal is realised by calculating its phase-noise spectrum. Phase-noise spectra show useful phase information as they determine the presence of undesired spurs close to the carrier frequency generated by the various processing stages that the signal undergoes.

Figure 7.33 illustrates a simplified block diagram of the Distributed RF system model.

The simulation environment consists of different modules each one implementing a different functionality in the distributed RF architecture. The functionality of each module in the simulation environment is briefly described below:

- **Signal Generator** - This module generates an RF signal. The generated signal can have different modulations, such as QAM, PSK, with flexible bandwidth. In addition, in case of generating a single tone signal reference, specific phase-noise can be added to the reference signal.

- **ADC and DAC** - This module is responsible for sampling and reconstructing the reference signal. The dynamic and static characteristics of the converters are parameters of this module. Thus, achieving a better approximation to the effective operation of the real device.

- **CORDIC Algorithm** - This module is able to simulate the generation of the mixing frequency using the CORDIC algorithm. In this module, parameters such as number of iterations, number of bits employed in the quantisation of

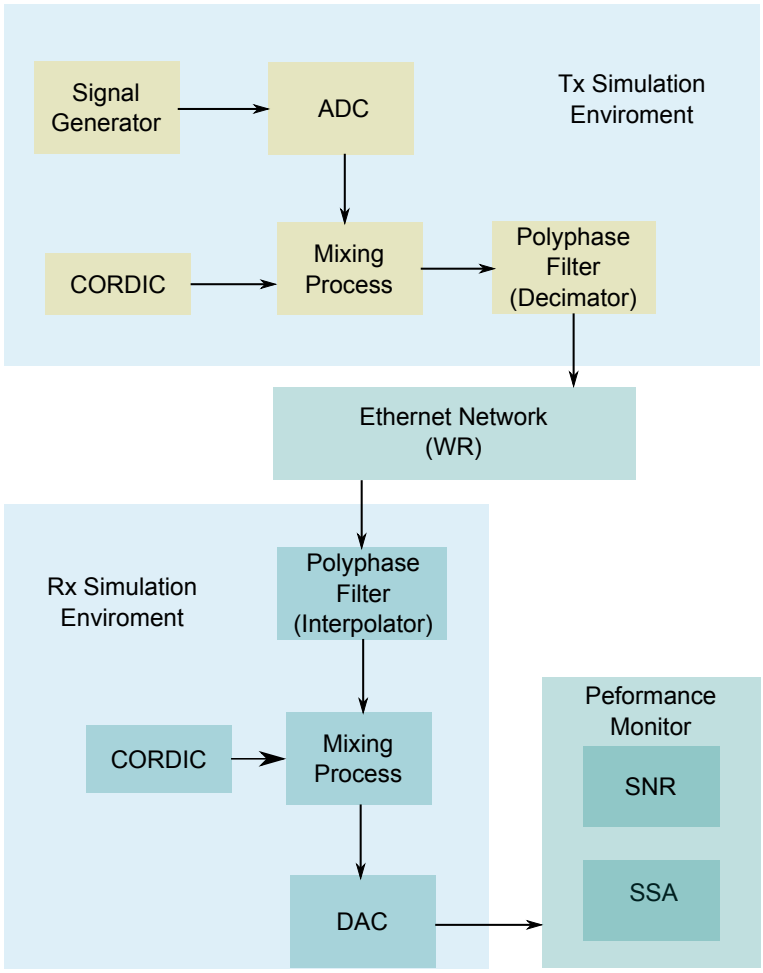


Figure 7.33: Simulation Environment Block Diagram

the stored inverse arctangent values or the quantisation of the micro rotations may be modified according to the scenario under evaluation.

- **Digital Mixer** - The mixing process is simulated using various system implementations, such as floating-point or fixed-point arithmetic.

- **Filter** - Various filter types are simulated, for instance IIR or FIR filters. The importance of this module relies in the advantage of being able to analyse the performance of the system under different filter types and especially the effect of different fixed-point implementations in optimised hardware structures.

- **Network Latency** - Packet-based networks suffer from variations in the delivery of the packets. The characteristics of this module may be customised in

the simulation environment.

- **Interpolation Filter** - In the interpolation module, several interpolation algorithms can be tested to evaluate their performance in the distributed RF system.
- **SNR Scope** - This module evaluates the performance of the distributed RF system for different configurations. The output of this module is the SNR between the input and the output signal.
- **SSA Scope** -This module, Signal source analyser (SSA), calculates the phase-noise spectrum of the reconstructed signal. This is important to evaluate jitter added in the various configurations.

7.6.1 Simulation Results

In this section, the simulated results of various system configurations are shown to illustrate the performance of the proposed RF distributed system. Based on the results, the hardware requirements for the implementation of the proposed system are obtained.

Simulations are carried out to analyse the effect of varying system parameters such as:

- ADC and DAC resolution
- Sampling clock jitter
- CORDIC accuracy
- Filter characteristics
  - Order
  - Fixed-point arithmetic

The most common distributed sinusoidal reference signals used by synchronisation systems are centred at 5, 10 and 100 MHz [144]. Moreover, this work aims to analyse the performance in distribution sinusoidal references centred at 40.078 MHz and 400.78 MHz [6]. These last two references are generated by the LHC RF cavities for equipment synchronisation, as previously mentioned in Chapter 2.

The description of the simulations results start with the analysis of how the limitation in the converters' resolution and the sampling clock jitter increase the noise present in the distributed and reconstructed RF signal on the receiver.

In addition, the noise contribution in the performance of the proposed system architecture of various system components' parameters components, such as CORDIC number of iterations, the mixing process, the decimation and interpolation, the filter type, order and arithmetic implementation, is analysed.

In the first simulation presented the RMS jitter of the sampling clock is set to 0.1 ps at 62.5MHz, the converters' sampling frequency. The simulated results are shown in Figure 7.34.

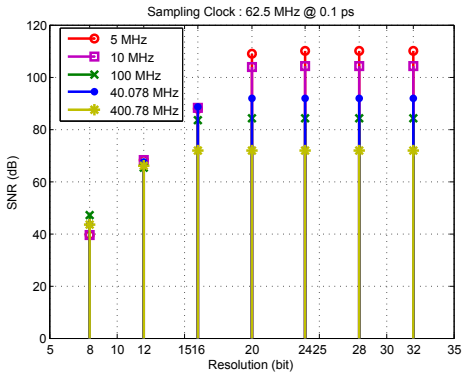


Figure 7.34: Converter's Resolution vs SNR

As shown in Figure 7.34, the SNR of the system increases with the number of effective bits used by the ADC and DAC. However, for bit resolution higher than 16-bit an increase in the bit resolution used by the converters is negligible to improve the SNR of the system. This results in reduced hardware resources during the implementation of the system in the FPGA. For higher bit resolution the majority of noise contribution

comes from the contribution of others sources, such as jitter in the sampling clock, as is shown by the next set of results.

However, the RMS jitter in the sampling clock used in the simulation to obtain the the results in Figure 7.34 is quite low for clock oscillator.

Additionally, Figure 7.35 illustrates the SNR of the system for different levels of RMS jitter in the sampling clock.

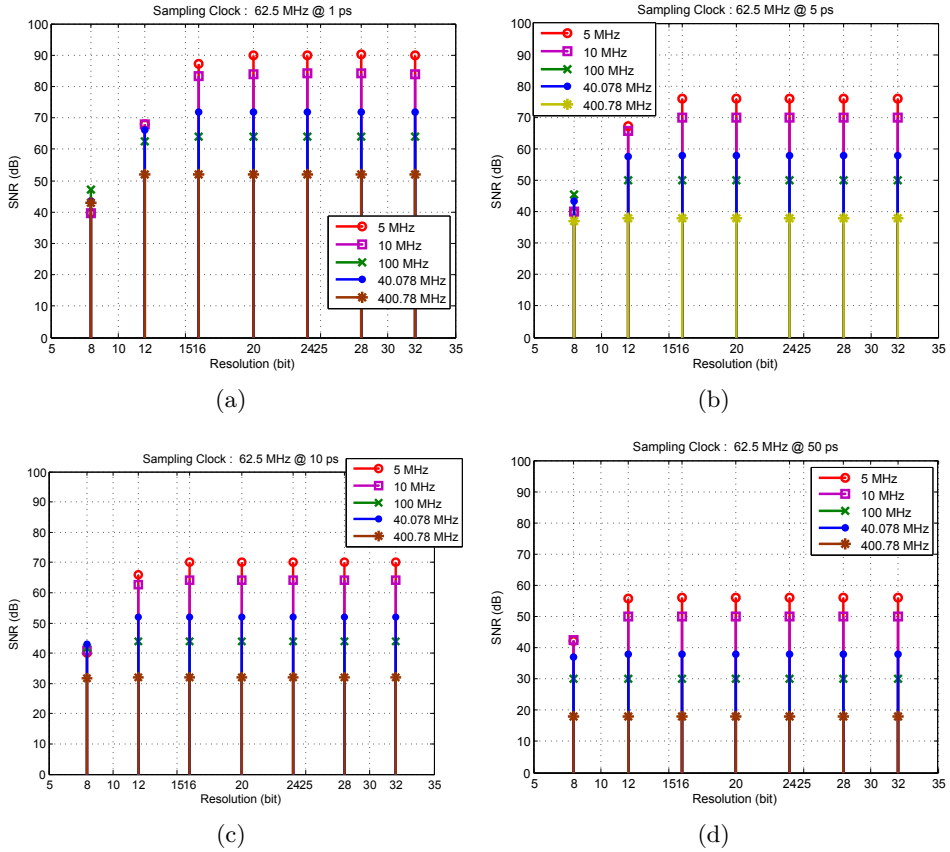


Figure 7.35: Resolution of the Converters vs Timing Jitter

Figure 7.35 shows that in case of using a sampling clock with relative high amount of jitter, an increase in the bit resolution applied in the conversion process does not improve the overall SNR of the system. Instead, it is desirable to have a sufficient number of bits in the conversion process so that the noise in the system is just due to the jitter in sampling clock. For distribution RF signals applications, such as [145], a SNR of 40 dB was a sufficient for the transportation of a RF modulated signal

digitally over the distribution system. This benchmark is also used as a minimal performance requirement for the proposed system.

Next, the influence of the number of iterations used by the CORDIC algorithm on the SNR of the reconstructed RF signal is analysed. As stated earlier, the number of iterations of the CORDIC's algorithm reflects the accuracy of the frequency reference synthesised. The sampling clock RMS jitter for the remaining simulations is set to 5 ps, within the range of a common clock reference.

Figure 7.36 shows that for a number of iterations higher than 10 the influence of the CORDIC algorithm is diminished when compared with the remaining sources of noise.

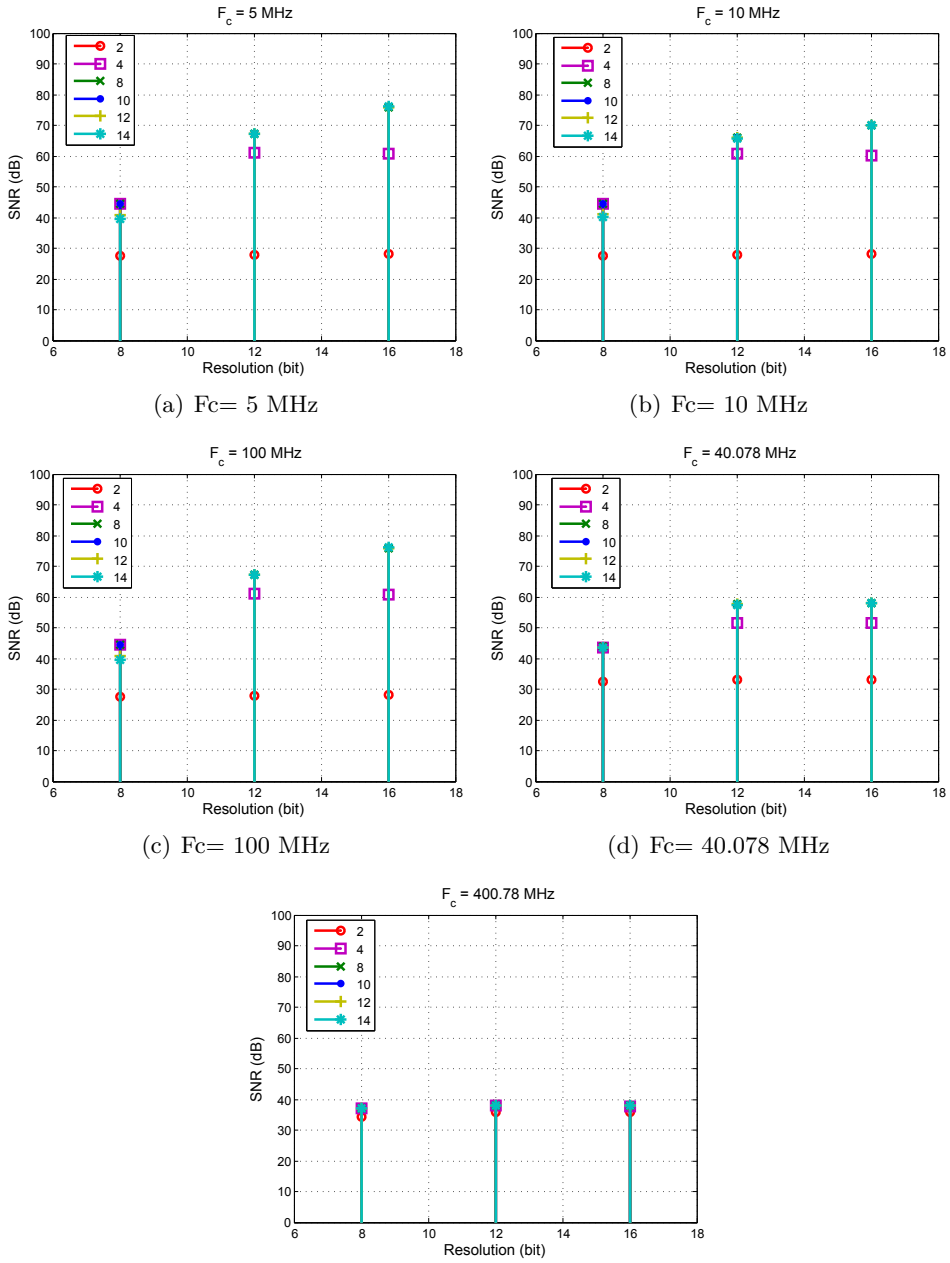
The work continues with the analysis of the digital filter types and filter order in the performance of the system.

There is a trade-off when selecting the filter specification. First, it is required to design a filter that attenuates the out-of-band tones generated by the mixing process and the additional noise. Second, the order should be set to a minimum to reduce the hardware requirements to implement the filter process in an FPGA system. Additionally, phase distortion of the signal should be minimised, an inherit characteristic of FIR filters.

The filter coefficients have been designed using the Parks-McClellan iterative algorithm [146] for optimal FIR filter order estimation. The low-pass filter calculated by the Parks-McClellan algorithm has the following specification:

- Passband Frequency f_p - 2 MHz
- Stopband Frequency f_c- 9 MHz
- Passband Stopband ripple ratio δ_1/δ_2- 0.002

Figure 7.37 shows that implementing a 60^{th} order low-pass filter results in a more than sufficient, that is higher than 40dB, system performance due to the correct



(a) Fc= 5 MHz

(b) Fc= 10 MHz

(c) Fc= 100 MHz

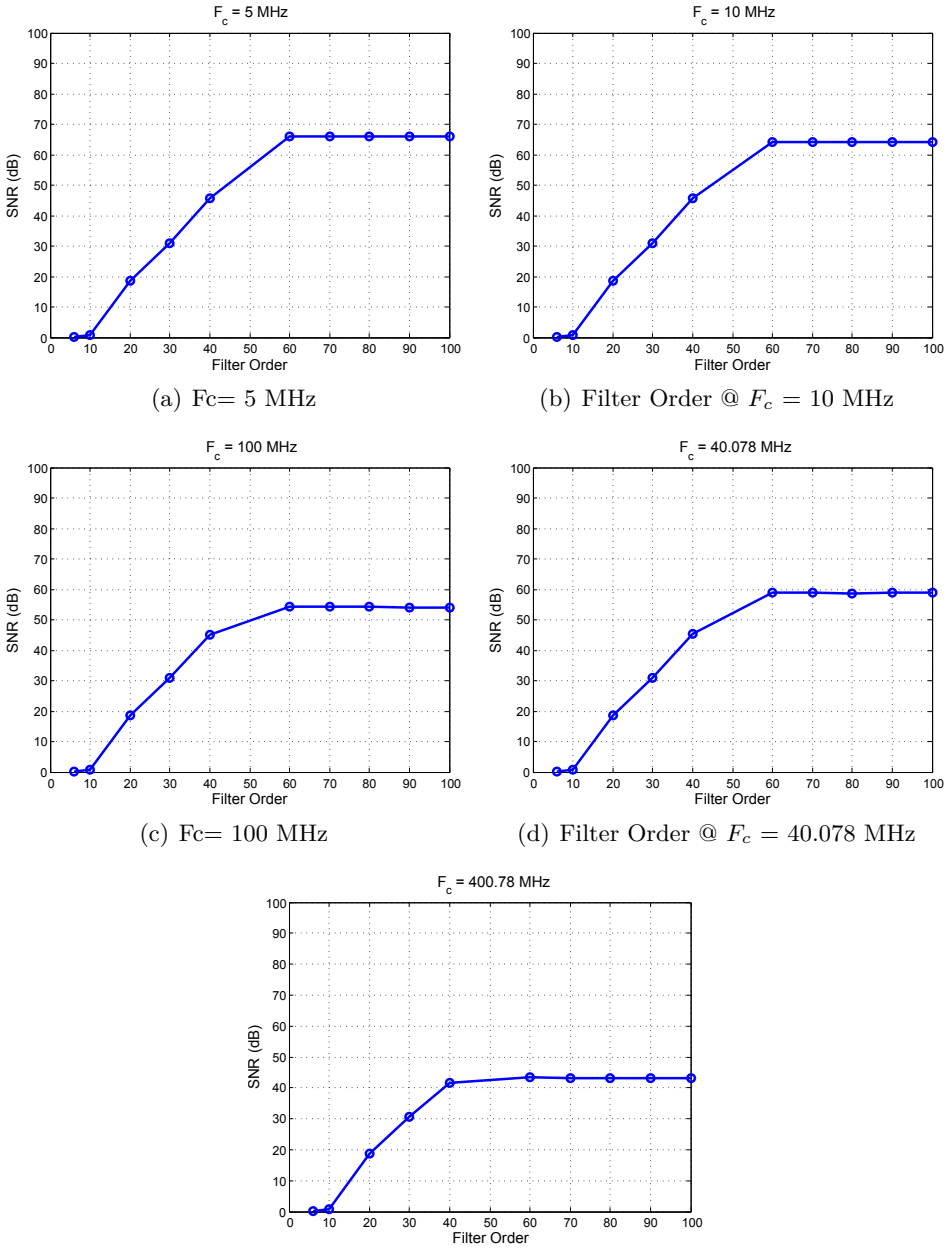
(d) Fc= 40.078 MHz

(e) Fc= 400.78 MHz

Figure 7.36: Iterations of the CORDIC vs SNR of the system

filtering of the high frequency components generated by the digital demodulation process.

Following the filtering process, a decimation process is implemented to reduce effectively the sampling rate of the signal and thus minimizing the necessary network bandwidth required to transmit the sampled RF signal to the receivers.



(a) Fc= 5 MHz

(b) Filter Order @ F_c = 10 MHz

(c) Fc= 100 MHz

(d) Filter Order @ F_c = 40.078 MHz

(e) Fc= 400.78 MHz

Figure 7.37: Filter Order vs SNR

In the receiver node, the received decimated data stream is interpolated back to the system's clock rate.

The decimation and interpolation processes also influence the performance of the system. It is essential to understand the upper bound on the decimation process following the filtering operation to prevent loss of information.

Subsequently, simulations illustrate how the decimation and interpolation processes degrades the SNR of the system.

Although the decimation is a simple process that removes a specific amount of samples from the signal to decrease the sample rate, the interpolation process requires some computation to reconstruct the data samples.

The interpolation process is implemented by a Lagrange polynomial interpolation [147] process that up-converts the decimated stream to the system's sampling rate.

Figure 7.38 shows how the decimation and interpolation filter order modifies the performance of the system in terms of SNR.

Results illustrated in Figure 7.38, show that for a decimation and interpolation factor of 10 the system reaches SNRs values above 40 dB. For decimation and interpolation factors lower than 10 the performance in terms of SNR ratio is undesirable.

The SNR is a valuable metric in determining the performance of the system distributing and reconstructing the reference signal in the Rx node. Nonetheless, phase-noise spectrum is also a valuable measure as it analysis the presence of inter-modulation distortion or harmonic distortion spurs close to the carrier that increases the noise in the signal.

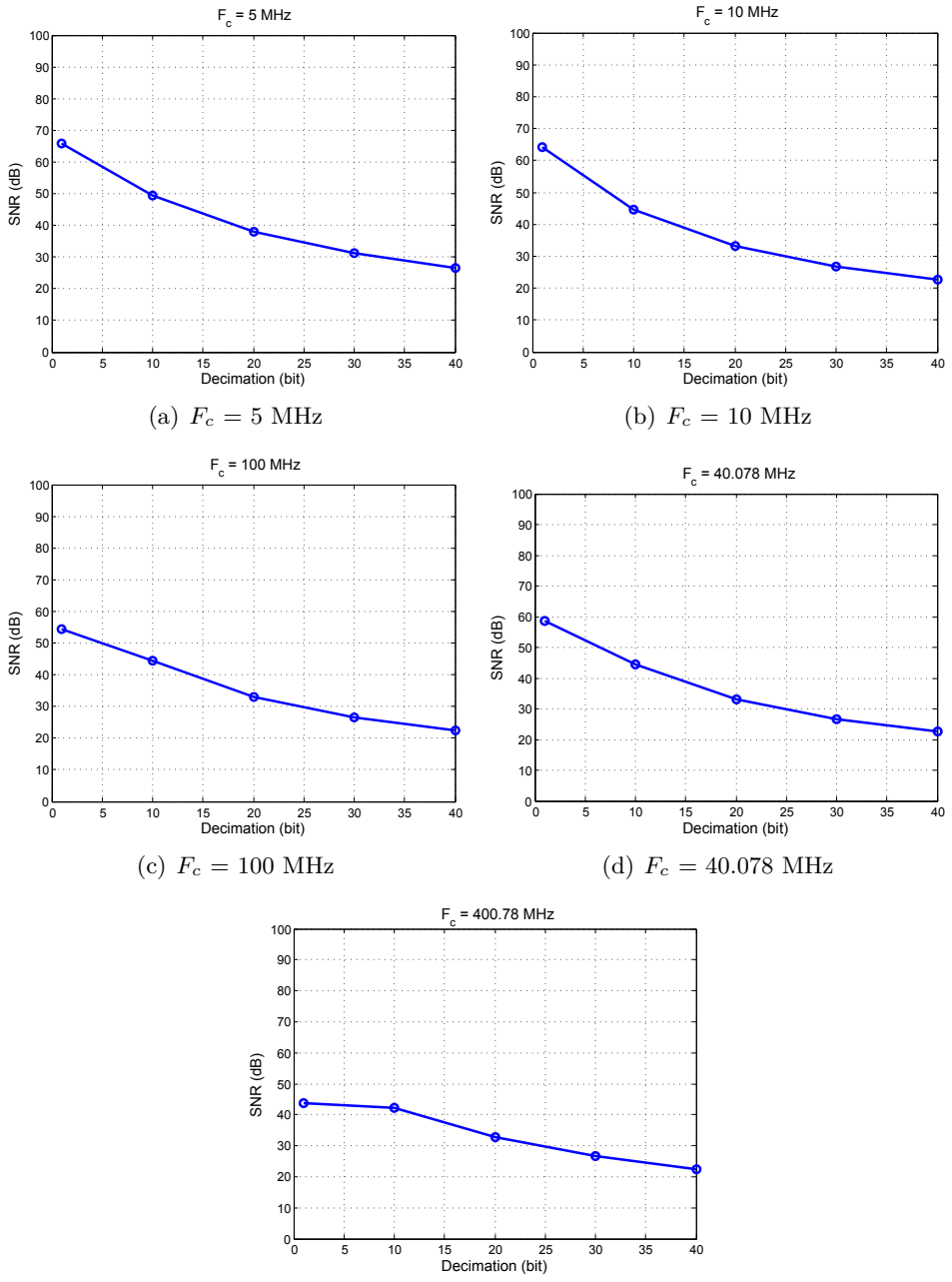
From Figure 7.39 to Figure 7.43, various phase-noise spectra of the sampled and reconstructed signals with different frequency references are shown.

Figure 7.39 shows the phase-noise spectra of the Tx and Rx node for the 5 MHz clock reference.

In addition to phase-noise spectrum estimation, the RMS jitter for the offset frequencies range shown in the x-axis of the phase-noise spectra noise is estimated.

The estimated RMS jitter for the 5 MHz RF signal measured in the transmitter is 4 ps while the estimated RMS jitter for the reconstructed signal is 6 ps.

Analysing the phase-noise spectra, displayed in Figure 7.39, between the sampled signal in the Tx Node and the reconstructed signal in the Rx, is clear to note that



(a) $F_c = 5$ MHz

(b) $F_c = 10$ MHz

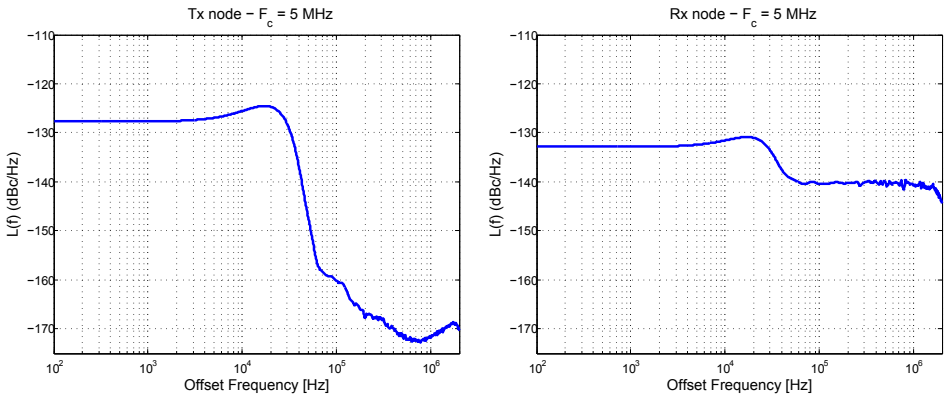
(c) $F_c = 100$ MHz

(d) $F_c = 40.078$ MHz

(e) $F_c = 400.78$ MHz

Figure 7.38: Decimation Factor vs SNR

the contribution to the increase of RMS jitter in the reconstructed signal is due to the increase in the phase-noise floor at higher offset frequencies, from 40 kHz upwards. Also interesting to observe is that phase-noise floor at lower offset frequencies, from 100 Hz to 10 kHz, is lower in the reconstructed signal than in the sampled signal.

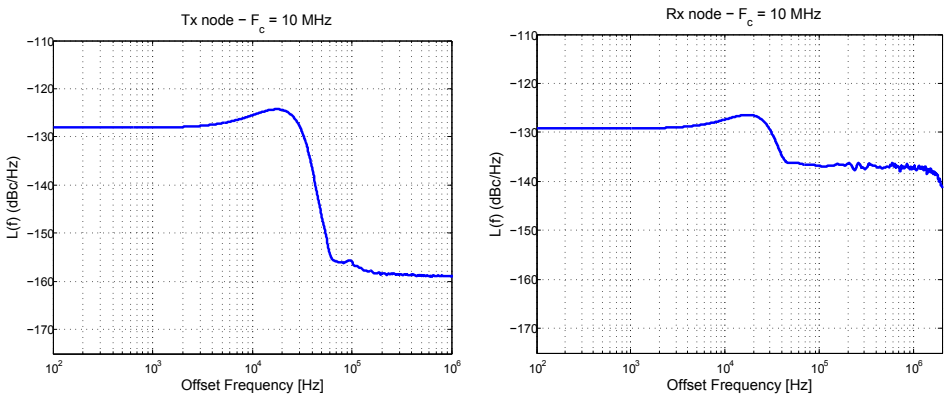


(a) Tx Node - $F_c = 5$ MHz (b) Rx Node - $F_c = 5$ MHz

Figure 7.39: Phase-Noise Spectrum of the Sampled and Reconstructed Signal - 5 MHz

The reason for this is that interpolation function plus modulator in the Rx node can reconstruct the 5 MHz signal with slight decrease of low offset frequency phase-noise, with respect than the signal sampled at the Tx node.

Figure 7.40 shows the phase-noise spectra of the Tx and Rx node for the distribution of a 10 MHz clock reference. The estimated RMS jitter for the RF signal measured in the transmitter is 2 ps, the estimated RMS jitter for the reconstructed signal is 4.5 ps.

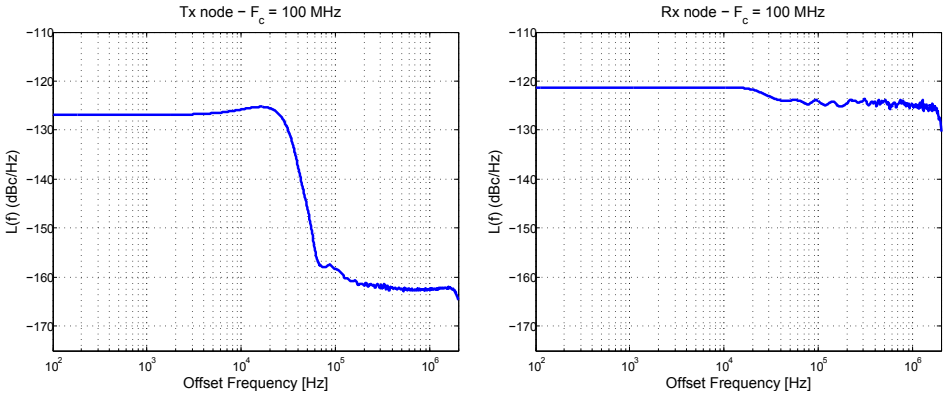


(a) Tx Node - $F_c = 10$ MHz (b) Rx Node - $F_c = 10$ MHz

Figure 7.40: Phase-Noise Spectrum of the Sampled and Reconstructed Signal - 10 MHz

Figure 7.41 shows the phase-noise spectra of the Tx and Rx node for the distribution of a 100 MHz clock reference. The estimated RMS jitter for the RF signal measured

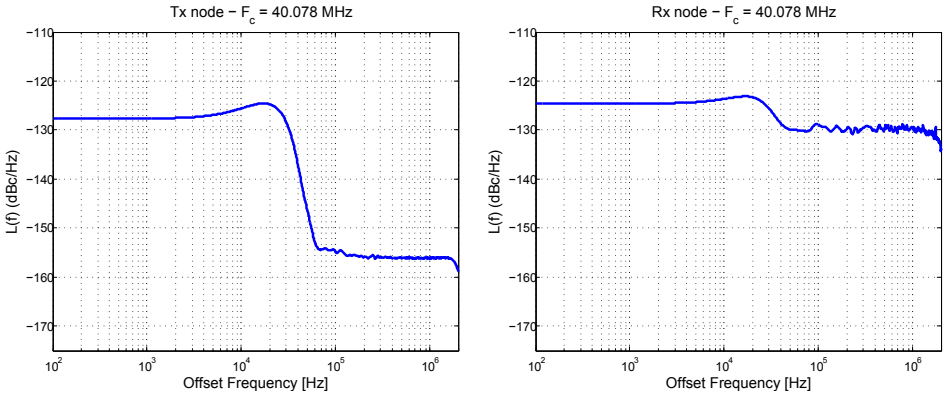
in the transmitter is 201 fs, the estimated RMS jitter for the reconstructed signal is 2 ps.



(a) Tx Node - $F_c = 100$ MHz (b) Rx Node - $F_c = 100$ MHz

Figure 7.41: Phase-Noise Spectrum of the Sampled and Reconstructed Signal - 100 MHz

Figure 7.42 shows the phase-noise spectra of the Tx and Rx node for the distribution of a 40.078 MHz clock reference. The estimated RMS jitter for the RF signal measured in the transmitter is 537 fs, the estimated RMS jitter for the reconstructed signal is 2.5 ps.

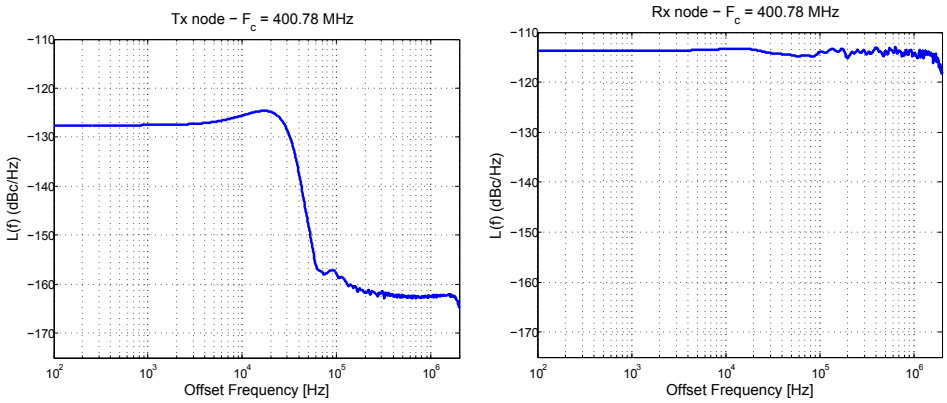


(a) Tx Node - $F_c = 40$ MHz (b) Rx Node - $F_c = 40$ MHz

Figure 7.42: Phase-Noise Spectrum of the Sampled and Reconstructed Signal - 40 MHz

Figure 7.43 shows the phase-noise spectra of the Tx and Rx node for the distribution of a 400.78 MHz clock reference. The estimated RMS jitter for the 400.78 MHz clock reference signal measured in the transmitter is 51 fs, the estimated RMS jitter for

the reconstructed signal is 1.5 ps.



(a) Tx Node - $F_c = 400$ MHz (b) Rx Node - $F_c = 400$ MHz

Figure 7.43: Phase-Noise Spectrum of the Sampled and Reconstructed Signal - 400 MHz

It is worthwhile to notice from Figure 7.39 to Figure 7.43 that the proposed system is capable of distributing RF signal with very low RMS jitter without transferring a significant amount of RMS jitter to the reconstructed signal. However, the transference of jitter is more noticeable for bandpass sampling signals, with an increase of phase noise through the measured range of offset frequencies.

7.7 Summary

In this chapter, the distributed RF over a non-dedicated timing network, the White Rabbit network, was described.

This system architecture is based on off-the-shelf components such as ADCs and DACs to digitise and reconstruct the RF signal. In addition, the proposed architecture relies on Ethernet's physical layer, a design characteristic that grant the implementation of the proposed distributing system in existing network infrastructures. This increases the distribution system competitiveness when compared with other solutions, which require dedicated transport networks.

The chapter starts with a description of the sampling theorem, followed with the bandpass sampling technique to down-convert high frequency signals to an IF signal

without the use of analogue mixers. Next, different hardware components and digital signal processing structures that make-up the proposed distributed system are analysed. A theoretical analysis of different sources of noise, which results from the conversion process, such as quantisation noise or aperture jitter are described.

The chapter continues with a description of modulation and demodulation algorithms that are realisable efficiently in hardware.

The modulation and demodulation algorithms reconstruct and split, respectively, the in-phase(I) and quadrature components(Q) of the digital RF signal. The IQ operations require an accurate numerical oscillator implemented in hardware with the minimum use of resources while having high frequency agility, to increase the input signal frequency range over which the system can operate. The CORDIC algorithm was presented as a solution to the numerical oscillator requirement. Some of the typical problems when using the CORDIC algorithm, such as long latency due to its recursive nature, are mitigated in hardware applications by using pipelined techniques. The hardware implementation of the CORDIC algorithm is further discussed in the next chapter.

The demodulation process of the input RF signal to baseband adds high frequency tones in the IQ components. These high frequency tones are removed before the components digital streams are transmitted to the received nodes. This filtering operation is a resource-hungry process that can be implemented using special architectures that optimise the required hardware resources, as described in section 7.5.

In Section 7.6, the distributed RF over White Rabbit simulation environment was presented. The simulation environment, built in MATLAB, is a model of the system that analyses the error contributions of the different components and signal processing blocks. The performance of the distributed RF system was investigated for various ADC resolutions, different RF signal carrier frequencies and signal bandwidth. In addition, the simulation environment also provided the digital signal processing requirements in terms of the accuracy of the CORDIC algorithm or the decimation

and interpolation filter requirements in terms of order and fixed-point specifications to achieve a defined SNR or phase-noise performance. The simulation results show the performance of the system under specific configurations.

In the next chapter, the hardware implementation of the proposed Distributed RF signal over Ethernet in an FPGA system is discussed.

Chapter 8

Implementation of the Distributed RF System

In this chapter, the Distributed RF over White Rabbit system is implemented on an FPGA. This prototype allows a better insight on the behaviour of the system in a real-world scenario. This is important as it is understood and accepted that computer simulations cannot accurately model hardware non-idealities, such as clock jitter, or delay path due to temperature fluctuations. In addition, to reduce the complexity of the computer-based simulation model, no particular care has been adopted to predict the real-time performance and the amount of resources needed by the proposed system architecture. These issues call for a design and implementation process to evaluate effectively the performance of the system.

To this end, Chapter 8 describes the design and implementation of the system architecture proposed and simulated in Chapter 7.

In Section 8.1, the hardware modules used in the implementation process are presented. Section 8.2 covers the FPGA design of the digital processing blocks composing the distribution RF system. In addition, Section 8.2 reports the area and speed optimisations performed to fit the requirements of the system in the FPGA. Section 8.3 analyses the performance of the system implementation by evaluating the phase-

noise spectrum, Allan Deviation and power spectrum density of the reconstructed RF signal.

The system implementation aims to test the proposed system architecture using direct and bandpass sampling techniques. In addition, it also aims to analyse the feasibility of using the proposed system in distributing clock references with low jitter in a very cost effective solution.

Part of the work presented in Chapter 8 has been published in:

- P. Moreira, I. Darwazeh **An FPGA implementation of the Distributed RF over White Rabbit**. IEEE International Frequency Control Symposium, June 2013.

8.1 Hardware

In order to implement the distributed RF system two SPEC boards were used. Each board features an FMC card that contains an ADC used for the sampling of the input clock and a DAC, which is used in the reconstruction of the distributed RF signals. Ethernet data is exchanged between the two boards over an optical link with 2.5 km length.

8.1.1 SPEC board

The SPEC board, depicted in Figure 8.1, is an FMC carrier that is equipped with one FMC connector and a SFP connector. The SPEC board also features a Xilinx SPARTAN-6 FPGA (XC6SLX45T), a PCIe interface of type 4-lane, an FPGA Mezzanine Card (FMC) slot with low pin count (LPC), an USB port, plus additional hardware [148].

The SPEC was designed to optimise cost and minimise cross-contamination between circuits to reduce board noise significantly. The SPEC is compatible with most of the

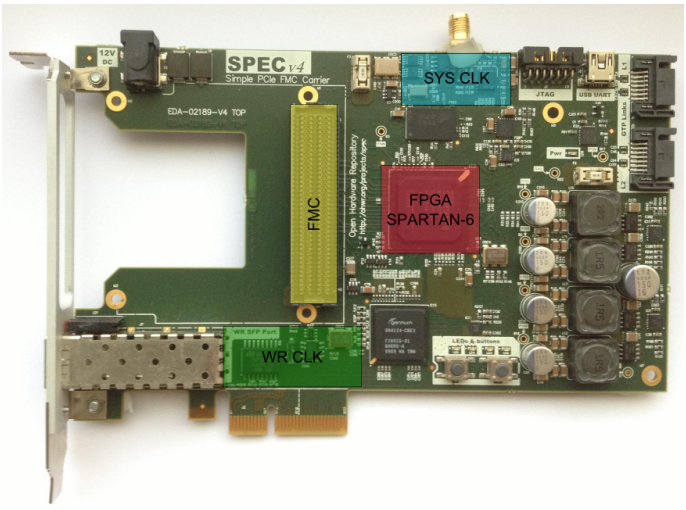


Figure 8.1: The SPEC Board

FMC cards designed within the OHR (open-hardware licensing) project (for example, ADC and Fine Delay cards) [148].

Figure 8.1 shows the top view of the SPEC board. It exposes the location of the board's functional blocks, such as the WR clocking system, the FMC, the FPGA, and the piggyback SMA connector that outputs the network clock. This output connector is used to supply the network clock reference to the FMC card.

8.1.2 FMC Mezzanine

The FMC daughter card used in this work is a commercial board from 4DSP [149], the FMC150. The FMC150, shown in Figure 8.2, provides a ADS62P49 dual channel 14-bit 250Msps ADC and a TI's DAC3283 dual channel 16-bit 800Msps DAC, which can be clocked by an external reference. The sampling clock used in the FMC150 is locked to the network clock, in other words the WR clock.

The analogue input bandwidth of the ADC is 700 MHz [150]. The clock management and fanout is performed by the TI's CDCE72010 clock synthesizer chip. The CDCE72010 device is a clock synthesizer and distribution chip with a low phase-noise clocking specification, as required for high-speed ADCs.

The FMC150 is only capable of operating the DAC and the ADC to a maximum rate of 62.5 MHz, which limits the reconstruction of high frequency RF signals. However, the reconstruction of high frequency RF signals can be done by feeding the output of the DAC to a frequency multiplier circuit, such as PLL, to up-convert the reconstructed signal to the required higher frequency.

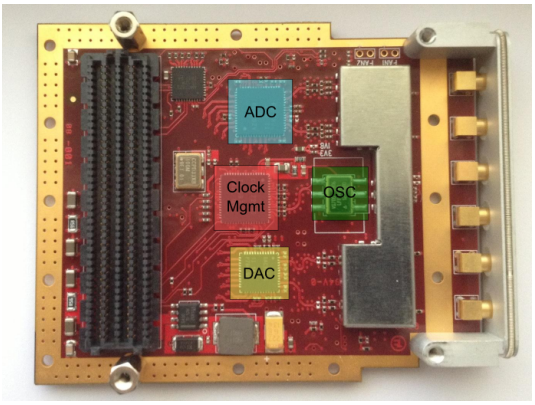


Figure 8.2: The FMC Mezzanine

The ADS62P49 has an DNL within [-0.95,+1.3] LSB and the INL is reported by the data-sheet to be around [-5, +5] LSBs. The DAC3283 has an DNL of ± 2 LSB and the INL is ± 4 LSBs. These DNL and INL quantities are suitable for the application proposed, although would be preferred to have a smaller specification, as discussed in Chapter 7.

8.2 FPGA Implementation

This section presents the Register Transfer Language (RTL) design of the distributed RF system for FPGA implementation. The digital blocks are written for FPGA implementation in VHDL. This section starts with the description of the design of the CORDIC algorithm, followed by the design strategy conducted to mitigate algorithm's inherit latency. Next, the filtering blocks, namely their polyphase structure design, is covered. Finally, the Tx/Rx streamer blocks, which implement the packaging and transmission of the DRF data to the network and the receiving and unpacking

of the DRF data from the link, respectively, are presented. The final RTL design converges to a form that meets the system requirements of functionality, area and timing in the FPGA device.

8.2.1 CORDIC

As mentioned in Chapter 7, the implementation of the CORDIC algorithm on FPGA only requires simple digital blocks such as adders, shifters and memory registers; this means that this algorithm is particularly suitable to be implemented in silicon platforms such as ASICs or FPGAs. Unfortunately, the iterative nature of the CORDIC algorithm results in high latency to converge to the desired amplitude value, especially when high accuracy is needed. However, the Pipelined CORDIC [84] architecture has the advantage of rising the accuracy of the CORDIC without further increase the latency of the algorithm, thus improving its throughput. Low latency is a requirement for the numerical oscillator used by the digital demodulator block implemented in the distributed RF system. This is because every sample arriving at a rate of 62.5 MSPS needs to be mixed by the appropriate amplitude value provided by the CORDIC.

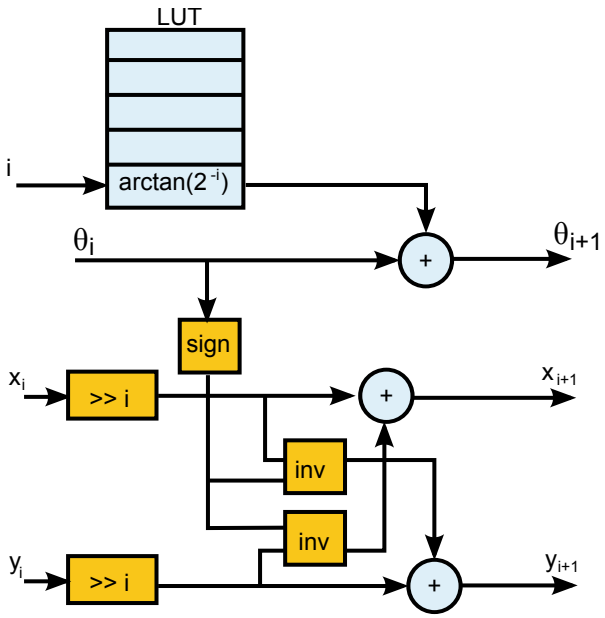


Figure 8.3: CORDIC RTL Single Stage Implementation

8.2 FPGA Implementation

A single stage of the CORDIC algorithm is composed by adders, fixed shift registers and memory blocks, as shown in Figure 8.3. The memory block is a look-up table (LUT) to store a single value of the inverse tangent function. This results in a highly optimised block for FPGA implementation.

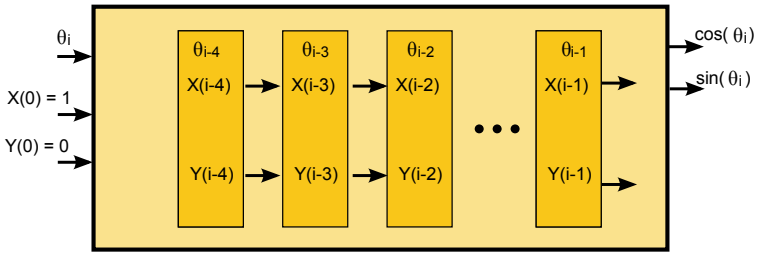


Figure 8.4: CORDIC Pipeline Implementation

Since the CORDIC iterations are similar, it is convenient to map them into a pipelined scheme, as shown in Figure 8.4. The main focus of the fast pipelined CORDIC implementation lies in the optimisation of the critical path of each stage. For the CORDIC, the critical-path is the amount of time required by the arithmetic operations. In other words, the critical-path of the pipelined CORDIC architecture relies mainly on the operation speed of the adder.

Functional verification of the design was done to simulate the behaviour of the CORDIC algorithm implemented in the FPGA. The verification was implemented in a mixed language scheme between Verilog [151] and VHDL [152] and simulated in ModelSim ®. The output values from the CORDIC RTL was then compared against a data stream generated by a MATLAB script, which models the behaviour of the CORDIC Algorithm. This verification procedure validates the functional implementation of the CORDIC algorithm in RTL.

The CORDIC implementation was realised with 17 pipelined iterations (or pipelined stages) that outputs a frequency reference with 16-bit resolution. It requires 13% of the available slices LUTS and 27% of the LUTs, in the targeted FPGA, as shown in Table 8.1. In addition, the CORDIC implementation is able to run at 120.366 MHz in the SPARTAN-6 FPGA.

Table 8.1: CORDIC - FPGA Resources

Logic Utilization	Used	Available	Utilization
Number of Slice Registers	1308	54576	2%
Number of Slice LUTs	3686	27288	13%
Number of fully used LUT-	1080	3914	27%
Number of bonded IOBs	66	296	22%
Number of BUFG/BUFGCTRLs	1	16	6%

In order to analyse the stability of the frequency reference generated by the CORDIC, the output of the CORDIC is transferred, via a serial interface, to the DAC and the phase-noise spectrum of the synthesised frequency reference is measured. Figure 8.5 and Figure 8.6 show the phase-noise spectrum of the CORDIC algorithm for a 10 MHz frequency signal in the transmitter and receiver node, respectively.

In addition, to analyse the influence of the system clock on the stability of the reference synthesised by the CORDIC algorithm, measurements were made with the system clock locked to a Caesium clock and with the system clock in a free-running state. The obtained results for both cases are shown in Figure 8.5 and Figure 8.6 for the transmitter and receiver node, respectively.

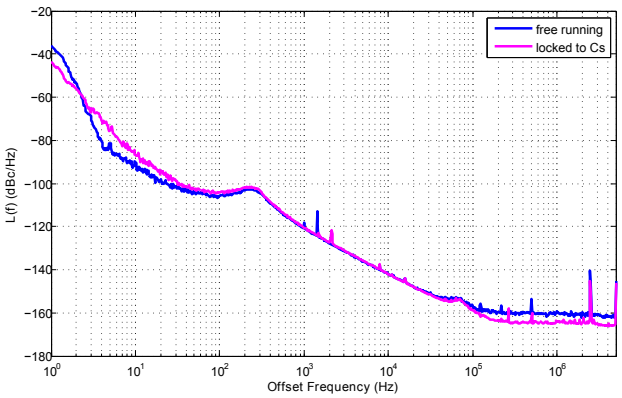


Figure 8.5: Tx CORDIC Phase Noise

As observed in Figure 8.5 and Figure 8.6, the CORDIC generates a stable frequency reference in Tx and Rx nodes. When the system clock is locked to a Caesium clock the output reference signal is more stable. However, the improvement in the phase-noise

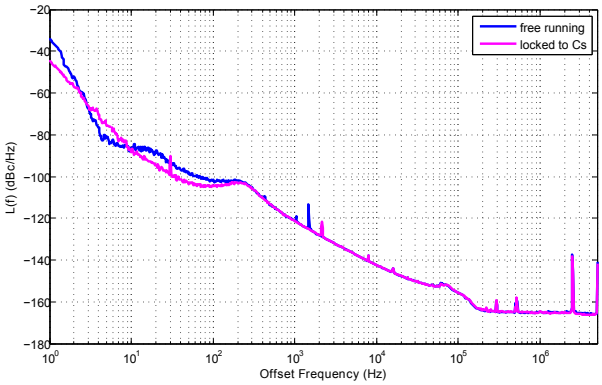


Figure 8.6: Rx CORDIC Phase Noise

is not significant and is only observed at offset frequencies below 200 Hz.

8.2.2 Polyphase Filter

Filtering and decimation/interpolation functions are merged and implemented using polyphase structures, as presented in Chapter 7. Polyphase structure allows the implementation of high order filters in a hardware sharing methodology by performing simultaneous filtering and decimation/interpolation tasks. The resulting implementation for decimation and interpolation operations is efficient both in terms of area and speed. The polyphase filter implemented in the distributed RF system consists in a set of 6^{th} order FIR filter banks, whose coefficients and delays are being multiplexed according the digital sample index.

As defined in Section 7.6.1, the filtered data stream can be decimated by a factor of 10, that is only the 10^{th} sample computed by the filter is packeted and sent to the distributed nodes. The other remaining samples are simply discarded. The combination between the filter bank order and the decimation factor results in an optimised 60^{th} order polyphase filter. The filter coefficients were calculated using the Parks-McClellan iterative algorithm [146] for optimal FIR filter order estimation. The low-pass filter calculated by the Parks-McClellan algorithm has the following specification:

- Filter Order N - 60
- Passband Frequency f_p - 2 MHz
- Stopband Frequency f_c- 9 MHz
- Passband Stopband ripple ratio δ_1/δ_2- 0.002

The frequency response of the 60^{th} order low-pass filter designed is illustrated in Figure 8.7. The frequency response was calculated with MATLAB, thus the attenuation of -160 dB in the Stopband.

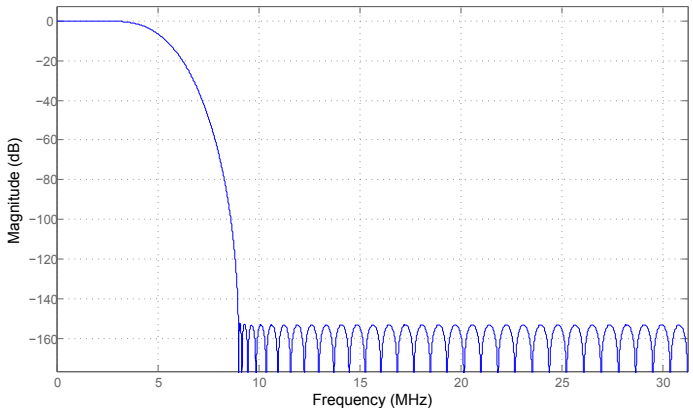


Figure 8.7: Polyphase Decimation Filter - Frequency Response

In Figure 8.8, the step response of the polyphase filter implemented in the FPGA using fixed-point arithmetic, is shown.

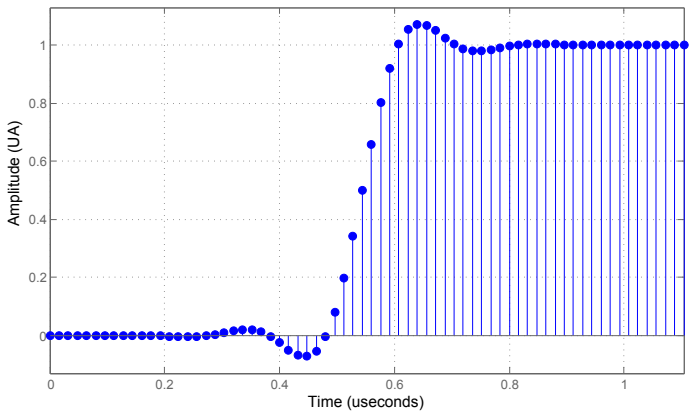


Figure 8.8: Polyphase Decimation Filter - Step Response

8.2 FPGA Implementation

The implementation of the FIR filter bank did not follow any design implementation strategy, because the target FPGA had the necessary resources for the filter's direct implementation. The only optimisation carried out in the implementation was in the translation between the floating-point to the fixed-point arithmetic. Just the implementation of the filter's arithmetic in floating-point depletes almost all the available resources in the targeted FPGA.

The filter coefficients were normalised to 25-bit precision and the input samples are 12-bit wide. This was verified by simulations, in chapter 7, to reach the desired SNR performance.

The conversion between floating-point to fixed-point arithmetic results in a reduction in the filter attenuation at higher frequencies when compared with the floating-point implementation. However, it is still sufficient to filter the high frequency components generated by the mixing process.

An alternative approach could use a multiplierless filter [139] with a more efficient implementation, as it can save up to 70% of resources relative the implementation strategy adopted here, however, offering less flexibility. The operation speed of the synthesised polyphase filter is within the requirements of the design, and is capable of running at maximum rate of 75 MHz in the SPARTAN-6 FPGA.

The polyphase filter requires 3% of the available Slice LUTs and 3% Slice Registers. However, the decimation filter needs 48% of the DSP481A1 [153] available in the FPGA, as shown in Table 8.2.

Table 8.2: Polyphase Filter - FPGA Resources

Logic Utilization	Used	Available	Utilization
Number of Slice Registers	2035	54576	3%
Number of Slice LUTs	969	27288	3%
Number of fully used LUT-FF pairs	206	2798	7%
Number of bonded IOBs	59	296	19%
Number of BUFG/BUFGCTRLs	1	16	6%
Number of DSP48A1s	28	58	48%

8.2 FPGA Implementation

At the receiver node, an interpolation filter is designed to up-convert rate of the decimated signal to the system clock rate. The implemented filter is a 60^{th} order FIR filter that performs Lagrange polynomial interpolation on a sequence interleaved with consecutive zeros at every sample received. The interpolation polyphase filter performs a joint process of up-sampling the decimated signal to its initial sampling frequency, as well as low-pass filtering to reconstruct the signal. The frequency response of the calculated interpolation filter is shown in Figure 8.9.

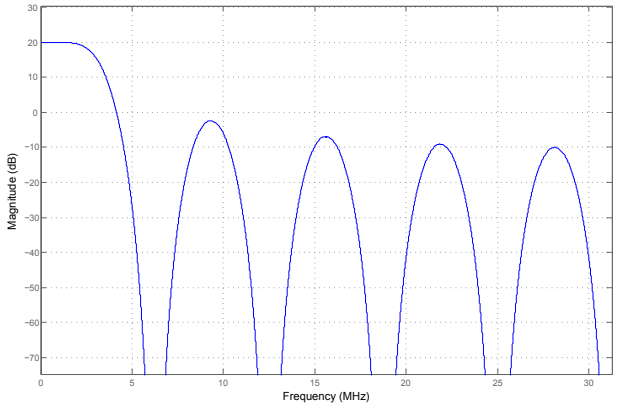


Figure 8.9: Polyphase Interpolation Filter - Frequency Response

Figure 8.8 shows the step response of the interpolation filter implemented in the FPGA using fixed-point arithmetic.

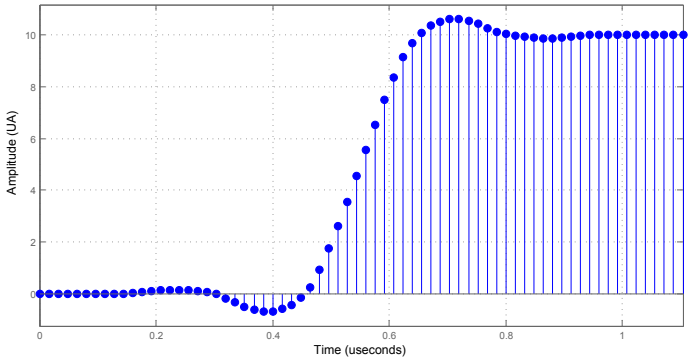


Figure 8.10: Polyphase Interpolation Filter - Step Response

The interpolation filter was designed to perform Lagrange polynomial interpolation, as defined in Chapter 7. The interpolation filter uses the same polyphase structure as implemented for the decimation process, with a 6^{th} order filter banks.

Therefore, the area and maximum execution speeds are similar to the ones required for the decimator, as shown in Table 8.2.

8.2.3 Data Streaming

Following the digital processing blocks required to demodulate the IQ values from the sampled RF signal, it is necessary to build an Ethernet frame with the processed data and transmit the frame to the receivers nodes. This section describes the blocks implemented to pack the decimated data into an Ethernet frame.

Next, is described the block implemented to demultiplex data from the Ethernet frame. These blocks are defined as the Tx Streamer and Rx Streamer modules, respectively.

Both the Tx and Rx nodes are designed to handle three types of data, these are timing data, DRF data and general-purpose data. These data types are stored in different buffers.

The data scheduler process employs a prioritisation and round-robin algorithm and handles the data queues to transmit the available data to the PCS layer.

Timing and DRF data have higher priority for transmission than the general purpose data. When there is a request to transfer PTP or DRF data-types, the data is immediately forwarded to the PCS layer.

A round robin process scheduler, illustrated as Link Arbitrator in Figure 8.11, is implemented to prevent PTP and DRF requests from fully using the Ethernet link. However, both PTP and DRF design implementation prevent the starvation of the transmission link by their traffic, because they are designed to use a lower network bandwidth.

The Rx streamer and Tx streamer are connected to the Xilinx GTA_DUAL Module [154] that performs the PCS operations, such as the encoding and decoding of the data.

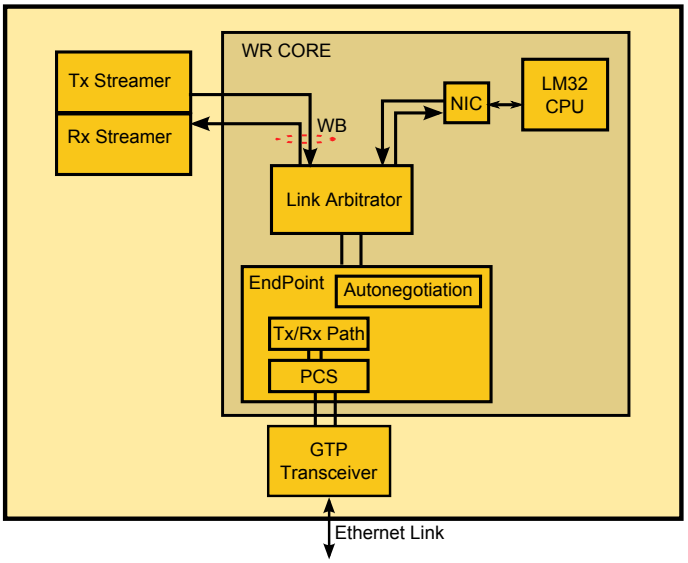


Figure 8.11: Block Diagram for Tx/Rx Path

The operation sequence when receiving Ethernet packets is described as follows: During the reception of a frame, IEEE 1588 time-stamping is always enabled and the it is performed when the Start-of-Frame symbol is detected. The time-stamp is appended to the frame received from the PCS and is transfer to the Rx FIFO before the corresponding receive symbol is written, as explained in Chapter 3.

Tx Streamer

In the Tx Streamer, a FIFO memory, the Tx FIFO, is used to buffer and align the IQ values that will be transmitted. When the FIFO reaches a pre-defined, and configurable, level the finite-state-machine implemented in the Tx Streamer starts to pipeline the stored IQ samples to construct the Ethernet frame. The Tx Streamer pipelines a complete Ethernet Frame with the Destination MAC Address, the Source MAC Address, the Type/Length fields, the payload data and the frame data CRC.

A specific Ethernet type, with the value 0xDBFF, was defined to differentiate the DRF Ethernet packet, which contains the distributed RF data, from the remaining Ethernet traffic, such as generic data or timing data.

The Ethernet packet that delivers distribute RF data the payload data structure,

which organises the distributed RF data stream, as shown in Figure 8.12:

	D15	D14	D13	D12	D11	D10	D09	D08	D07	D06	D05	D04	D03	D02	D01	D00
0 ... 1	DATA_COUNTER(16 bit)															
2	I[1] (8 bit)								Q[1] (8 bit)							
...																
n	I[n] (8 bit)								Q[n] (8 bit)							
n+1	CRC(16 bit)															

Figure 8.12: Distributed RF (DRF) - Ethernet Frame

The Ethernet packet after the data multiplexing is transmitted to the End Point block through the wishbone fabric module, as illustrated by 8.11. Pulse signalling is used to limited the start and end of a frame they are the start-of-frame (SOF) descriptor and a last-of-frame (EOF) descriptor, respectively. The wishbone fabric module accommodates the SOF, EOF and Ethernet data into a wishbone bus interface to delivered the frame to the WR CORE. A block diagram of the Tx Streamer is shown in Figure 8.13

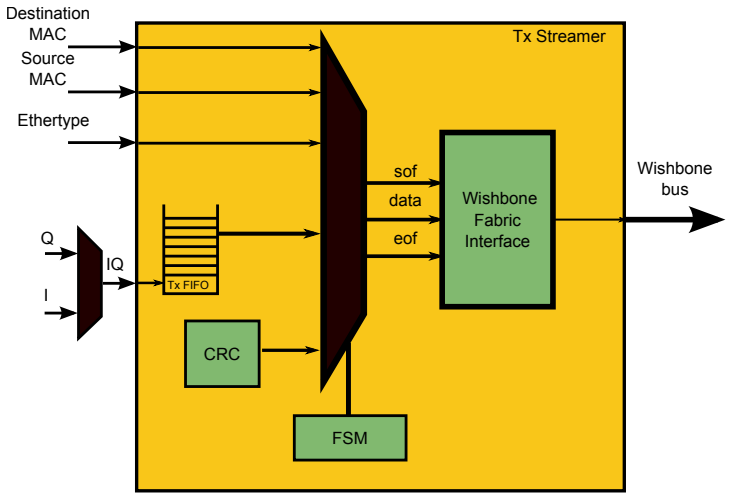


Figure 8.13: Tx Streamer Block Diagram

The Tx FIFO was designed to accommodate small delays in the data transmission to the optical link. The maximum amount of delay that the system can absorbed without loosing DRF frames depends on the number of frames that can be stored in the Tx FIFO. However, due to the tight synchronisation achieved by the nodes

in the WR network, the level of the FIFOs remains constant during the distribution operation. For every frame that leaves the FIFO another frame is being stored.

The functional behaviour of the various signals in Tx Streamer is illustrated in Figure 8.14.

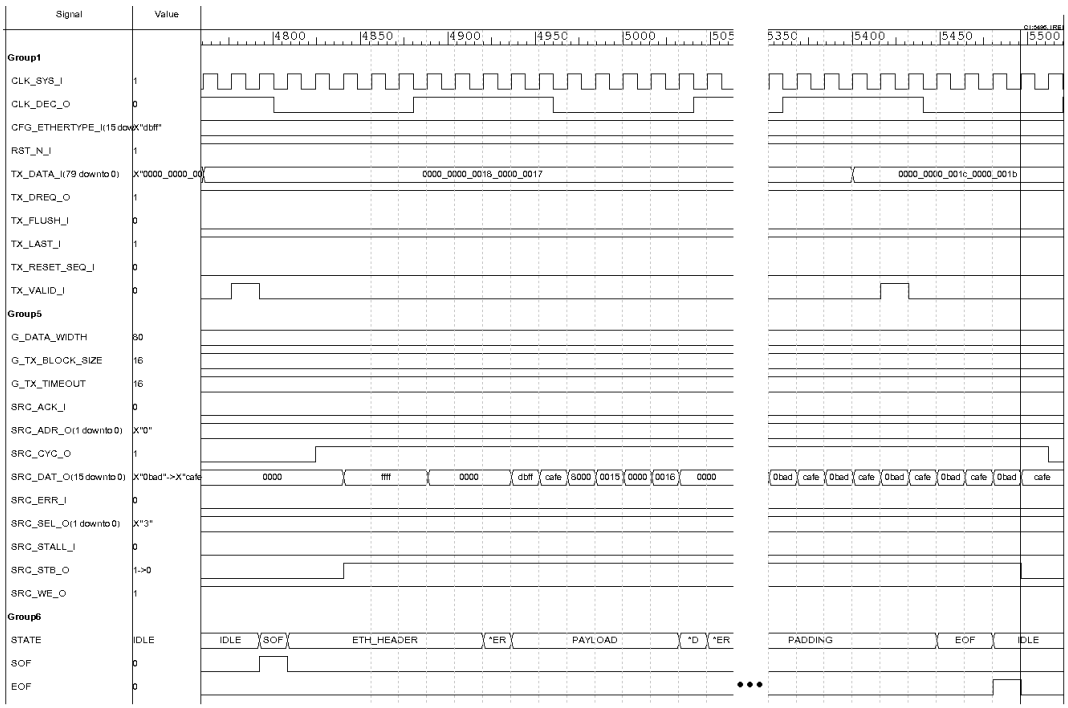


Figure 8.14: Timing Diagram - Tx frame streamer

In Figure 8.14 the behaviour of a DRF frame transfer during the transmission of a frame to the wishbone bus is verified. The sequences during the DRF frame transfer is described next.

The pulse observed in the signal “SOF” informs about the start of a frame, the Tx Streamer then changes its signal “STATE” to “ETH_HEADER” where it sends the destination and source mac address. The Tx Streamer continues with the transmission with the data type and payload data. The DRF frame transfer stops when the signal EOF outputs a pulse.

Rx Streamer

The received Ethernet frames with the designated Ethertype, 0xDBFF, are processed by the Rx Streamer to demultiplex the IQ data from the DRF frame. When the Rx Streamer receives a frame, it pushes in data along with byte enables. The Rx Streamer also indicates the SOF and EOF, as shown in Figure 8.15. The received IQ values are then realigned and stored in a FIFO, the Rx FIFO, until the modulator block requests for further samples. After the EOF is received, the streamer goes to Idle and waits for another Ethernet frame.

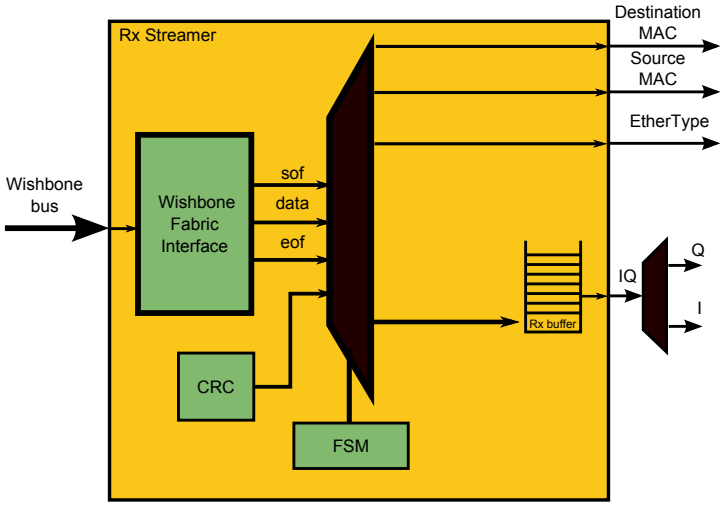


Figure 8.15: Rx Streamer Block Diagram

In Figure 8.16 the functional behaviour of the Rx Streamer is illustrated, during the reception of a DRF frame.

In Figure 8.16 is verified the behaviour of a DRF frame transfer during the reception of a frame from the wishbone bus, which is described next.

The pulse observed in the signal "SOF" signals the Rx streamer that a frame is about to start, the Rx Streamer then receives the destination and source mac address. The reception of a frame continues with the data type and payload data. The DRF frame transfer stops when the signal EOF outputs a pulse (which is not displayed in Figure 8.16, for simplicity).

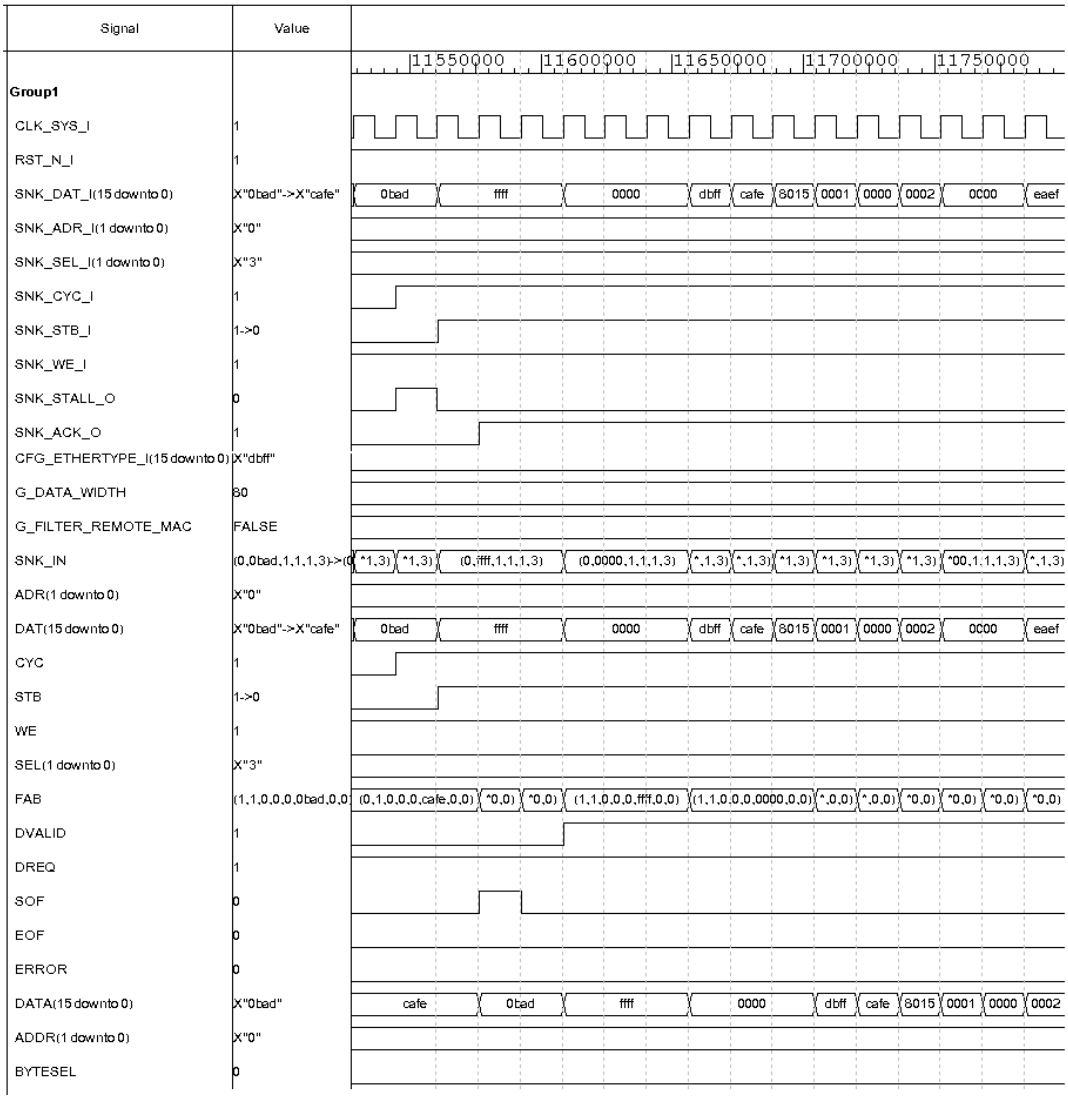


Figure 8.16: Timing Diagram - Rx frame streamer

8.2.4 Summary

The current RTL implementation for the DRF core, the Tx and Rx nodes, requires the FPGA's resources shown in Table 8.3.

The resource utilisation, shown in Table 8.3, include, in addition to the DRF core, an instantiation of the LM32 processor plus wishbone interfaces between the processor and the DRF core. Nonetheless, there is further room to optimise the required resources to implement the DRF core (for example, by reducing the fixed-point representation of the filters) at an expense of lowering the performance of the system.

The complete DRF core is able to ran at defined system frequency of 62.5 MHz, in the target FPGA.

Table 8.3: Full DRF Implementation - FPGA Resources

Device Utilization Summary (estimated values)			
Logic Utilization	**Used**	**Available**	**Utilization**
Slice Registers	9133	54576	16%
Slice LUTs	14038	27288	51%
fully used LUT-FF pairs	3922	27951	14%
Block RAM/FIFO	33	116	28%
DSP48A1s	54	58	93%
PLL_ADVs	2	4	50%

The performance of the system implementation is reported in the following section.

8.3 Experimental Results

The experiments are divided by the two sampling scheme used. Firstly, the performance of the system architecture when the RF signal is sampled using direct sampling is presented. The second set of measurements reports the performance of the distribution of high frequency signals, which use bandpass sampling. The performance of the system is evaluated through measurements of the phase-noise spectrum, RMS jitter, Allan Deviation and PSD of the distributed signal.

The hardware used in the performance validation of the proposed architecture, is illustrated in Figure 8.17. The distributed system architecture is implemented using two SPECS boards, which are connected by an 2.5 km optical fibre link (G.652). The link is connected through transceiver modules that are SFP compliant. The SFPs use a long wavelength 1310/1490 nm Fabry-Perot (FP) laser diode to transmit data and a 1310/1490 nm Positive-Intrinsic-Negative (PIN) photodiode in the receiver. This transceiver does data transmission up to 10 km stated by its specification. As mentioned earlier, the ADC samples the RF input signal with 14-bit resolution, while the signal's reconstruction is done by a 16-bit DAC. The phase-noise spectra of the clocks signal are measured with Agilent's E5052B Signal Source Analyser [70].

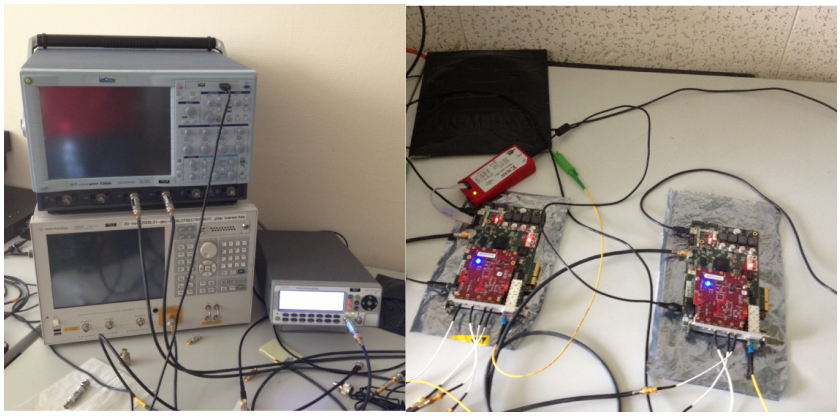


Figure 8.17: Experimental Setup

The performance of the distributed RF system over the WR network was analysed for both direct and bandpass sampling. The centre frequencies of the RF signal were chosen to be 10 MHz, 40.07 MHz, 100 MHz and 447.5 MHz. The sampling rate is 62.5 MHz (WR network clock) that results in direct sampling only for the signal centred at 10 MHz and bandpass sampling for the remaining signals.

The first measurement is the phase noise of the reference clock, which is centred at 62.5 MHz. This clock signal is also the sampling clock reference for the ADC in the Tx node.

Figure 8.18 shows two traces; one is the phase-noise spectrum of the sampling clock when the reference clock is free-running. The second is the phase-noise spectrum of the sampling clock when the system clock is locked to the WR network clock reference, which is locked to a Caesium clock in the Tx node.

The estimated RMS jitter of the free running clock, calculated between 5 Hz and 5 MHz frequency range, is ~14 ps. The sampling clock locked to the WR network clock shows an estimated RMS jitter of ~13 ps for the same frequency range. In Figure 8.18 the appearance of frequency spurs above 10 kHz is noticeable. These spurs are attributed to external electromagnetic interference coupled with inadequate shielding of the cables used for measurements.

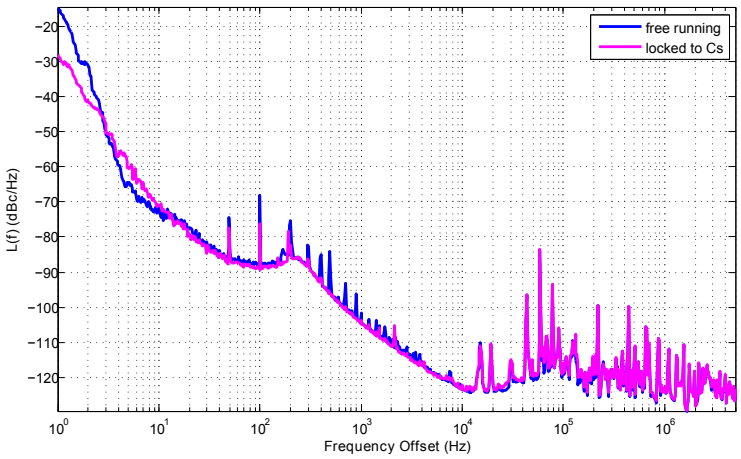


Figure 8.18: Phase-Noise Spectrum - Sampling Clock

8.3.1 Direct Sampling

An application for the distributed RF system is to distribute high stable clock references, such as atomic clock, over a local area network, using the WR network. As such, this system configuration aims to distribute a RF clock signal generated by a commercial Caesium atomic clock, the Symmetricom CS4000 [155].

The RF signal generated by the CS4000 is a highly stable 10 MHz signal, as verified by the phase-noise spectrum shown in Figure 8.19. The phase-noise measurement shown in Figure 8.19 is a print screen of the Agilent's E5052B SSA. Unfortunately, the E5052B does not show units in the axis of the graph. For reference, whenever this print-screen appear the x-axis corresponds to Offset Frequency with units Hz, while the y-axis is the phase-noise with units dBc/Hz.

The estimated RMS jitter generated by the clock reference, between 5 Hz to 5 MHz, is ~1 ps.

The 10 MHz signal is sampled at 62.5 MHz, by the sampling clock that is locked to the WR network clock. The sampling process creates a spectral image of the analogue signal in the ADC's Nyquist range. The CORDIC algorithm is configured to generate a 10 MHz mixing frequency. The mixing process multiples the CORDIC outputs with the sampled RF signal to generate the IQ components of the sampled

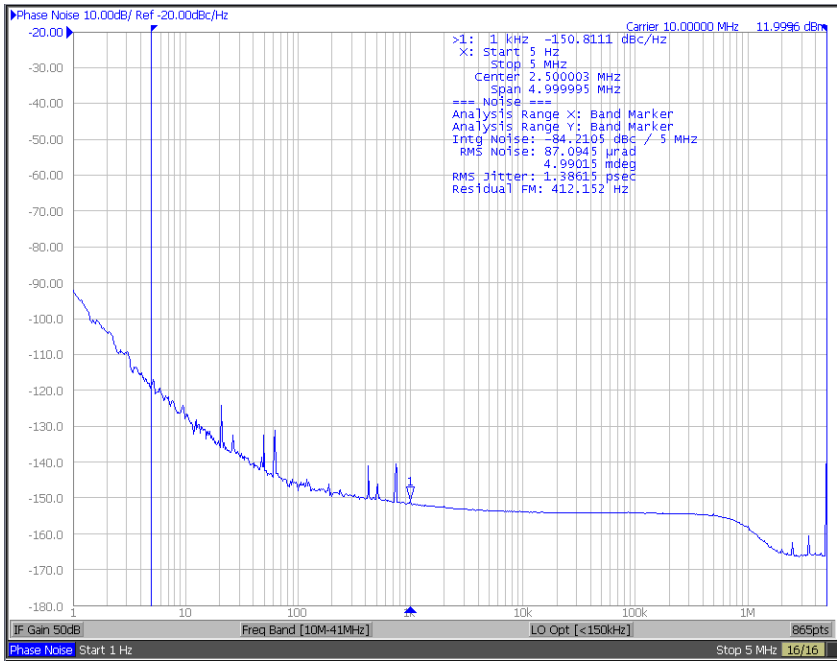


Figure 8.19: Phase-Noise Spectrum of the 10 MHz - Tx Node

signal. The outputs from mixing process are filtered to remove the high frequency components, as discussed in Chapter 7. The filtered components are decimated by 10 to reduce the required network bandwidth required to transmit the DRF stream. This decimation factor results in an utilisation of only 17.5% of the available network bandwidth to distribute the components over the network. The network utilisation is estimated by multiplying the two signals by the quantisation factor (14-bit) and the decimated sampling rate.

On the receiver side, the decimated stream composed of the IQ samples are received synchronously. The IQ modulation process is then employed to reconstruct the RF signal. The reconstructed signal in the Rx node is phase and frequency locked to the input RF signal in the Tx node. The measurements were conducted at room temperature.

Figure 8.20 shows the phase-noise spectrum of the reconstructed RF signal.

The RMS jitter of the reconstructed signal calculated from the phase-noise spectrum in Figure 8.20 is ~7.2 ps. A reduction in the RMS jitter in the reconstructed signal

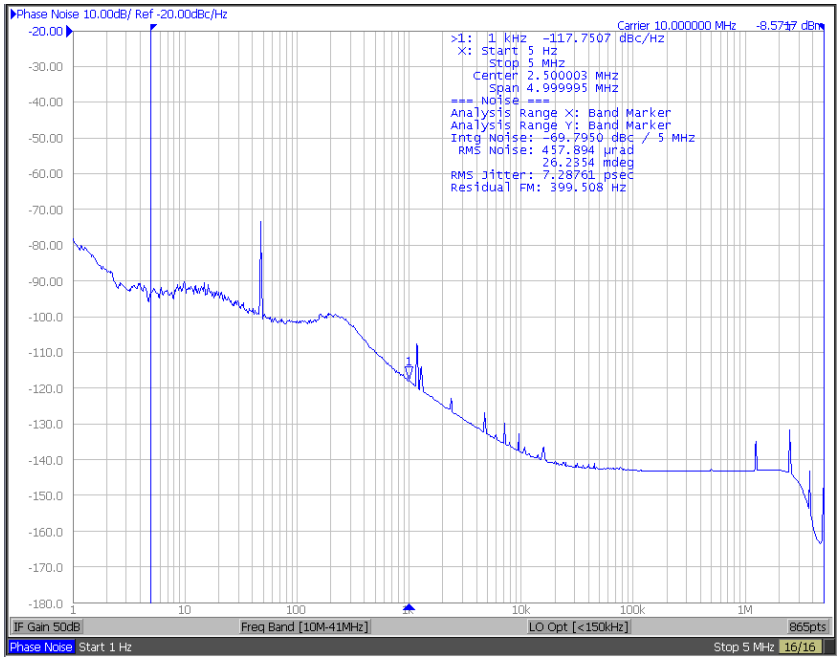


Figure 8.20: Phase-Noise Spectrum of the 10 MHz - Rx Node

can be achieved by designing a PLL circuit to filter the noisy frequency components present in the reconstructed RF signal [156].

For the best of the author's knowledge, this is the first time such a highly stable signal source is synchronously distributed using an Ethernet network. Although the results in relative terms shown a significant increase in the RMS jitter, as the RMS between the Tx node and the Rx node increases seven times, in absolute terms the obtained result is excellent as the system is able to reconstruct a signal synchronous with the Caesium reference with an RMS jitter of ~7.2 ps.

Allan Deviation

To characterise the long term stability of the reconstructed signal measurement of the time interval between the signal in the Tx node and the reconstructed signal was carried out. The time interval measurements are used to calculate the Allan Deviation of the reconstructed signal.

The Allan Deviation is a simple and effective measure to verify random drift error as

a function of averaging time. It is used to determine the characteristics of different random processes [71].

Figure 8.21 illustrates the time interval measurements between the RF signal in the Tx node and the reconstructed signal in the Rx node. The time interval measurements were realised using the Pendulum CNT-91 Timer Counter [74]. The Pendulum CNT-91 Timer Counter provides a 50 ps single shot time resolution.

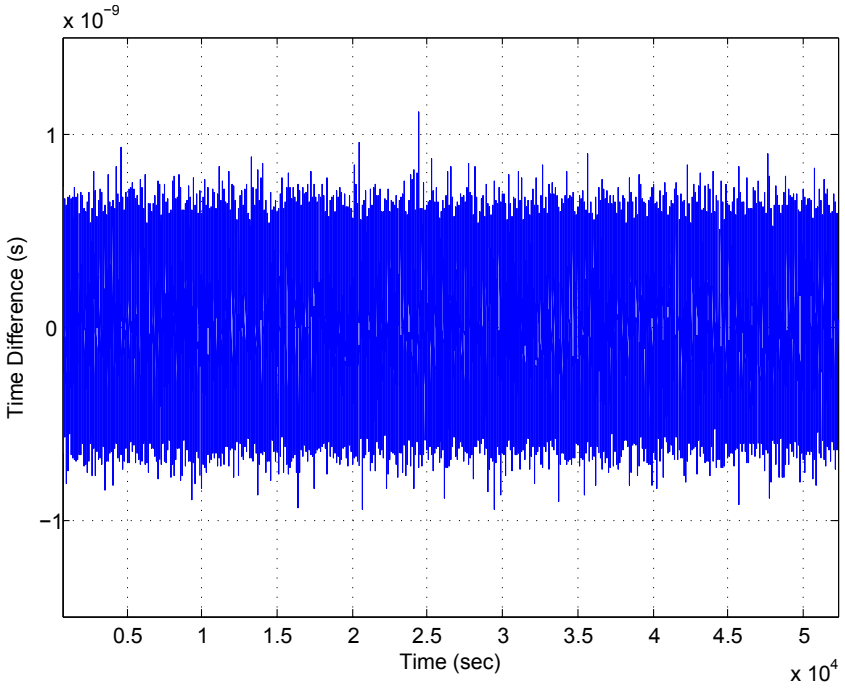


Figure 8.21: Time Interval measurements between the reference and the reconstructed signals

Figure 8.22 shows a fourteen hours uninterrupted working period, within this period the maximum time interval was 1.11 ns with a standard deviation of 208 ps.

Figure 8.22 shows the Allan Deviation calculated from the time interval measurements. For a averaging time of 1 second (τ) the Allan Deviation is ~28 ps while for a averaging time of 1800 seconds the Allan Deviation is ~62.7 fs.

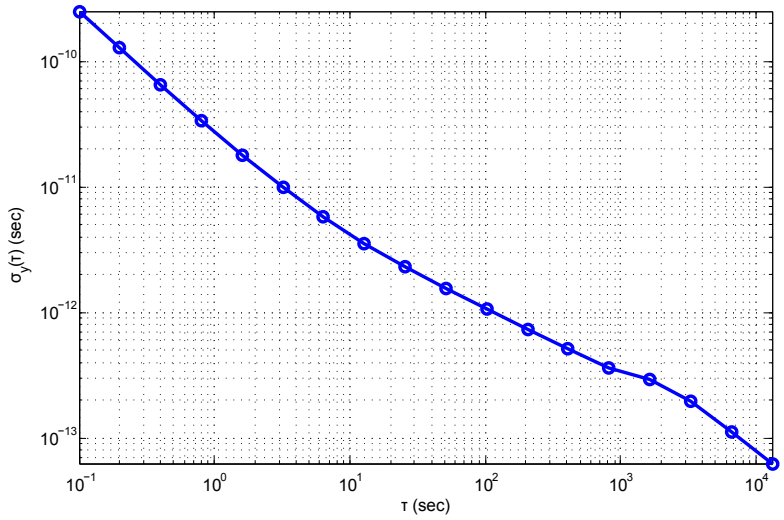


Figure 8.22: Allan Deviation of the 10 MHz Rx Signal

Power Spectrum Density

The PSD gives the distribution of the signal power among a range of frequencies. The PSD is measured to assess harmonic distortion in the signal due to the presence of frequencies spurs, inter-modulation and harmonics.

In Figure 8.23 the PSD of the 10 MHz sampled in Tx node is depicted. In addition, the PSD of the reconstructed signal in the Rx node is also shown in Figure 8.23. The SNR at the output of the DAC after the LP Filter is ~51.78 dB. It is noteworthy, to see in Figure 8.23 that the SNR measured in the Tx node is identical to that measured for the Rx node.

The proposed system architecture does not influence the SNR of the distributed signal, the degradation of the SNR is due to the domain conversion between digital to the analogue domain. The PSD of the signal is measured at the Tx node using the DAC of the Tx node. This is realised by directly feeding the sample signal obtained by the ADC to the DAC.

From Figure 8.23 harmonics at 10 ± 1.6 MHz, at 10 ± 2.5 MHz and at 10 ± 3.2 MHz are identified. These intermodulation and harmonics components are due to the sampling and reconstruction processes.

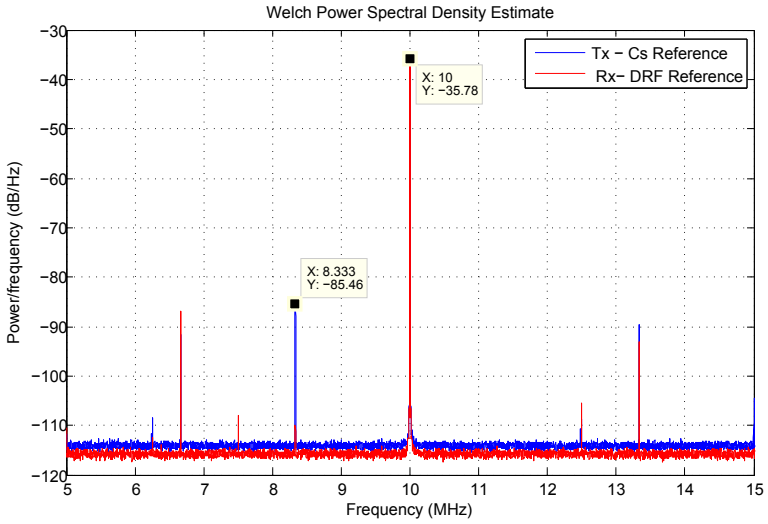


Figure 8.23: PSD of the 10 MHz Caesium Clock reference and the Reconstructed Clock signal

8.3.2 Bandpass Sampling

In this section, the performance of the system when bandpass sampling is used to sample the RF signal is presented.

In this first application of the bandpass sampling scheme, the distribution of an RF signal centred at 40.078 MHz clock is shown. Due to a hardware design flaw in the ADC/DAC FMC, namely in the board's oscillator, the sampling/reconstruction frequency has a maximum value of 62.5 MHz. This limits the reconstruction capabilities of higher frequencies, such as the 40.078 MHz. However, a frequency multiplier can be added, which gets as input reference the output signal generated from the DAC, to up-convert the frequency reference at the Rx node.

Two methods are employed to measure the performance of the distribution of high frequency signals, ie. above the Nyquist frequency. The methods described below are only applicable for the reconstruction of the signal in the Rx node.

The two methods are described as follows: One method uses a specific sampling frequency, lower than the system clock, and selected so that the reconstruction of the initial signal is performed by multiplying the sampling frequency by an integer value. For instance, if it is required to distributed at 40.078 MHz signal, this method

samples the RF signal at 30.0585 MHz. With this sampling frequency the input signal is digitally down-converted to an intermediate frequency centred 10.0195 MHz.

In the Rx node, the reconstruction of the signal to its initial centred frequency is performed by increasing four times the frequency, which up-converts the signal to the 40.078 MHz.

The second method, which up-converts the RF signal in the Rx node, is to use a signal generator with a 10 MHz input reference clock. In the Rx, the digital demodulator instead of up-converting the baseband signal to the intermediate frequency it up-converts to a 10 MHz signal. The 10 MHz signal reconstructed by the Rx node is then connected to input clock reference of the signal generator. The signal generator is configured to output a RF signal centred of 40.078 MHz, which is locked to the RF signal in Tx node.

In Figure 8.24, a block diagram of the second method described to reconstruct the reference signal in the Rx node is shown.

Nonetheless, by adding to the ADC/DAC FMC board an oscillator with a higher frequency the distributed signal can be reconstructed by the DAC removing the need for a frequency multiplier circuit, or signal generator.

The RF signal centred at 40.078 MHz input signal is generated, in the Tx node, by a signal generator from Rohde & Schwarz SML03 [157].

The sampling frequency is set to 62.5 MHz, which down-converts the input signal to a digital image centred at 22.422 MHz. The CORDIC algorithm is configured to generate a digital clock reference, the mixing frequency, centred at 22.422 MHz.

The phase-noise spectrum of the reference clock measured from the digital generator in the Tx node is shown in Figure 8.25. The RMS jitter calculated for the input signal, between the offset frequencies ranging from 5 Hz to 5 MHz, is ~37.9 ps.

In the Rx node, the CORDIC algorithm is configured to 10 MHz for this system configuration.

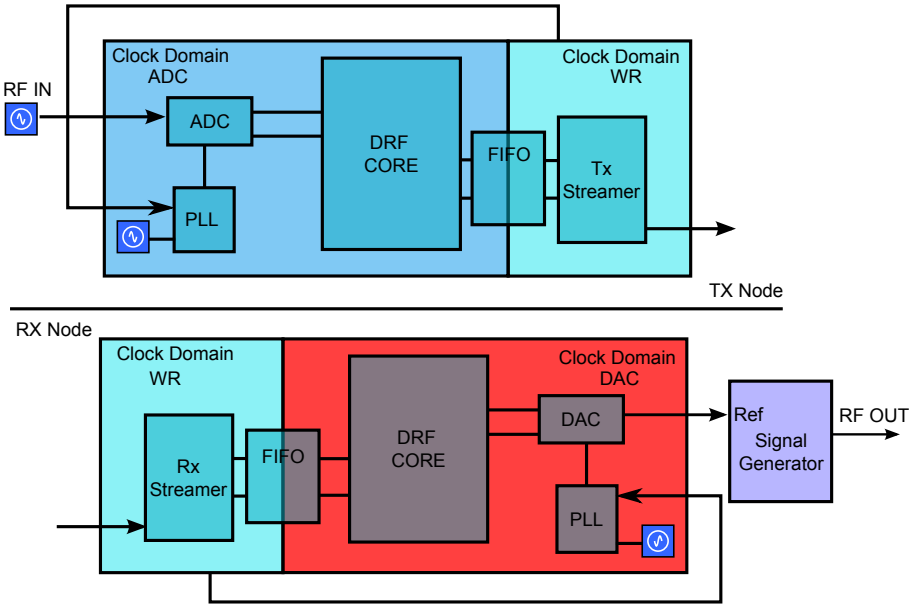


Figure 8.24: Bandpass sampling - Reconstruction Method

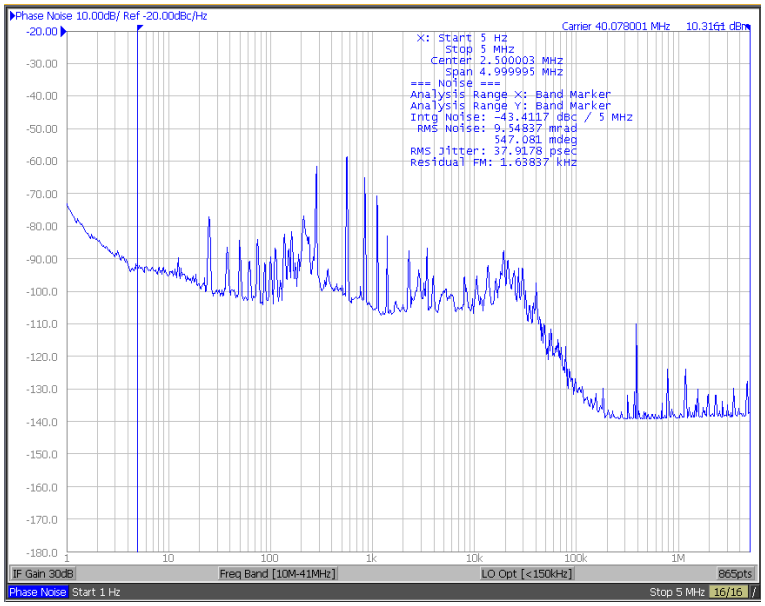


Figure 8.25: Phase-Noise Spectrum of the 40.078 MHz Tx Signal

The phase-noise spectrum of the reconstructed clock at the Rx node after being up-converted by the signal generator is shown in Figure 8.26. The measured RMS jitter in the reconstructed signal is ~8.2 ps.

The phase-noise spectra shown in Figure 8.25 and Figure 8.26 indicates that the reconstructed clock signal at the Rx node is more stable than the reference signal

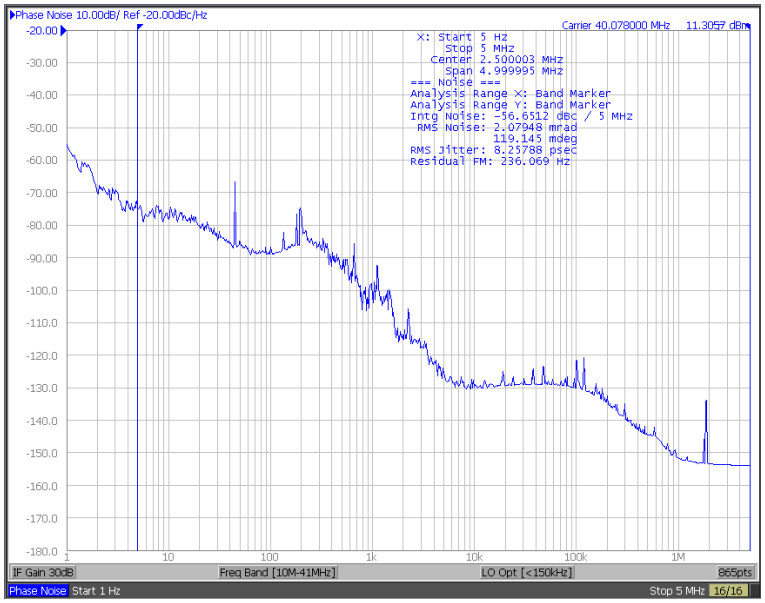


Figure 8.26: Phase-Noise Spectrum of the 40.078 MHz Rx Signal

sampled in the Tx side. Moreover, it is noticeable that the RMS jitter in Tx signal is mainly due to the frequency spurs, only observed at offset frequencies below ~20 Hz. These spurs are filtered out, in the Rx signal, due to the PLL's narrow bandwidth in the signal generator.

A second set of measurements is realised to evaluate the behaviour of the proposed system in distributing RF signals with carrier frequencies higher than the sampling clock. The effect of the jitter in the sampling clock during the bandpass sampling is reported.

The two RF signals to be distributed by the distributed RF system are centred at 100 MHz and 447.5 MHz. The reasons to select these two types of signals are the following:

- The 100 MHz signal, is typically used as reference clock by timing systems implemented worldwide [144].
- The 447.5 MHz, has the particularity of being down-converted by the bandpass sampling process to 10 MHz. This 10 MHz reference is then used as input reference for the signal generator, which then up-converts the signal to the

desired frequency reference.

The phase-noise spectrum of the 100 MHz reference clock is shown in Figure 8.27. The low phase-noise observed in the 100 MHz signal is due to the fact that the generator is locked to a Caesium clock.

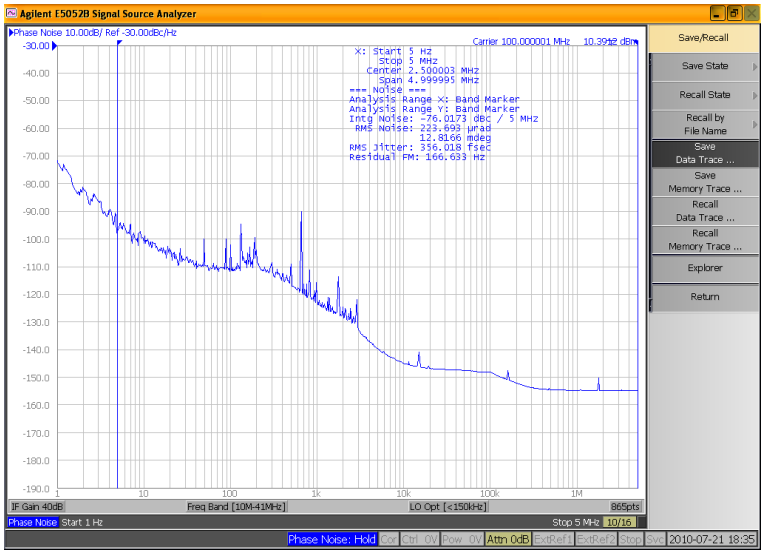


Figure 8.27: Phase-Noise Spectrum of the Tx 100 MHz signal

Sampling the 100 MHz input reference signal, with a 62.5 MHz sampling clock, creates a digital image of the input signal centred at 25 MHz. The CORDIC signal generator creates a frequency at 25 MHz to down-convert the IF signal to baseband by the digital demodulation process.

The phase-noise spectrum of the 100 MHz RF signal after the digital down-conversion process is shown in Figure 8.28. The phase noise measurement was done by feeding the digital signal converted by the ADC directly to the DAC, via serial interface.

The phase-noise spectrum of the signal shown in Figure 8.28 is of the down-converted digital signal that is outputted to the signal analyser using the remaining DAC in the FMC card.

The RMS jitter present in the down-converted signal is significant, around ~22 ps, for the Tx node and the Rx node. This excessive RMS jitter is mainly due the low quality of the sampling clock, that is most observable at low offset frequencies.

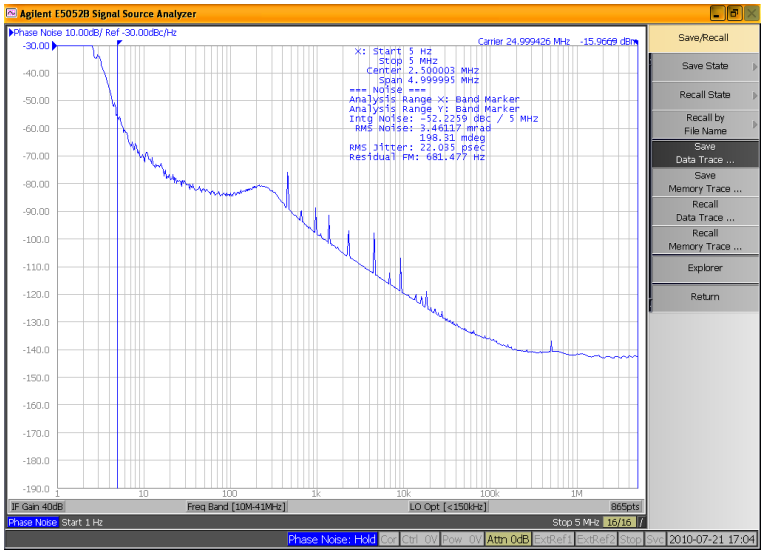


Figure 8.28: Phase-Noise Spectrum - Tx 100 MHz - IF = 25 MHz

As discussed earlier, excellent clock stability in bandpass sampling is critical to achieve good digital down-conversion. Jitter in the sampling clock takes special concern for bandpass high frequency signals [158].

In Figure 8.29, the phase noise of the reconstructed RF signal shows an identical shape of the RF signal in the Tx side. This results in a similar RMS jitter in both the down-converted signals for the Tx and Rx nodes. The proposed system design does not add significant jitter to the distributed signal in the digital domain.

However, due to the excess of jitter in the down-converted signal the up-conversion process did not produce a clock locked in phase and frequency to the signal at the Tx Node.

This work continues with the distribution of the RF signal centred at 447.5 MHz.

In Figure 8.30 the phase noise spectrum of the input RF signal in the Tx node is shown. The 447.5 MHz input signal is also obtained by a signal generator that is locked to the output of a Caesium clock, hence its low RMS jitter.

The input clock centred at 447.5 MHz is down-converted to an intermediate frequency in the digital domain centred 10 MHz signal. With the 62.5 MHz sampling clock, the 447.5 MHz reference is in the 14th Nyquist zone [112]. The phase-noise spectrum of

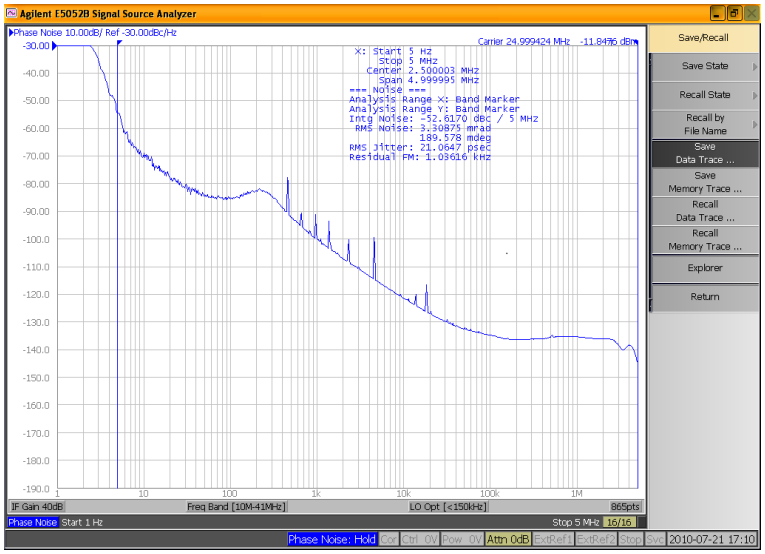


Figure 8.29: Phase-Noise Spectrum - Rx 100 MHz - IF = 25 MHz

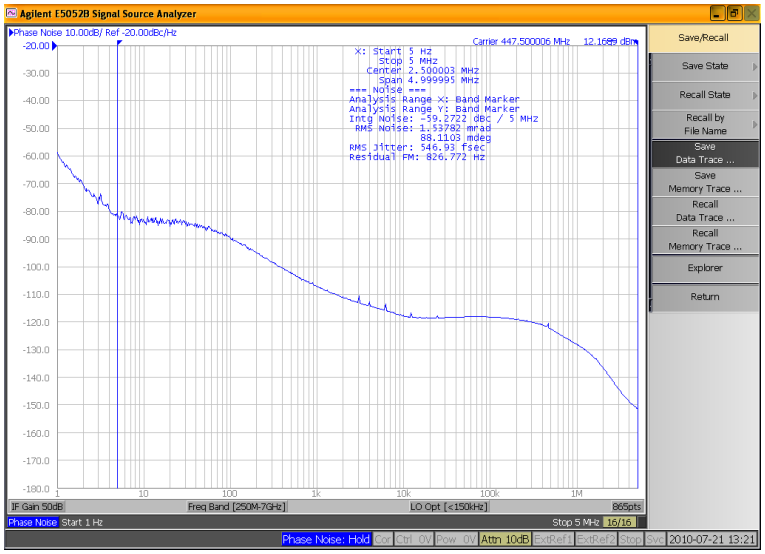


Figure 8.30: Phase-Noise Spectrum of the 447.5 MHz Tx signal

the intermediate frequency is shown in Figure 8.31.

It is clear from Figure 8.31 that the sampling clock down-converts the 447.5 MHz with and excess of RMS jitter. This issue can be reduced by using a higher quality oscillator than the one used in the FMC board.

In Figure 8.32 is shown the phase spectrum of the reconstructed signal at its intermediate frequency, which is 10 MHz.

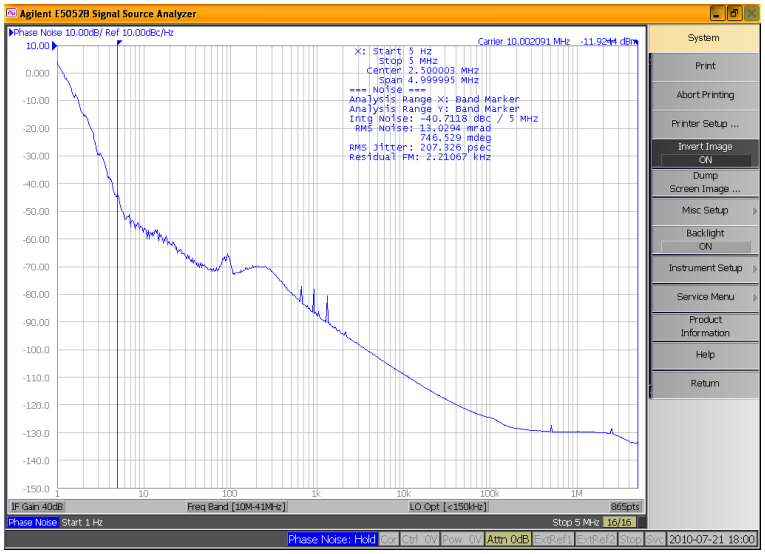


Figure 8.31: Phase-Noise Spectrum - Tx 447.5 MHz - IF = 10 MHz

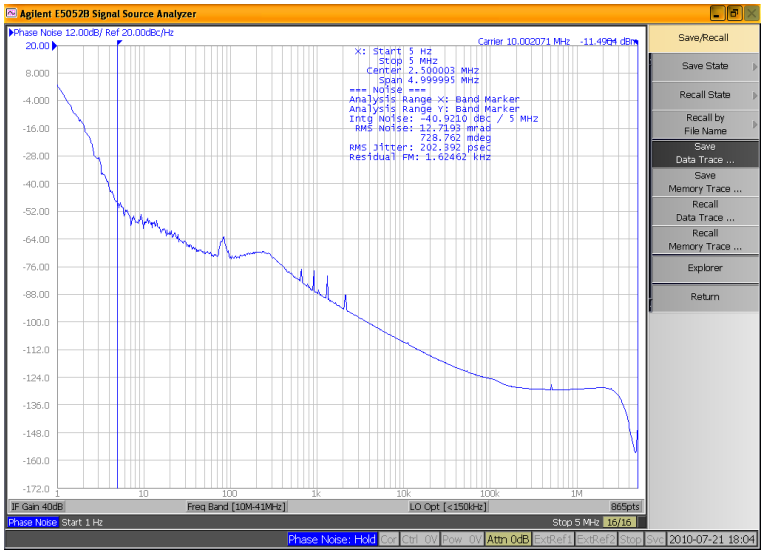


Figure 8.32: Phase-Noise Spectrum - Rx 447.5 MHz - IF = 10 MHz

Analysing Figure 8.31 and Figure 8.32 it is noticeable that they show identical phase-spectrum shape. The proposed system is able to distribute RF signals with high phase-noise and still maintain the phase and frequency characteristics of the input signal. This is due to the fact that the system was designed to distributed signals with bandwidth below 2 MHz.

To up-convert the frequency of the IF signal, the output of the DAC, which is a 10 MHz signal, is connected to the 10 MHz input reference clock of a signal generator.

8.3 Experimental Results

The phase-noise spectrum of the output from the signal generator is shown in Figure 8.33.

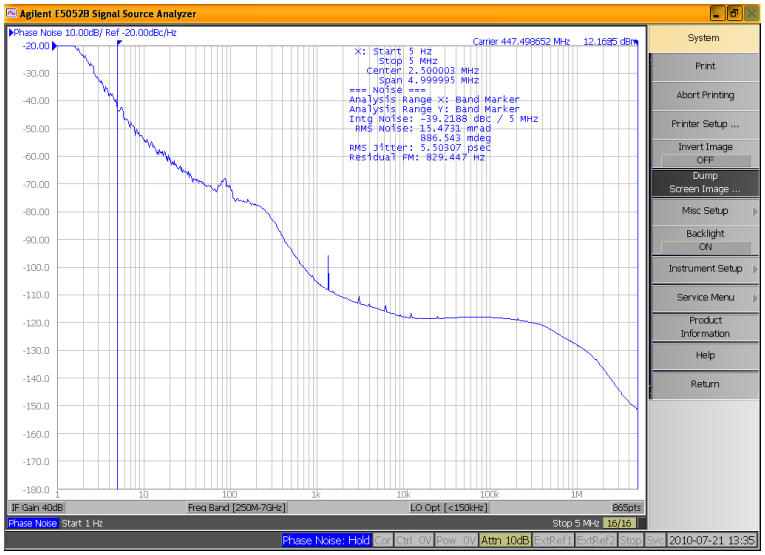


Figure 8.33: Phase-Noise Spectrum of the 447.5 MHz Rx Signal

The phase-noise spectrum of the reconstructed signal has an identical phase-noise spectrum as the 10 MHz down-converted signal in the Tx Node. However, the reconstructed signal is not phase and frequency locked with the signal at the Tx node. This can be observed in the centre frequency data displayed in Figure 8.31 and Figure 8.33. This is due to the clock sampling jitter added to the down-converted signal during the bandpass sampling process.

Nonetheless, the down-converted signals in the Tx and Rx nodes are locked in phase and frequency, which can be verified in Figure 8.31 and Figure 8.32.

Power Spectrum Density

Figure 8.34 provides the PSD of the intermediate frequency signal, at 22.42 MHz, resulting from sampling the 40.078 MHz input signal. The PSD shows the IF at the Tx node and in the Rx node.

The SNR estimated in the down-converted signals is ~56.26 dB. From Figure 8.34 frequency spurs at 22.42 MHz $\pm$ 500kHz and 22.42 MHz $\pm$ 1MHz are observable,

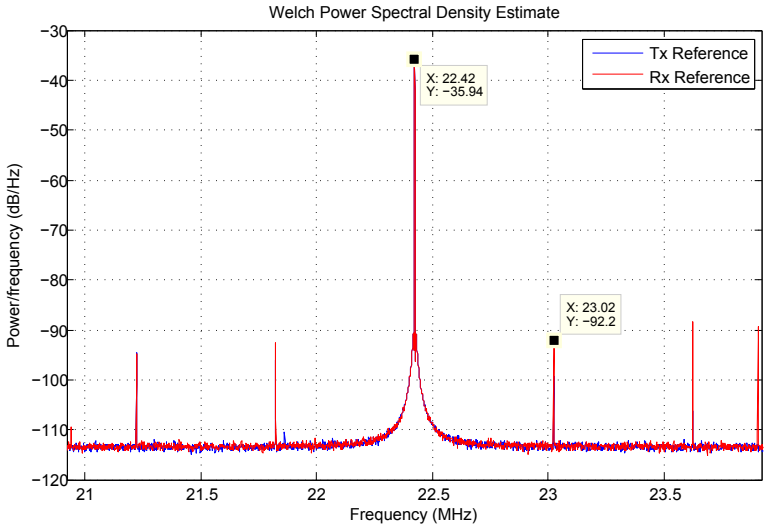


Figure 8.34: PSD of the downconverted 40.078 MHz

these frequency spurs are due to the intermodulation and harmonic processes.

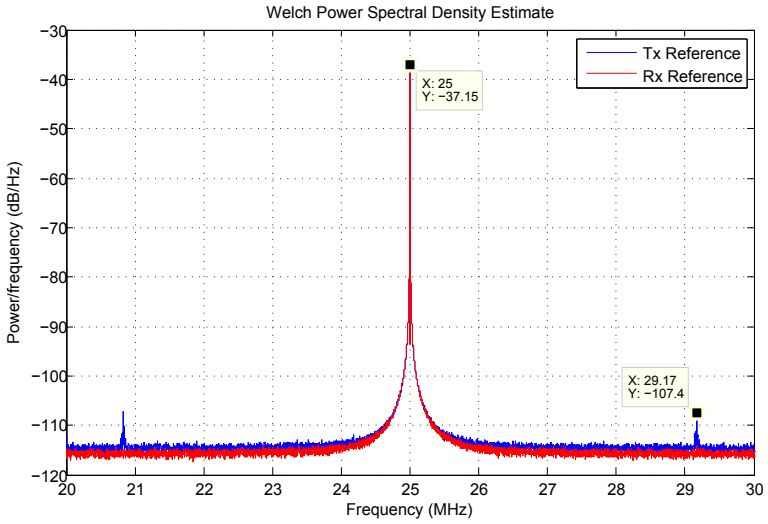


Figure 8.35: PSD of the downconverted 100 MHz

In Figure 8.35 the PSDs of the 100 MHz down-converted signal in the Tx node and the down-converted reconstructed signal at the Rx node are shown. The SNR for the 100 MHz down-converted signals is ~70.25 dB. In addition, Figure 8.35 shows frequency spurs located at 25 MHz ± 4.2MHz originated by the sampling conversion process.

Finally, Figure 8.36 the PSDs of the down-converted signal from the 447.5 MHz signal

in the Tx node and the distributed signal in the Rx node are shown.

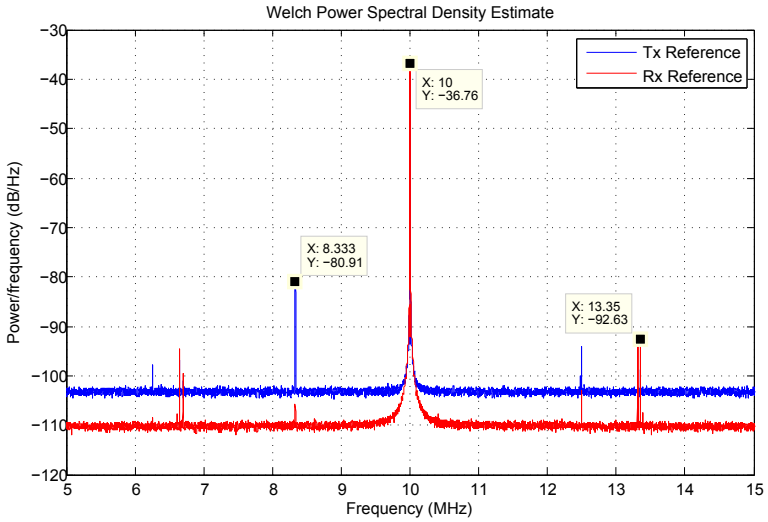


Figure 8.36: PSD of the downconverted 447.5 MHz

By sampling a reference signal located at a high frequency from the sampling clock, the noise floor of the down-converted signal increases due to the aliasing of the noise from DC to the reference signal.

The aliasing of high frequencies phenomenon is observed in Figure 8.36 where it is visible an increase of the noise floor, which is ~(-103) dB, when compared with signal at the Rx Node, which is ~(-110) dB.

The noise floor seen in the Rx node decreases due to the signal processing stages that removes the noise away from the bandwidth of the signal.

The SNR of the down-converted signal in the Tx node is -44.15 dB and -55.87 dB in the Rx Node.

8.4 Summary

In this chapter, the Distributed RF over White Rabbit system was experimentally demonstrated using the SPEC board, ADC/DAC FMC and standard single-mode fibre links. Additionally, the hardware implementation in an FPGA system of the digital blocks that formulate the distributed RF system has been discussed.

8.4 Summary

The hardware used in the experiments consisted of two SPEC boards and ADC/DAC FMC boards that are connected using the WR network. The system was tested using both Direct sampling and Bandpass sampling. In the direct sampling configuration, the proposed system has successfully distributed a 10 MHz clock reference. The reconstructed signal in the receiver node is locked in phase and frequency to the reference signal at the transmitter node. The measured RMS jitter of the distributed signal in the receiver node is approximately 5 ps. The Allan Deviation for the 10 MHz reconstructed signal is 28 ps at 1 Hz measurement interval and it goes down to 62.7 fs at 1800 seconds. These are good results, however, in terms of clock stability, the performance could be further improved by adding a PLL to the output of the DAC to clean further the jitter generated by the conversion process. To the best of the author's knowledge, this is the first report describing the successful distribution of a highly stable clock reference using Ethernet.

This chapter also shows the distribution of RF signals whose centre frequency is higher than the sampling clock. In this configuration, the input signal is down-converted, using bandpass sampling, to an intermediate frequency between DC and half the sampling frequency.

To test the bandpass sampling configuration three clock references, 40.078 MHz, 100 MHz and 477.5 MHz were selected to be distributed.

The up-conversion of the distributed RF signal to the initial frequency is done by a frequency multiplier, which uses the output from the DAC as a reference signal.

For the distribution of the 40.078 MHz the measured RMS jitter in the reconstructed signal was ~8.2 ps. However, for the distribution of the 100 MHz and 477.5 MHz, the distributed signals were not phase and frequency locked to their respective reference signals. This is due to the excess of jitter added by the sampling frequency, which is more noticeable when bandpass sampling high frequency signals.

Nonetheless, despite the excess of sampling jitter, which corrupts the carrier frequency phase and frequency, the 100 MHz down-converted signals (to 25 MHz) are

locked in frequency and phase in the transmitter and receiver nodes.

For the 477.5 MHz, the down-converted signals (to 10 MHz) are also locked in frequency and phase and the phase-noise spectrum shape is identical in both nodes.

However, the sampling jitter can be reduced by locking the system clock, thus the sampling clock, to a Caesium clock. Thereby, improving the performance of the implemented system in bandpass sampling configuration.

This newly proposed system, which is verified experimentally, shows a small improvement in the results relatively to other solutions that implement the same distribution functionality but using dedicated, complex and costly systems [24].

The work presented in Chapter 8 resulted in a publication for the IFCS 2013 international conference proceedings [159].

Chapter 9

Conclusions

The work reported in this thesis has addressed digital circuit design for high energy physics timing systems within the scope of the White Rabbit Project. The White Rabbit is a collaborative project that aims to use Ethernet as data and timing network with sub-nanosecond accuracy in a distributed configuration. An accurate clock synchronisation service is a primary and absolute requirement for all distributed real-time systems. The work presented in this thesis has two main contributions. The first main contribution is a novel digital circuit design capable of measuring clock's time difference with sub-picosecond time resolution. The circuit design proposed is a digital circuit based on the analogue DMTD circuit design by Allan [80].

The DDMTD circuit is capable of measuring time differences between two digital clocks with sub-picosecond time resolution. In addition, the proposed system can be implemented using FGPA systems, which brings advantages in terms of modularity and system integration. The DDMTD was first simulated and then implemented using a custom-made circuit board that integrates optical transceivers for gigabit Ethernet links, FPGAs and additional hardware required by the timing network. The DDMTD circuit implemented in the FPGA was analysed in two different configurations. The first design successfully showed the DDMTD circuit acting as a phase detector in a PLL design. The second configuration shows the DDMTD as system

clock phase shifter with picosecond time resolution.

The second main contribution of this work is a system architecture whose main purpose is to distribute RF signals, such as the one from a clock reference over a wide local network using Ethernet.

The proposed system architecture has advantages over existing dedicated network infrastructures that distribute clock references with low jitter, such as being a cost effective solution. The system validation was performed by simulation and experimental verifications.

To the best of the author's knowledge, the aforementioned contributions developed for timing and RF distribution, constitute a world first.

9.1 Thesis Overview

This thesis described in Chapter 2, CERN's accelerator complex followed by a detailed description of the current timing systems running for the LHC, the GMT and the TTC. A description of a dedicated timing system, the ALMA, used for radio-astronomy applications was also given, followed by a brief introduction to the operation of the GPS, a satellite synchronisation system widely used for a variety of timing applications. It was found that timing systems currently in operation for the accelerators chains at CERN, although achieving the required timing performance, the lack network bandwidth and the required automatic timing compensation. An alternative to upgrade the current timing network for future accelerators generation is required.

Chapter 3 presented how synchronisation is achieved in Gigabit Ethernet links. The chapter starts by describing the functionality of the layers specified by the IEEE 802.3 standard and how these layers exchange information among each other to achieve link synchronisation. Nonetheless, Ethernet is a non-synchronous network, in other words the clock signal travels from node-to-node but is not distributed from a node to all the

nodes in the network. In order to distribute a frequency reference through an Ethernet link, ITU-T created the Synchronous Ethernet (Sync-E) recommendation. This recommendation states a set of rules to enable frequency distribution over multiple networks nodes.

Moreover, to address the issue of timing distribution, the PTP standard was introduced as a data link layer protocol to synchronise distributed nodes in packet based network.

The White Rabbit project was introduced in the last section of Chapter 3. It aims to design a network based Ethernet network capable of delivering sub-nanosecond timing accuracy using standardised technologies such as Synchronous Ethernet and PTP. In addition, the White Rabbit's link delay model was presented as well as an overview of the external factors that contribute to the variation of the link's delay.

A set of solutions that aims to reduce the external factors influence in the White Rabbit's nodes timing accuracy was proposed. It was shown that using the standardised communication is not enough to achieve sub-nanosecond accuracy with single mode optical links for two main reasons. First, the link asymmetry between the up-stream and down-stream need to be compensated in the PTP engine, as the PTP standard assumes that the network link is symmetric. Second reason is that the packet data has a 8 ns granularity due to the 125 MHz network clock. In order to achieve a sub-nanosecond time accuracy, a circuit capable of measuring the phase variations between the transmitted and the recovered clock is required. Again, this difference needs to be compensated in the PTP engine.

Chapter 4 focused on the mathematical description of the timing signal and its phase noise analysis. The chapter describes the clock stability in both frequency and time domains, and how the different types of clock noise are characterised for each domain. This chapter provides two types of characterisation tools. One of the tools, the phase noise spectrum, is required to analyse the short term stability of a clock oscillator or the clock output of a PLL system. The second tool is used to characterise the

stability of a clock signal on the long term. This is measured by the Allan Variance or Modified Allan Variance.

The phase-noise spectrum of a Rubidium atomic clock was measured to characterise the power-laws of its clock signal. In addition, it was shown the Allan Variance of a Caesium oscillator and the characterisation of the various power-laws was given. The work presented in this chapter gave rise to one publications in conference proceedings [160].

In Chapter 5, a novel digital time difference measurement circuit was proposed. The DDMTD circuit is capable of measuring the time different between clock signals with high time resolution. It was shown how that the selection of the beat frequency generated by the Helper PLL influences the time resolution of the DDMTD. In addition, it was shown that this circuit is able to measure time difference between two digital clocks with femto-second time resolution. However, the output of circuit is characterised by glitches during the DDMTD clock edge transitions. To deal with this problem, three deglitching techniques were proposed. The performance of these techniques was evaluated and improved through the use of computer simulation. The simulations evaluated the performance of the different techniques with regards with the obtain their improvement in the circuit's accuracy. It was found that the Zero Count Selection deglitching algorithm improves the accuracy of the DDMTD circuit at an expense of using more area that the others algorithms. The work presented in this chapter gave rise to two publications in conference proceedings [161] [162].

In Chapter 6, the implementation of the DDMTD circuit in a PLL configuration was presented. In addition, the capabilities of the DDMTD circuit to phase shift clock signals with a picosecond time resolution were demonstrated. This was implemented on a custom FPGA circuit board. The circuit board was designed not just with the intention of implementing the DDMTD, but also all the remaining timing block required by the WR network. A detailed analysis of the various hardware blocks responsible for the synchronisation at the link layer was given.

The Chapter 6 continues with a brief overview of theory and design of PLLs. The components composing a PLL such as the phase detectors, and loop controllers were analysed. In Section 6.4, the design and FPGA's implementation of the Helper PLL and the DDMTD circuit as phase detector in a PLL configuration were presented. The behaviour of the DDMTD PLL's output clock phase-noise spectra and RMS jitter measurements was analysed and discussed for the loop controllers designed. The DDMTD circuit achieved great performance as a phase detector in a PLL configuration. The DDMTD PLL was able to clean the noisy recovered clock with 14 ps RMS jitter to a cleaner clock with 2.6 ps RMS jitter. In addition, the DDMTD's performance as phase shifter circuit with picosecond resolution has been demonstrated by phase shifting the feedback clock by 54 ps and 108 ps. The stability of the phase shifted clock is identical to the phase stability of the output clock obtained by the DDMTD's PLL.

The remaining two chapters of this work discuss the issue of distributing RF signals over relatively long distances, with a maximum range of 10 km, using non-dedicated links and maintaining phase and frequency information of the source signal, with minimal latency.

Chapter 7 introduces the proposed Digital RF over Ethernet system architecture. The proposed system architecture uses AD and DA converters to sample and reconstruct the reference RF signal. In addition, the system is implemented using existing network infrastructure for timing and data distribution. This requirement makes this architecture very appealing as it results in a reduction of costs in installation, maintenance and operation of the network. The chapter follows with a detailed description of the system modelling and hardware components that compose the proposed system. Namely, a theoretical background of the sampling theorem and the bandpass sampling technique are given.

This is followed by a description of the ADC and DAC architectures highlighting their advantages and disadvantages. The next sections describe the signal processing

stages that the digital stream undergoes before being packeted to an Ethernet frame and distributed to the Rx nodes.

In the last section of Chapter 7, the performance of the proposed distributed RF system by a simulation model is examined. The simulation model assists in the optimisation of the parameters of the analogue and digital components, so that the distributed system architecture reaches the desired system performance. In particular, contribution of the various noise sources, such as the quantisation noise, the clock jitter in the sampling process, the numerical oscillators and filtering action in the overall performance of the system was analysed. The performance of the system is characterised by measuring the SNR and phase-noise spectrum of the reconstructed RF signal. Simulations verified the applicability of the proposed system architecture in distributing RF signals with low RMS jitter. It was found that distributing highly stable RF signals, such the clocks generated by Caesium clocks, can be very efficient and cost-effective using the proposed system architecture. The RMS jitter of the reconstructed signal can be as low as 4 ps. For these RF signals the required gigabit network bandwidth is approximately 8% of the available network capacity. However, the performance of the bandpass sampling was shown to be very dependent on the RMS jitter of the sampling clock, which is more evident for high frequencies.

In Chapter 8, the implementation of the proposed distributed RF system in FPGA was presented. The FPGA board, described in Section 8.1, is designed in the scope of the White Rabbit project using commercial generic components. In Section 8.2 describes the RTL blocks implemented in the FPGA. Various design strategies to fit and optimised the design in the resources available in the FPGA were discussed. The performance of proposed system in distributing signals with different frequency using both direct and bandpass sampling techniques is validated experimentally.

To the best of the author's knowledge, it is the first time that a stable clock reference signal is successfully and synchronously distributed using an Ethernet network. Although the results in relative terms show a significant increase in the RMS jitter,

as the RMS jitter between the Tx node and the Rx node increases by almost seven times, the obtained result is very good as the system is able to reconstruct a signal synchronous with the Caesium reference with an RMS jitter of ~7.2 ps. This proposed non-dedicated system shows an increase of performance when compared with some dedicated distributed RF signals systems [163].

9.2 Proposals for Future Work

Despite this work contributions, experimental studies and theoretical analysis for optimising the proposed DDMTD circuit, the White Rabbit network, and the distributed RF system architecture are still an open area of research.

The work investigations gave support to further optimisations these different topics. A number of research items were identified, but could not be carried out as part of this work.

However, they would be interesting for future contributions, namely:

On the Digital Dual Mixer Time Difference:

- The locking phase noise level of the DDMTD can be further optimised to achieve a lower RMS jitter. This would increase both the bandwidth of the PLL system and the phase stability of the filtered clock source.
- Subject the DDMTD circuit to a wide range of external environmental factors such as temperature and vibrations, and analyse their impact on the DDMTD circuit performance.
- Examining the use of a counter with greater phase stability as time difference measurer, particularly when the DDMTD is configured in the femtosecond time resolution range.

On the Timing Network:

- Upgrade the network for 10 GbE networks. The White Rabbit network was designed to work in Gigabit network, however, 10 GbE networks are already being deployed on the network backbone. This means that deploying WR over 10 GbE will become more cost effective over time. However, it must be pointed out that the synchronisation of the 10 GbE is different from the one in GbE network, particularly in the synchronisation scheme: the GbE implements 10B/8B encoding while the 10 GbE uses 64B/66B encoding.

On the Distributed Radio Frequency over White Rabbit System:

- Analyse the Distributed RF system performance under adverse environment variations. The performance of the system reported in this work occurred under "normal" external factors. Nonetheless, the proposed distributed system can only be deployed in the CERN if they are prepared to operate under a wide-range of temperature variations and vibrations.

- Phase-Noise Reduction in the Distributed signal. The phase-noise of the recovered clock can be further reduced by employing oscillators with higher stability or by optimising the routing in the FPGA fabric. Optimising the routing in the FPGA decreases critical paths that the clock signal id required to travel, reducing the timing uncertainty or jitter. This results in more stable reconstructed signal in the Rx node. The automatic routing planner of the FPGA does an efficient work in mapping the design in the FPGA fabric, but there are still space for optimisation by doing some specific FPGA's routing manually.

- Multi Channel Signal Distribution. The distributed RF system proposed in this work distributes a single tone RF signal. However, there are applications that require multiple channel distribution. In applications where network bandwidth is available, is feasible to distributed multiple channel using the proposed system with small changes to the system architecture to accommodate multiple channel streaming.

- Improve Filtering blocks area and power consumption. The critical path bottleneck of the Distributed RF design are the filtering blocks, implemented by polyphase structures. Using multiplierless structures to implement the filtering block would reduce the area and improve power consumption. In addition, the critical path of the design would also benefit from the multiplierless structures.

- Upgrade DAC's sampling oscillator. Verify the performance of the system by using higher frequency sampling oscillators in the DAC FMC. This will allow the reconstruction of higher signal frequencies, and provide the current hardware with a wider frequency operation, without the need for a PLL circuit to up-convert the reconstructed signal.

In conclusion, this thesis confirmed the practicability of timing distribution with sub-nanosecond accuracy and the distribution of Radio-Frequency signals over Ethernet. Furthermore, the network is able to perform timing and radio-frequency distribution with the proposed requirements and as well as the exchange of non-critical data .

References

[1] R. Robrock, "Quartz crystal applications in digital transmission," in *25th Annual Symposium on Frequency Control*, 1971, pp. 70 – 73.

[2] A. Clairon, P. Laurent, G. Santarelli, S. Ghezali, S. Lea, and M. Bahoura, "A cesium fountain frequency standard: preliminary results," *IEEE Transactions on Instrumentation and Measurement*, vol. 44, no. 2, pp. 128 –131, April 1995.

[3] S. S. Wiltermuth and C. Heath, "Synchrony and Cooperation," *Psychological Science*, vol. 20, no. 1, pp. 1–5, 2009. [Online]. Available: http://pss.sagepub.com/content/20/1/1.abstract

[4] N. W. F. Bode, J. J. Faria, D. W. Franks, J. Krause, and A. J. Wood, "How perceived threat increases synchronization in collectively moving animal groups," *Proceedings of the Royal Society B: Biological Sciences*, vol. 277, no. 1697, pp. 3065–3070, 2010.

[5] S. Bregni, *Synchronization of Digital Telecommunications Networks*, Wil, Ed. Wiley, 2002.

[6] B. Taylor, "TTC distribution for LHC detectors," *IEEE Transactions on Nuclear Science*, vol. 45, no. 3, pp. 821–828, June 1998.

[7] E. J. Wollack, "What is the Universe Made Of?" January 2010. [Online]. Available: http://map.gsfc.nasa.gov/universe/uni_matter.html

[8] L. Evans and P. Bryant, "LHC Machine," *Journal of Instrumentation*, vol. 3, no. 08, p. S08001, 2008. [Online]. Available: http://stacks.iop.org/1748-0221/3/i=08/a=S08001

[9] CERN, "The Accelerator Complex," 2010. [Online]. Available: http://user.web.cern.ch/public/en/Research/AccelComplex-en.html

[10] P. W. Higgs, "Broken Symmetries and the Masses of Gauge Bosons," *Phys. Rev. Lett.*, vol. 13, pp. 508–509, October 1964.

[11] J. A. Bagger, "Supersymmetry at LHC and NLC," in *5th International Conference on Supersymmetries in Physics*, vol. 62, no. 1–13, 1998, pp. 23–35.

[12] G. Bhattacharyya, A. Datta, S. K. Majee, and A. Raychaudhuri, "Exploring the universal extra dimension at the LHC," *Nuclear Physics B*, vol. 821, pp. 48–64, 2009.

[13] The ALICE Collaboration, "The ALICE experiment at the CERN LHC," *Journal of Instrumentation*, vol. 3, no. 08, p. S08002, 2008. [Online]. Available: http://stacks.iop.org/1748-0221/3/i=08/a=S08002

[14] The LHCb Collaboration, "The LHCb Detector at the LHC," *Journal of Instrumentation*, vol. 3, no. 08, p. S08005, 2008. [Online]. Available: http://stacks.iop.org/1748-0221/3/i=08/a=S08005

[15] The TOTEM Collaboration, "The TOTEM Experiment at the CERN Large Hadron Collider," *Journal of Instrumentation*, vol. 3, no. 08, p. S08007, 2008. [Online]. Available: http://stacks.iop.org/1748-0221/3/i=08/a=S08007

[16] The LHCf Collaboration, "The LHCf detector at the CERN Large Hadron Collider," *Journal of Instrumentation*, vol. 3, no. 08, p. S08006, 2008. [Online]. Available: http://stacks.iop.org/1748-0221/3/i=08/a=S08006

[17] J. C. Bau, M. Jonker, and J. Lewis, "The Central Beam and Cycle Management of the CERN Accelerator Complex," in *7th Biennial International Conference*

on Accelerator and Large Experimental Physics Control Systems, October 1999, pp. 14–17.

[18] F. J. Ballester, D. Dominguez, J. Gras, J. Lewis, J. Savioz, and J. Serrano, "An FPGA Based Multiprocessing CPU for Beam Synchronous Timing in CERN's SPS and LHC," in *9th International Conference on Accelerator and Large Experimental Physics Control Systems*, October 2003, pp. 113–116, CERN-AB-2003-112-CO.

[19] J. Lewis, "LHC General Machine Timing (GMT)," Tech. Rep., August 2007. [Online]. Available: http://indico.cern.ch/getFile.py/access?sessionId=1&resId=1&materialId=0&confId=20321

[20] P. Alvarez-Sanchez, D. Dominguez, Q. King, J. Lewis, J. Serrano, and B. Todd, "PLL Usage in the General Machine Timing System for the LHC," in *9th International Conference on Accelerator and Large Experimental Physics Control Systems*, November 2003, pp. 114–118, CERN-AB-2003-114-CO.

[21] P. Alvarez-Sanchez, D. Dominguez, J. Lewis, and J. Serrano, "Nanosecond Level UTC Timing Generation and Stamping in CERN's LHC," in *9th International Conference on Accelerator and Large Experimental Physics Control Systems*, October 2003, pp. 119–123, CERN-AB-2003-111-CO.

[22] J. Troska, E. Corrin, T. Rohlev, J. Varela, and Y. Kojevnikov, "Implementation of the timing, trigger and control system of the CMS experiment," in *IEEE-NPSS Conference on Real Time*. Washington, DC, USA: IEEE Computer Society, 2005, pp. 24–27.

[23] T. H. Toifl, P. Moreira, and A. Marchioro, "Measurements of radiation effects on the timing, trigger and control receiver (TTCrx) ASIC," *6th Workshop on Electronics for LHC Experiments*, pp. 226–230, 2000.

[24] I. Papakonstantinou, C. Soos, S. Papadopoulos, S. Detraz, C. Sigaud, P. Stejskal, S. Storey, J. Troska, F. Vasey, and I. Darwazeh, "A Fully Bidirectional

Optical Network With Latency Monitoring Capability for the Distribution of Timing-Trigger and Control Signals in High-Energy Physics Experiments," *IEEE Transactions on Nuclear Science*, vol. PP, no. 99, pp. 1–13, March 2011.

[25] J. F. Cliche and B. Shillue, "Precision timing control for radioastronomy: maintaining femtosecond synchronization in the Atacama Large Millimeter Array," *IEEE Control Systems*, vol. 26, no. 1, pp. 19–26, February 2006.

[26] B. Shillue, "Atacama Large Millimeter Array photonic local oscillator: femtosecond-level synchronization for radio astronomy," in *IEEE International Frequency Control Symposium*, June 2010, pp. 569–571.

[27] Y. Y. Jau and W. Happer, "All-photonic clock: Laser-atomic oscillator," in *IEEE International Frequency Control Symposium,*, May 2008, pp. 665 –668.

[28] IEEE, "Global positioning system-the newest utility," *IEEE Aerospace and Electronic Systems Magazine*, vol. 15, no. 10, pp. 89–95, October 2000.

[29] J. Jasperneite, P. Neumann, M. Theis, and K. Watson, "Deterministic real-time communication with switched Ethernet," in *IEEE International Workshop on Factory Communication Systems*, 2002, pp. 11–18.

[30] J. Chen, Z. Wang, and Y. Sun, "Real-time capability analysis for switch industrial Ethernet traffic priority-based," in *Proceedings of the 2002 International Conference on Control Applications*, vol. 1, 2002, pp. 525 – 529 vol.1.

[31] E. Jasperneite and P. Neumann, "Switched Ethernet for factory communication," in *IEEE International Conference on Emerging Technologies and Factory Automation*, 2001, pp. 205 –212 vol.1.

[32] P. Pedreiras, R. Leite, and L. Almeida, "Characterizing the real-time behavior of prioritized switched-Ethernet," in *2nd Workshop on Real-Time LAN's in the Internet Age*, 2003.

[33] Z. Wang, Y. Song, J. Chen, and Y. Sun, "Real time characteristics of Ethernet and its improvement," in *World Congress on Intelligent Control and Automation*, vol. 2, 2002, pp. 1311 – 1318 vol.2.

[34] J. Loeser and H. Haertig, "Low-latency hard real-time communication over switched Ethernet," in *16th Euromicro Conference on Real-Time Systems*, June 2004, pp. 13–22.

[35] B. Choi, S. Song, N. Birch, and J. Huang, "Probabilistic approach to switched Ethernet for real-time control applications," in *Proceedings 7th International Conference on Real-Time Computing Systems and Applications*, 2000, pp. 384 –388.

[36] IEEE Std 802.3bf, "IEEE Standard for Information technology–Telecommunications and information exchange between systems–Local and metropolitan area networks–Specific requirements part 3: Carrier Sense Multiple Access with Collision Detection (CSMA/CD) Access Method and Physical Layer Specifications," IEEE, Tech. Rep., 2011.

[37] A. Tanenbaum, *Computer Networks*, 4th ed. Prentice Hall Professional Technical Reference, 2002.

[38] A. X. Widmer and P. A. Franaszek, "A DC-balanced, partitioned-block, 8B/10B transmission code," *IBM Journal of research and development*, vol. 27, pp. 440–451, September 1983.

[39] J. Ferrant, M. Gilson, S. Jobert, M. Mayer, M. Ouellette, L. Montini, S. Rodrigues, and S. Ruffini, "Synchronous Ethernet: a method to transport synchronization," *IEEE Communications Magazine*, vol. 46, no. 9, pp. 126–134, September 2008.

[40] ITU, *G.8261 - Timing and synchronization aspects in packet networks*, ITU Std., 2008.

[41] ——, *G.811 - Timing characteristics of primary reference clocks*, ITU Telecommunication Standardization Sector Std., 1997. [Online]. Available: http://www.itu.int/rec/T-REC-G.811/e

[42] S. Rodrigues, "IEEE-1588 and Synchronous Ethernet in Telecom," in *IEEE International Symposium on Precision Clock Synchronization for Measurement, Control and Communication*, October 2007, pp. 138–142.

[43] IEEE, *IEEE 1588-2008 Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control Systems*, IEEE Instrumentation and Measurement Society Std., 2008.

[44] P. Loschmidt, R. Exel, A. Nagy, and G. Gaderer, "Limits of synchronization accuracy using hardware support in IEEE 1588," in *IEEE International Symposium on Precision Clock Synchronization for Measurement, Control and Communication*, September 2008, pp. 12–16.

[45] ITU, *G.813 - Timing characteristics of SDH equipment slave clocks (SEC)*, http://www.itu.int/rec/T-REC-G.813-200506-I!Cor1/en, ITU Std., 2006.

[46] J. Eidson, J. Mackay, G. Garner, and V. Skendzic, "Provision of Precise Timing via IEEE 1588 Application Interfaces," in *IEEE International Symposium on Precision Clock Synchronization for Measurement, Control and Communication*, October 2007, pp. 1–6.

[47] L. D. Vito, S. Rapuano, and L. Tomaciello, "One-way delay measurement: State of the art," *IEEE Transactions on Instrumentation and Measurement*, vol. 57, no. 12, pp. 2742–2750, December 2008.

[48] ITU, *G.652 - Characteristics of a single-mode optical fibre and cable*, ITU Std., 2009.

[49] V. Alwayn, *Optical Network Design and Implementation*. Cisco Press, 2004.

[50] H.-J. Yoon and J.-G. Kim, "622-Mb/s Bidirectional SFP Optical Transceiver Using an Integrated WDM Optical Subassembly," *IEEE Transactions on Advanced Packaging*, vol. 31, no. 2, pp. 423 –428, May 2008.

[51] H. Kogelnik, L. Nelson, and R. M. Jopson, "Tutorial: polarization mode dispersion impairment in lightwave transmission systems," in *Optical Fiber Communication Conference and Exhibit*, March 2002, p. 194.

[52] J. Gowar, *Optical communication systems*, ser. Optoelectronics. Prentice Hall, 1993.

[53] M. Hamp, J. Wright, M. Hubbard, and B. Brimacombe, "Investigation into the temperature dependence of chromatic dispersion in optical fiber," *IEEE Photonics Technology Letters*, vol. 14, no. 11, pp. 1524–1526, November 2002.

[54] Y. Kang, "Calculations and Measurements of Raman Gain Coefficients of Different Fiber Types," Master's thesis, The Faculty of the Virginia Polytechnic Institute and State University, 2002.

[55] P. Khare and A. Swarup, *Engineering Physics: Fundamentals & Modern Applications*. Jones And Bartlett P, 2009.

[56] Corning Incorporated. SMF-28 Datasheet. http://www.corning.com/opticalfiber/products/SMF-28e+_fiber.aspx.

[57] P. Andre, A. Pinto, and J. Pinto, "Effect of temperature on the single mode fibers chromatic dispersion," in *Proceedings of the Microwave and Optoelectronics Conference*, vol. 1, September 2003, pp. 231–234.

[58] G. Ghosh, "Temperature dispersion of refractive indexes in some silicate fiber glasses," *IEEE Photonics Technology Letters*, vol. 6, no. 3, pp. 431–433, March 1994.

[59] ——, *Handbook of thermo-optic coefficients of optical materials with applications*. Academic Press, 1998.

[60] K.-C. Lin, C.-J. Lin, and W.-Y. Lee, "Effects of gamma radiation on optical fibre sensors," *IEE Proceedings on Optoelectronics*, vol. 151, no. 1, pp. 12 – 15, February 2004.

[61] S. M. Ross, *Introduction to Probability and Statistics for Engineers and Scientists*, 2nd ed., E. A. Press, Ed. Elsevier Academic Press, 2009.

[62] E. Rubiola, *Phase Noise and Frequency Stability in Oscillators*, ser. The Cambridge RF and Microwave Engineering. Cambridge University Press, 2010.

[63] D. B. Sullivan, D. W. Allan, D. A. Howe, and F. L. Walls, "Characterization of Clocks and Oscillators," NIST Technical Note 1337, Tech. Rep., 1990.

[64] ISO, *International vocabulary of metrology – Basic and general concepts and associated terms*, International Organization for Standardization Std., 2007. [Online]. Available: http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=45324

[65] D. W. Allan, N. Ashby, and C. C. Hodge, "The Science of Timekeeping," *Hewlett-Packard*, 1997.

[66] J. Devore, *Probability and Statistics for Engineering and the Sciences*. Cengage Learning, 2008.

[67] J. A. Barnes, A. R. Chi, L. S. Cutler, D. J. Healey, D. B. Leeson, T. E. McGunigal, J. A. Mullen, W. L. Smith, R. L. Sydnor, R. Vessot, and G. M. R. Winkler, "Characterization of Frequency Stability," *IEEE Transactions on Instrumentation and Measurement*, vol. IM-20, no. 2, pp. 105–120, May 1971.

[68] IEEE, "IEEE Standard Definitions of Physical Quantities for Fundamental Frequency and Time Metrology - Random Instabilities," *IEEE Std 1139-1999*, 1999.

[69] W. J. Riley, "Handbook of Frequency Stability Analysis," *NIST Special Publication*, 2007. [Online]. Available: http://www.stable32.com/Handbook.pdf

[70] Agilent Technologies, "E5052B SSA Signal Source Analyzer." [Online]. Available: http://www.home.agilent.com/agilent/product.jspx?id=1081579&pageMode=OV&pid=1081579

[71] D. Allan, "Statistics of atomic frequency standards," *Proceedings of the IEEE*, vol. 54, no. 2, pp. 221–230, February 1966.

[72] J. Rutman, "Characterization of Phase and frequency instabilities in precision frequency sources: Fifteen years of progress," *Proceedings of the IEEE*, vol. 66, no. 9, pp. 1048–1075, September 1978.

[73] P. Lesage and C. Audoin, "Characterization of Frequency Stability: Uncertainty due to the Finite Number of Measurements," *IEEE Transactions on Instrumentation and Measurement*, vol. 22, no. 2, pp. 157–161, June 1973.

[74] Spectracom, "Pendulum CNT-91 Timer." [Online]. Available: http://www.spectracomcorp.com/ProductsServices/SignalTest/FrequencyAnalyzersCounters/CNT-9191RTimerCounterAnalyzerCalibrator/tabid/1283/Default.aspx

[75] D. W. Allan and J. A. Barnes, "A modified "Allan variance" with Increased Oscillator Characterization Ability," in *Proceedings of the 35th Annual Frequency Control Symposium*, 1981, pp. 470–475.

[76] G. Zampetti, "Synopsis of timing measurement techniques used in telecommunications," in *24th Annual Precise Time and Time Interval (PTTI)*, June 1993, pp. 313–326.

[77] ITU, *G.810 - Definitions and terminology for synchronization networks*, ITU Telecommunication Standardization Sector Std., 1996.

[78] R. Staszewski, S. Vemulapalli, P. Vallur, J. Wallberg, and P. Balsara, "Time-to-digital converter for RF frequency synthesis in 90 nm CMOS," in *IEEE Radio Frequency integrated Circuits (RFIC) Symposium*, June 2005, pp. 473 – 476.

[79] M. F. Wagdy and M. S. P. Lucas, "Errors in Sampled Data Phase Measurement," *IEEE Transactions on Instrumentation and Measurement*, vol. 34, no. 4, pp. 507–509, December 1985.

[80] D. Allan and H. Daams, "Picosecond Time Difference Measurement System," in *Proceedings of the 29th Annual Symposium on Frequency Control*, 1975, pp. 404–411.

[81] G. Brida, "High resolution frequency stability measurement system," *Review of Scientific Instruments*, vol. 73, no. 5, pp. 2171 – 2174, May 2002.

[82] P. Moreira, J. Serrano, T. Wlostowski, P. Loschmidt, and G. Gaderer, "White rabbit: Sub-nanosecond timing distribution over Ethernet," *International Symposium on Precision Clock Synchronization for Measurement, Control and Communication*, pp. 1–5, October 2009.

[83] D. Chen, W. Wang, and T. Kwasniewski, "Design Considerations for a Direct RF Sampling Mixer," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 54, no. 11, pp. 934–938, November 2007.

[84] Y. Hu, "CORDIC-based VLSI architectures for digital signal processing," *IEEE Signal Processing Magazine*, vol. 9, no. 3, pp. 16–35, July 1992.

[85] A. Cantoni, J. Walker, and T.-D. Tomlin, "Characterization of a Flip-Flop Metastability Measurement Method," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 54, no. 5, pp. 1032–1040, May 2007.

[86] PICMG, *AdvancedTCA©Base Specification*, PICMG PICMG 3.0, Rev. 3.0, March 2008.

[87] Altera, "Cyclone III FPGA Family Overview." [Online]. Available: http://www.altera.com/devices/fpga/cyclone3/overview/cy3-overview.html

[88] Axcen Photonics Corp, "AXGE-1254-0531," http://www.axcen.com.tw/products/BIDISFPLC/ISS-0901077%20AXGE-1254-053x%20_SFP-1000BX10-U4___V1.2.pdf.

[89] ——, "AXGE-3454-0531," http://www.axcen.com.tw/products/BIDISFPLC/ISS-0901081%20AXGE-3454-053x%20_SFP-1000BX10-D4___V1.2.pdf.

[90] Texas Instruments, "Gigabit Ethernet Serdes," http://www.ti.com/product/tlk1221.

[91] Analog Devices, "ADN4600 4.25 Gbps 8 x 8 Digital Crosspoint Switch." [Online]. Available: http://www.analog.com/en/broadband-products/digital-crosspoint-switches/adn4600/products/product.html

[92] ——, "AD5662: 2.7-5.5V, 16-Bit nanoDAC Converter." [Online]. Available: http://www.analog.com/en/digital-to-analog-converters/da-converters/ad5662/products/product.html

[93] ——, "AD9516-4 Datasheet." [Online]. Available: http://www.analog.com/en/clock-and-timing/clock-generation-and-distribution/ad9516-4/products/product.html

[94] Texas Instruments, "CDCM61004 - 1:4 Ultra Low Jitter Crystal-in Clock Generator," http://www.ti.com/product/cdcm61004.

[95] OpenCores, *Wishbone bus specification (rev. B4)*, OpenCores/OrSOC Std., 2010. [Online]. Available: http://cdn.opencores.org/downloads/wbspec_b4.pdf

[96] F. M. Gardner, *Phaselock Techniques*. John Wiley & Sons, 1979.

[97] C. R. Hogge, "A self correcting clock recovery circuit," *IEEE Transactions on Electron Devices*, vol. 32, no. 12, pp. 2704–2706, December 1985.

[98] J. Alexander, "Clock recovery from random binary signals," *IET Electronics Letters*, vol. 11, pp. 541–542, 1975.

[99] J. Lee, K. S. Kundert, and B. Razavi, "Analysis and modeling of bang-bang clock and data recovery circuits," *IEEE Journal of Solid-State Circuits*, vol. 39, no. 9, pp. 1571–1580, September 2004.

[100] J. Brown, "A Digital Phase and Frequency-Sensitive Detector," *Proceedings of the IEEE*, vol. 59, no. 4, pp. 717–718, April 1971.

[101] F. Gardner, "Charge-pump Phase-lock Loops," *IEEE Transactions on Communications*, vol. 28, no. 11, pp. 1849–1858, November 1980.

[102] D. Banerjee, *PLL Performance, Simulation, and Design.* Dean Banerjee Pubns, 2003.

[103] P. V. Brennan, *Phase-Locked Loops: Principles and Practice.* McGraw-Hill Professional, May 1996.

[104] D. White, "The noise bandwidth of sampled data systems," *IEEE Transactions on Instrumentation and Measurement*, vol. 38, no. 6, pp. 1036–1043, December 1989.

[105] M. de Carvalho, *Dynamical systems and automatic control.* Upper Saddle River, NJ, USA: Prentice-Hall, Inc., 1993.

[106] W. F. Egan, *Phase-Lock Basics.* Wiley-IEEE Press, 2007.

[107] R. E. Best, *Phase-Locked Loops: Design, Simulation, and Applications*, ser. McGraw-Hill professional engineering. McGraw-Hill, 2003.

[108] P. K. Hanumolu, M. Brownlee, K. Mayaram, and U. Moon, "Analysis of charge-pump Phase-Locked loops," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 51, no. 9, pp. 1665–1674, September 2004.

[109] J. Piqueira, S. Castillo-Vargas, and L. Monteiro, "Two-way master-slave double-chain networks: limitations imposed by linear master drift for second order PLLs as slave nodes," *IEEE Communications Letters*, vol. 9, no. 9, pp. 829–831, September 2005.

[110] E. Rubiola and V. Giordano, "Correlation-based Phase noise measurements," *Review of Scientific Instruments*, vol. 71, no. 8, pp. 3085 –3091, August 2000.

[111] J. Varela, "Timing and Synchronization in the LHC experiments," in *6th Workshop on Electronics for LHC Experiments*, September 2000.

[112] R. Vaughan, N. Scott, and D. White, "The theory of bandpass sampling," *IEEE Transactions on Signal Processing*, vol. 39, no. 9, pp. 1973–1984, September 1991.

[113] P. Meher, J. Valls, , T.-B. Juang, K. Sridharan, and K. Maharatna, "50 Years of CORDIC: Algorithms, Architectures, and Applications," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 56, no. 9, pp. 1893–1907, September 2009.

[114] H. Nyquist, "Certain Topics in Telegraph Transmission Theory," *Transactions of the American Institute of Electrical Engineers*, vol. 47, no. 2, pp. 617–644, 1928.

[115] C. E. Shannon, "Communication in the Presence of Noise," *Proceedings of the IRE*, vol. 37, no. 1, pp. 10–21, 1949.

[116] J. Liu, X. Zhou, and Y. Peng, "Spectral arrangement and other topics in first-order bandpass sampling theory," *Signal Processing, IEEE Transactions on*, vol. 49, no. 6, pp. 1260–1263, June 2001.

[117] Analog Devices, "Which ADC Architecture Is Right for Your Application?" [Online]. Available: http://www.analog.com/library/analogdialogue/archives/39-06/architecture.pdf

[118] D. Ribner, "A Comparison of Modulator Networks for High-Order Oversampled ΣΔ Analog-to-Digital Converters," *IEEE Transactions on Circuits and Systems*, vol. 38, no. 2, pp. 145–159, 1991.

[119] IEEE, *IEEE Standard 1241 for Terminology and Test Methods for Analog-to-Digital Converters*, IEEE Std., 2010.

[120] Analog Devices, "Understand SINAD, ENOB, SNR, THD, THD+N, and SFDR so You Don't Get Lost in the Noise Floor." [Online]. Available: http://www.analog.com/static/imported-files/tutorials/MT-003.pdf

[121] V. Considine, "Digital complex sampling," *Electronics Letters*, vol. 19, no. 16, pp. 608–609, 1983.

[122] G. Saulnier, C. M. Puckette, R. J. Gaus, R. J. Dunki-Jacobs, and T. Thiel, "A VLSI demodulator for digital RF network applications: Theory and Results," *IEEE Journal on Selected Areas in Communications*, vol. 8, no. 8, pp. 1500–1511, 1990.

[123] J. E. Volder, "The CORDIC Trigonometric Computing Technique," *IRE Transactions on Electronic Computers*, vol. 8, no. 3, pp. 330–334, 1959.

[124] J. S. Walther, "A unified algorithm for elementary functions," in *Proceedings of spring joint computer conference*. New York, NY, USA: ACM, 1971, pp. 379–385. [Online]. Available: http://doi.acm.org/10.1145/1478786.1478840

[125] J. E. Volder, "The Birth of CORDIC," *Journal of VLSI signal processing systems for signal, image and video technology*, vol. 25, pp. 101–105, June 2000.

[126] R. Andraka, "A Survey of CORDIC Algorithms for FPGA Based Computers," in *Sixth International Symposium on Field Programmable Gate Arrays*, ser. FPGA '98. New York, NY, USA: ACM, 1998, pp. 191–200. [Online]. Available: http://doi.acm.org/10.1145/275107.275139

[127] K. Kota and J. Cavallaro, "Numerical accuracy and hardware tradeoffs for CORDIC arithmetic for special-purpose processors," *IEEE Transactions on Computers*, vol. 42, no. 7, pp. 769–779, 1993.

[128] H. Dawid and H. Meyr, "The differential CORDIC algorithm: Constant scale factor redundant implementation without correcting iterations," *IEEE Transactions on Computers*, vol. 45, no. 3, pp. 307–318, 1996.

[129] M. Park, K. Kim, and J. A. Lee, "CORDIC-Based Direct Digital Synthesizer: Comparison with a ROM-Based Architecture in FPGA Implementation," *IEICE Transactions on Fundamentals*, vol. E83-A, no. 6, pp. 1282–1285, June 2000.

[130] J. Vankka, M. Kosunen, J. Hubach, and K. Halonen, "A CORDIC-based multicarrier QAM modulator," in *Global Telecommunications Conference*, vol. 1A, 1999, pp. 173–177.

[131] C. M. Rader, "A Simple Method for Sampling In-Phase and Quadrature Components," *IEEE Transactions on Aerospace and Electronic Systems*, vol. AES-20, no. 6, pp. 821–824, 1984.

[132] T. Ramstad, "Digital methods for conversion between arbitrary sampling frequencies," *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. 32, no. 3, pp. 577–591, 1984.

[133] L. Erup, F. M. Gardner, and R. Harris, "Interpolation in digital modems - Part II: Implementation and performance," *IEEE Transactions on Communications*, vol. 41, no. 6, pp. 998–1008, 1993.

[134] P. Vaidyanathan, "Multirate Digital Filters, Filter Banks, Polyphase Networks, and Applications: A Tutorial," *Proceedings of the IEEE*, January 1990.

[135] I. Kuon and J. Rose, "Measuring the Gap between FPGAs and ASICs," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 26, no. 2, pp. 203–215, February 2007.

[136] R. Hewlitt and E. S. Jr., "Canonical signed digit representation for FIR digital filters," in *IEEE Workshop on Signal Processing Systems*, 2000, pp. 416–426.

[137] K.-H. Chen and T.-D. Chiueh, "A Low-Power Digit-Based Reconfigurable FIR Filter," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 53, no. 8, pp. 617–621, 2006.

[138] G. Reitwiesner, "Binary Arithmetic," *Advances in Computers*, vol. 1, no. C, pp. 231–308, 1960.

[139] H. Samueli, "An improved search algorithm for the design of multiplierless FIR filters with powers-of-two coefficients," *IEEE Transactions on Circuits and Systems*, vol. 36, no. 7, pp. 1044–1047, 1989.

[140] Y. C. Lim, J. Evans, and B. Liu, "Decomposition of binary integers into signed power-of-two terms," *IEEE Transactions on Circuits and Systems*, vol. 38, no. 6, pp. 667–672, 1991.

[141] R. Hashemian, "A new method for conversion of a 2's complement to canonic signed digit number system and its representation," in *Conference on Signals, Systems and Computers*, 1996, pp. 904–907 vol.2.

[142] Spiral, "Multiplierless Constant Multiplication." [Online]. Available: http://www.spiral.net/hardware/multless.html

[143] J. Shyh-Jye, J. Kai-Yuan, C. Hsiao-Yun, and W. An-Yeu, "Multiplierless multirate decimator/interpolator module generator," in *IEEE Asia-Pacific Conference on Advanced System Integrated Circuits*, August 2004, pp. 58–61.

[144] M. Lombardi, L. Nelson, A. Novick, and V. Zhang, "Time and frequency measurements using the global positioning system," *International Journal of Metrology*, 2001. [Online]. Available: http://www.world-cal.com/pdf_files/1424.pdf

[145] P. Gamage, A. Nirmalathas, C. Lim, D. Novak, and R. Waterhouse, "Experimental Demonstration of the Transport of Digitized Multiple Wireless Systems Over Fiber," *IEEE Photonics Technology Letters*, vol. 21, no. 11, pp. 691–693, June 2009.

[146] J. McClellan and T. Parks, "A personal history of the Parks-Mcclellan algorithm," *Signal Processing Magazine, IEEE*, vol. 22, no. 2, pp. 82–86, 2005.

[147] R. Schafer and L. Rabiner, "A Digital Signal Processing Approach to Interpolation," *Proceedings of the IEEE*, vol. 61, no. 6, pp. 692–702, June 1973.

[148] "Open hardware repository," http://www.ohwr.org/.

[149] 4DSP, "FMC150." [Online]. Available: http://www.4dsp.com/FMC150.php

[150] Texas Instruments, "ADS62P49 - Dual Channel 14 Bit, 250 MSPS ADC," http://www.ti.com/product/ADS62P49.

[151] IEEE, "IEEE Standard for Verilog Hardware Description Language," *IEEE Std 1364-2005 - Revision of IEEE Std 1364-2001*, 2006.

[152] ——, "IEEE Standard VHDL Language Reference Manual," *IEEE Std 1076-2008 - Revision of IEEE Std 1076-2002*, January 2009.

[153] Xilinx, "Spartan-6 FPGA DSP48A1 Slice User Guide," www.xilinx.com/support/documentation/user_guides/ug389.pdf.

[154] ——, "Spartan-6 FPGA GTP Transceivers," www.xilinx.com/support/documentation/user_guides/ug386.pdf.

[155] "Cesium frequency standard," http://www.symmetricom.com/products/frequency-references/cesium-frequency-standard/Cs4000/.

[156] W. D. Grover, "A new method for clock distribution," *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications*, vol. 41, no. 2, pp. 149–160, February 1994.

[157] Rohde & Schwarz, "SML03 Signal Generator," http://www.metrictest.com/catalog/pdfs/product_pdfs/rs_sml01.pdf.

[158] M. Patel, I. Darwazeh, and J. O'Reilly, "Bandpass sampling for software radio receivers, and the effect of oversampling on aperture jitter," in *IEEE 55th Vehicular Technology Conference*, vol. 4, 2002, pp. 1901 – 1905.

[159] P. Moreira and I. Darwazeh, "An FPGA implementation of the Distributed RF over White Rabbit," in *2013 Joint European Frequency and Time Forum International Frequency Control Symposium (EFTF/IFC)*, July 2013.

[160] M. Lipinski, T. Wlostowski, J. Serrano, P. Alvarez, J. G. Cobas, A. Rubini, and P. Moreira, "Performance results of the first White Rabbit installation for CNGS time transfer," in *IEEE Symposium on Precision Clock Synchronization for Measurement Control and Communication (ISPCS)*, September 2012, pp. 1–6.

[161] P. Moreira, P. Alvarez-Sanchez, J. Serrano, I. Darwazeh, and T. Wlostowski, "Digital dual mixer time difference for sub-nanosecond time synchronization in Ethernet," *IEEE International Frequency Control Symposium*, pp. 449–453, June 2010.

[162] P. Moreira and I. Darwazeh, "Digital femtosecond time difference circuit for CERN's timing systems," in *London Communication Symposium*, September 2011.

[163] D. M. Kolotouros, S. Baron, M. B. Marin, C. Fountas, C. Soos, J. Troska, F. Vasey, and P. Vichoudis, "Metrics and methods for TTC-PON system characterization," *Journal of Instrumentation*, vol. 9, no. 01, p. C01015, 2014. [Online]. Available: http://stacks.iop.org/1748-0221/9/i=01/a=C01015