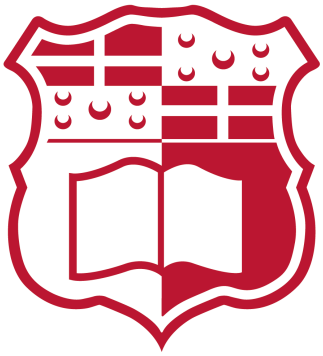
A Remotely Configurable Delay Generator for the HMPID L0 Trigger

Jordan Lee Gauci



Department of Microelectronics and Nanoelectronics

University of Malta

September 2019

Submitted in partial fulfilment of the requirements for the degree of

Doctor of Philosophy (Microelectronics and Nanoelectronics)

The research work disclosed in this publication is partially funded by the
Endeavour Scholarship Scheme (Malta). Scholarships are part-financed
by the European Union - European Social Fund (ESF) -
Operational Programme II – Cohesion Policy 2014-2020
"*Investing in human capital to create more opportunities and promote the well-being of society*".

European Union – European Structural and Investment Funds
Operational Programme II – Cohesion Policy 2014-2020
"Investing in human capital to create more opportunities and promote the well-being of society"
Scholarships are part-financed by the European Union - European Social Funds (ESF)
Co-financing rate: 80% EU Funds;20% National Funds

Dedicated to Luana

COPYRIGHT NOTICE

UNIVERSITY OF MALTA

FACULTY OF INFORMATION AND COMMUNICATION TECHNOLOGY

SUBMISSION OF DISSERTATION FOR EXAMINATION

Student I.D. **582091M**

Student Name **Jordan Lee Gauci**

Course **Doctor of Philosophy**

Title of work submitted
A Remotely Configurable Delay Generator for the HMPID L0 Trigger

I am hereby submitting my dissertation for examination by the Board of Examiners.

_____ **September 2019**

Signature of Student *Date*

Submission noted.

Prof. Ing. Edward Gatt _____

Principal Supervisor *Signature*

September 2019

Date

UNIVERSITY OF MALTA

FACULTY OF INFORMATION AND COMMUNICATION TECHNOLOGY

DECLARATIONS BY POSTGRADUATE STUDENTS

Student I.D. **582091M**

Student Name **Jordan Lee Gauci**

Course **Doctor of Philosophy**

Title of work submitted **A Remotely Configurable Delay Generator for the HMPID L0 Trigger**

(a) Authenticity of Dissertation

I hereby declare that I am the legitimate author of this thesis and that it is my original work.

No portion of this work has been submitted in support of an application for another degree or qualification of this or any other university or institution of higher education,

I hold the University of Malta harmless against any third party claims with regard to copyright violation, breach of confidentiality, defamation and any other third party right infringement.

(b) Research Code of Practice

As a Ph.D. student, as per Regulation 49 of the Doctor of Philosophy Regulations, I accept that my thesis be made publicly available on the University of Malta Institutional Repository.

_____ **September 2019**

Signature of Student *Date*

FACULTY OF INFORMATION AND COMMUNICATION TECHNOLOGY

DECLARATION

Plagiarism is defined as "the unacknowledged use, as one's own work, of work of another person, whether or not such work has been published" (Regulations Governing Conduct at Examinations, 1997, Regulation 1 (viii), University of Malta).

I, the undersigned, declare that this thesis submitted is my work, except where acknowledged and referenced.

I understand that the penalties for making a false declaration may include, but are not limited to, loss of marks; cancellation of examination results; enforced suspension of studies; or expulsion from the degree program.

Work submitted without this signed declaration will not be corrected, and will be given zero marks.

Jordan Lee Gauci _____

Student Name *Signature*

A Remotely Configurable Delay Generator for the HMPID L0 Trigger

Title of work submitted

September 2019

Date

ABSTRACT

This work deals with the design, implementation, and testing of a remotely configurable delay generator which will be used in the High Momentum Particle Identification Detector (HMPID) as part of the LHC-wide upgrade in preparation for Run3 (2021 - 2023). While during the previous two runs (2010 - 2018) HMPID worked with three trigger levels, this will no longer be the case as the triggering mechanism will change. Although HMPID was triggered by the Level-Zero (L0) trigger, which arrives at 1.2 µs after an event, this will no longer be the case as the L0 trigger will arrive later. Instead, the Level-Minus (LM) trigger will be used which arrives at 800 ns after an event. As such, a delay generator is required such that the trigger signal is delayed by at least 400 ns so that the detector is triggered at the peak of the charges on the pads. The work undertaken and described in this dissertation is therefore classified into three parts.

The first part relates to the design, implementation, analysis, and testing of a digital delay generator that is based on a shift-register architecture. This delay generator has been implemented on a Xilinx Virtex-5 FPGA and can achieve a delay range of 525 ns, with a resolution of 1 ns.

The second part focuses on the design, analysis, optimisation, and testing of a novel delay generator that is based on a delay locked loop (DLL) architecture that is calibrated in closed loop, and then used in open-loop configuration. While also describing the proposed architecture, this part focuses on the optimisations performed on a symmetric rail-to-rail delay element to maximise linearity. The delay generator was implemented in the X-FAB XH018 technology, and has a range varying from 935 ns to 183 ns with a resolution of 2 ns.

Finally, the last part of this work focuses on the firmware upgrades that

had to be performed in the detector such that it can handle the new triggering scheme. Through the use of a new firmware architecture, the readout rate of the firmware has been increased from a maximum of 4.57 kHz to 14.4 kHz, without sacrificing any data integrity.

ACKNOWLEDGEMENTS

These four years would not have been possible without the encouragement and support of a number of individuals. First and foremost, I would like to thank my supervisors Prof. Ing. Edward Gatt, and Dr Ing. Owen Casha. Without the guidance and invaluable comments of these two people, this work would not have been possible.

I would also like to thank Dr Giacinto De Cataldo, who was my supervisor and mentor while I was working at CERN on the HMPID. His expertise on the inner workings of HMPID was really helpful in understanding the physics at work.

I would like to express my gratitude towards my friend Ing. Francarl Galea for his constant support and made this journey an enjoyable one.

Special thanks go to my family, in particular my parents, for their encouragement and backing.

Most of all, I would like to thank my girlfriend, Luana, to whom this thesis is dedicated, for her love, encouragement, and patience while I was going through some particularly difficult times.

This work was made possible through a collaboration between the department of Microelectronics and Nanoelectronics at the University of Malta and the ALICE HMPID at the Conseil Europeen pour la Recherche Nucleaire (CERN).

LIST OF PUBLICATIONS

Sections of the work presented in this thesis were published in five peer reviewed conference proceedings papers and one journal paper:

1. J. L. Gauci, E. Gatt, G. De Cataldo, O. Casha, and I. Grech, "An Analytical Model of the Delay Generator for the Triggering of Particle Detectors at CERN LHC," in *CAS (NGCAS), 2017 New Generation of,* pp. 69–72, IEEE, 2017.

2. J. L. Gauci, E. Gatt, O. Casha, G. De Cataldo, and I. Grech, "On the Design of a Linear Delay Element for the Triggering Module at CERN LHC," in *14th IEEE Conference on PhD Research in Microelectronics and Electronics (PRIME),* 2018.

3. J. L. Gauci, E. Gatt, O. Casha, G. De Cataldo, I. Grech, and J. Micallef, "Design of a Quasi-Linear Rail-to-Rail Delay Element with an Extended Programmable Range," in *2018 25th IEEE International Conference on Electronics, Circuits and Systems (ICECS),* pp. 265–268, IEEE, 2018.

4. J. Gauci, E. Gatt, O. Casha, ALICE Collaboration, et al., "Preparing the ALICE-HMPID RICH for the high-luminosity LHC period 2021–2023, *"Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, 2019.

5. J. L. Gauci, E. Gatt, O. Casha, G. De Cataldo, and I. Grech, "Particle Swarm Optimization of a Rail-to-Rail Delay Element for Maximum Linearity," in *2019 International Conference on Control, Decision and Information Technologies (CoDIT),* pp. 420-424, IEEE, 2019.

6. J. L. Gauci, E. Gatt, O. Casha, G. De Cataldo, I. Grech, and J. Micallef, "A novel delay generator for the triggering of particle detectors at the CERN LHC," in *2019 15th Conference on PhD Research in Microelectronics and Electronics (PRIME)*, pp. 33–36, IEEE, 2019.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF SOURCE CODES

LIST OF ACRONYMS

ADC Analogue-to-Digital Converter

ALICE A Large Ion Collider Experiment

ASIC Application Specific Integrated Circuit

BCID Bunch Crossing Identification

CERN Conseil Européen pour la Recherche Nucléaire

CPV Charge Particle Veto

CTP Central Trigger Processor

DAC Digital-to-Analogue Converter

DAQ Data Acquisition

DDL Detector Data Link

DLL Delay Locked Loop

DNL Differential Non-Linearity

FEE Front End Electronics

HMPID High Momentum Particle Identification Detector

L0 Level-Zero

L1 Level-One

LHC Large Hadron Collider

LM Level-Minus

LTU Local Trigger Unit

LUT Look-up Table

LVDS Low-Voltage Differential Signalling

MSE Mean-Square Error

MWPC Multiwire Proportional Chambers

PCB Printed Circuit Board

PSO Particle Swarm Optimisation

PVT Power, Voltage, and Temperature

QGP Quark Gluon Plasma

RCB Readout Control Board

RDH Raw Data Header

RICH Ring Imaging Cherenkov

SFDR Spurious-Free Dynamic Range

SIU Source Interface Unit

SNDR Signal-to-Noise-and-Distortion Ratio

SPI Serial-Peripheral Interface

TTC Timing, Trigger, and Control

TTCRx Timing, Trigger, and Control Receiver

TTYPE Trigger Type

VCDL Voltage Controlled Delay Line

1. Introduction

The Conseil Européen pour la Recherche Nucléaire (CERN), founded in 1954 and located on the Franco-Swiss border, is a large organisation made up of twenty three member states. Its purpose is to study the fundamental building blocks of matter. CERN's flagship project is the Large Hadron Collider (LHC) (refer to Figure 1.1), which is the largest scientific instrument ever constructed. The LHC is a sophisticated circular particle accelerator and collider, with a circumference of almost 27 km. Two beams are circulated in opposite directions, and are collided together at various points, known as experiments. One of these experiments is referred to as A Large Ion Collider Experiment (ALICE).

ALICE [1] is a general-purpose heavy-ion detector at the CERN LHC dedicated to the study and characterisation of Quark Gluon Plasma (QGP) at extremely high energy density and temperature conditions. For this reason, ALICE consists of a number of detectors dedicated to performing particle identification in a wide momentum range from hundreds of MeV/c to 100 GeV/c through the combination of different detector techniques and technologies [2].

The High Momentum Particle Identification Detector (HMPID) (Figure 1.2) is one such detector [4]. The detector performs charged particle track-by-track identification by measuring the emission angle of Cherenkov photons, in conjunction with the momentum measured by the ALICE tracking detectors, which are the Inner Track-

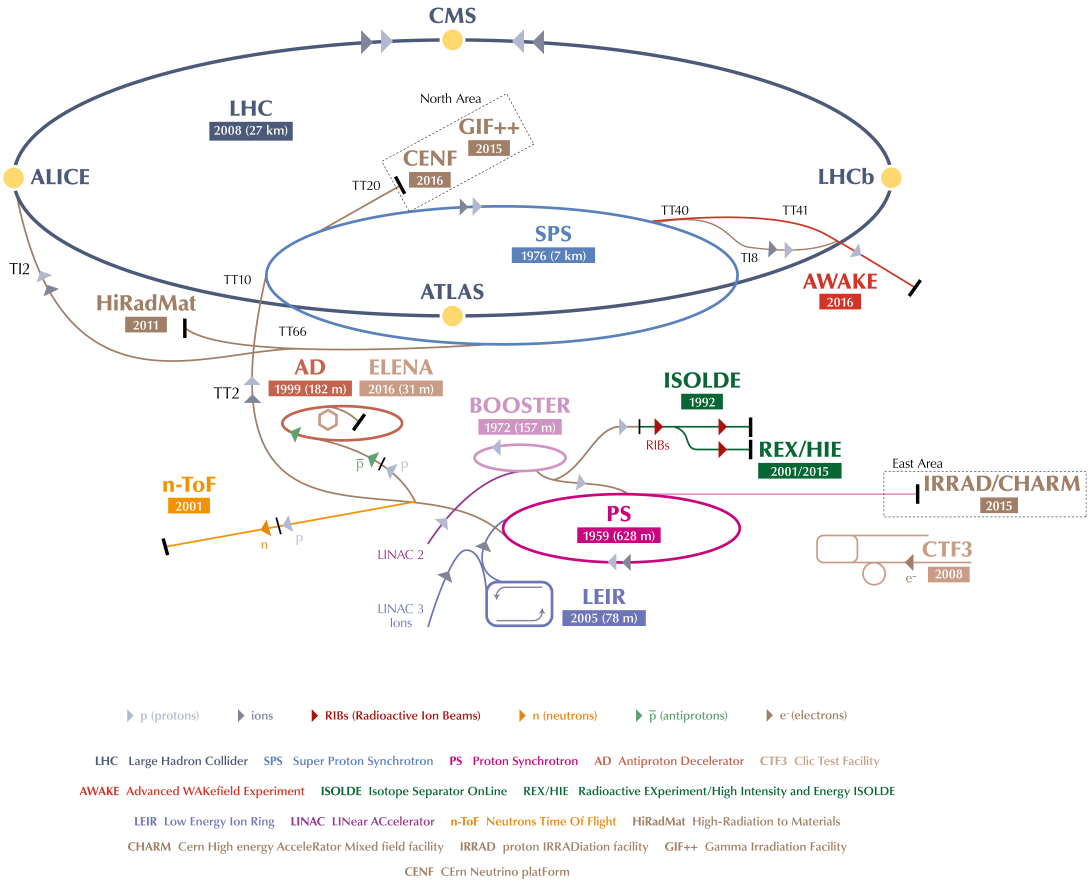


Figure 1.1: The LHC accelerator complex [3]

ing System (ITS) and the Time Projection Chamber (TPC). It provides a three sigma separation particle identification capability in the range of $1-5$ GeV/c, thus extending the useful range for particle identification on a track-by-track bases up to 3 GeV/c for pions and kaons, and up to 5 GeV/c for (anti-)protons, in proton-proton (p-p) collisions.

HMPID is based on seven 1.3×1.3 m^2 proximity focusing Ring Imaging Cherenkov (RICH) counters, split into left and right photocathodes, referred to as RICH0-RICH6, for a total active area of approximately 10.7 m^2. Cherenkov photon detection is achieved by pad segmented photocathodes coated with a 300 nm thick caesium iodide (CsI) layer, installed in Multiwire Proportional Chambers (MWPC) operated with methane (CH_4). Liquid perfluorohexane (C_6F_{14}) is used as the

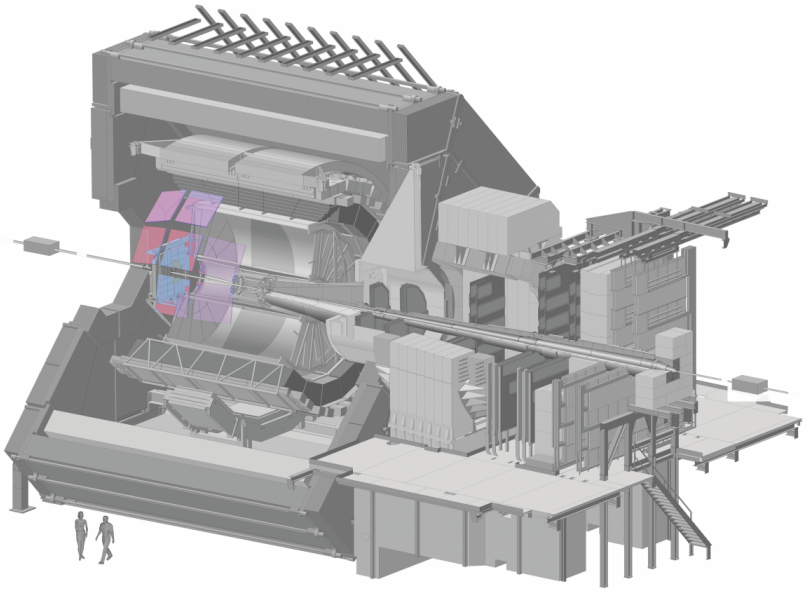


Figure 1.2: The HMPID illustrated within the ALICE experiment

Cherenkov radiator. When an event occurs, charged particles impinge on this gas and are converted into electrons which are read by the charge sense amplifiers. The main physics concepts are illustrated in Figure 1.3, and further details on the performance and stability of the HMPID for the period from 2015 to 2018 are given in [5].

The Front End Electronics (FEE) are based on the GASSIPLEX integrated circuit, which was developed at CERN for the gaseous detectors [6], and is connected to the cathode pads of the MWPC. It consists of sixteen channels, with a charge sensing pre-amplifier having a peaking time of 1.2 µs. A 12-bit Analogue-to-Digital Converter (ADC) digitises the signal stored in the track and hold circuit and the DILOGIC, a dedicated readout chip, starts the transmission of data to the Readout Control Board (RCB) and the Data Acquisition (DAQ) system.

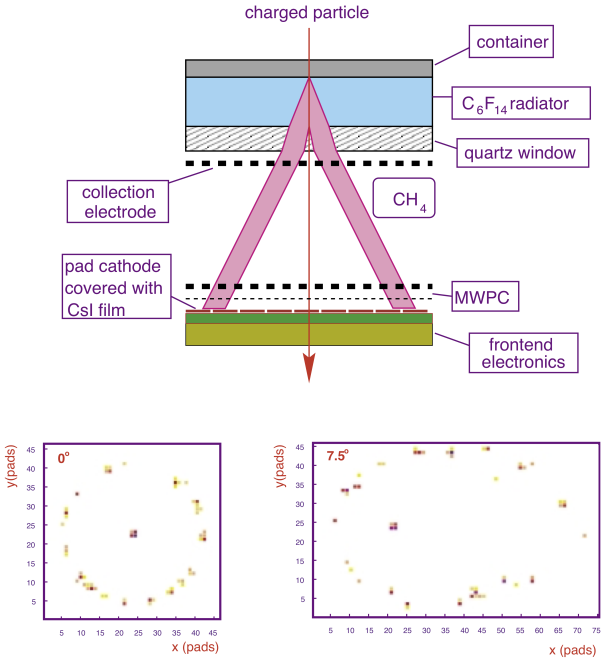


Figure 1.3: Conversion of the charged particles into Cherenkov photons and electrons [4]

1.1 Detector Triggering during Run1 and Run2

The current trigger system for each detector, employed during Run1 and Run2 (2009-2018), consists of the Central Trigger Processor (CTP), the Local Trigger Unit (LTU), and the Timing, Trigger, and Control (TTC) boards. The CTP, shown in Figure 1.4, accepts as its input signals from the triggering detectors, and processes them to generate three level triggers - Level-0 (L0), Level-1 (L1), and Level-2 (L2). The signals are distributed to the detectors using the LTU.

After an interaction occurs, a trigger signal is generated by the respective triggering detectors. Some detectors, such as the HMPID, require a trigger at 1.2 μs, after each interaction. As such, the trigger signal is split into 2 levels:

- **Level 0 (L0):** reaches detectors at 1.2 μs, but is too fast to receive all the trigger inputs;

- **Level 1 (L1):** reaches detectors at 6.5 μs, which picks up all remaining fast inputs.

The third trigger level, L2, arrives after 108.3 μs, and is the so-called "past-future protection." This is required due to the high-multiplicity rates during lead-lead (Pb-Pb) collisions, making events having more than one central collision difficult to construct. The triggering mechanism used during Runs 1 and 2 is shown in Figure 1.5.

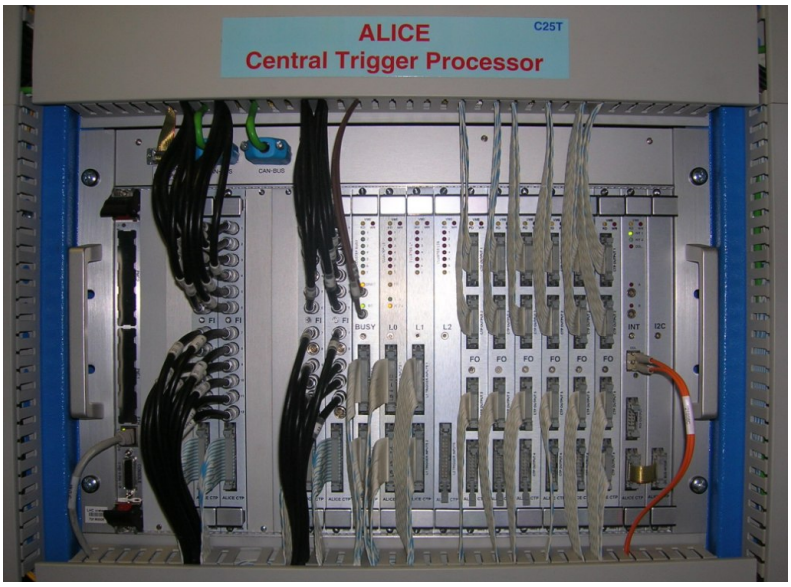


Figure 1.4: The ALICE CTP installed within the ALICE cavern at experiment point 2

The role of the ALICE trigger system is to gather information from the various trigger detectors, and read-out detectors (such the calibration requests and BUSY signal), and to form, in each bunch-crossing (BC) interval, affirmative and non-affirmative trigger decisions for the three different trigger levels. Such decisions are then sent to the TTC system, which distributes them to the front-end electronics of the detectors, where the capturing of data is initiated and readout is controlled. The event identifier, trigger type, and list of participating detectors are also received by the TTC from the CTP. The detector front-end electronics, and DAQ read-out

chain, are synchronised with the CTP through the use of the BUSY signal. This signal is used to indicate to the CTP that the detector is processing an event and therefore cannot accept another event. The L0 trigger is the only signal that is transmitted via a low propagation delay, Low-Voltage Differential Signalling (LVDS) cable. The TTC system is used for the other trigger outputs to the front-end electronics of the detectors, as well as to transmit messages about the event which will allow proper event identification during reconstruction.

Figure 1.5: Timing diagram for the triggering during Run1 and Run2

1.2 Run3 Triggering Modifications

The general approach for the ALICE upgrade is to read out all the Pb-Pb heavy particles at the anticipated interaction rate of 50 kHz, and read-out p-p and p-Pb events at an interaction rate of 200 kHz. The detector electronics, the trigger and the online computing system are designed to keep the nominal performance, even if background noise is larger than anticipated. As such, a new triggering strategy is required, that allows the majority of detectors to read-out the nominal interaction rate (without any dead-time), and provides backward compatibility to detectors such as HMPID which will not be upgrading their electronics for Run3 [7].

For Run3, three different triggers, with different latencies, known as LM, L0, and Level-One (L1) will be used. HMPID will use the LM signal on electrical cables, since the L0 signal takes 1.2 µs to reach the CTP and will therefore arrive too late at the detector. It is estimated, that the L0 signal would arrive in HMPID at 1.5 µs, which would result in a signal loss of 22% [7]. The solution is to utilise

the LM signal which reaches the CTP after 425 ns. Taking into consideration the delays present due to processing by the CTP, CTP-LTU Fan-Out, LTU, and the transmission on 40 m of fast LVDS cables implies that the LM signal will arrive at approximately 800 ns after an event. Since the peaking-time of the charge on the pads is equal to 1.2 µs this implies that there is a need to delay the trigger signal by at least 400 ns. While a fan-in/fan-out module is currently in use to generate a maximum delay of 2.5 µs, this module has a resolution of 25 ns, which is too coarse for use by HMPID during Run3.

In addition to the LM trigger, HMPID will also receive an asynchronous L1 message on Channel B of the Timing, Trigger, and Control Receiver (TTCRx) chip. This message will have a different construction than the one received for Runs 1 and 2. The upgraded triggering mechanism is illustrated in Figure 1.6.

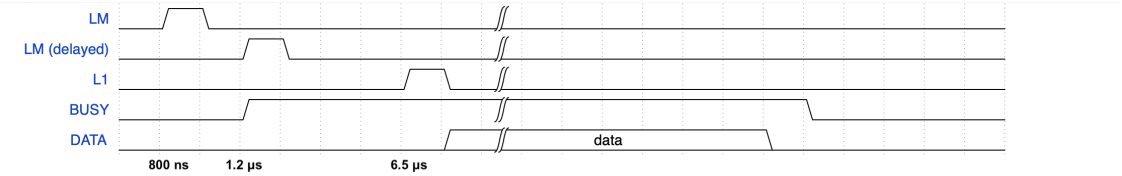


Figure 1.6: Timing diagram for the triggering during Run3

1.3 Main Objectives and Methodology

Thus finding their roots in the above introduction, the main objectives and sought contributions of this research are the following:

1. To design, implement, and test a novel remotely configurable delay generator, which features a delay range of at least 400 ns with a resolution of at least 2 ns, and

2. to upgrade the firmware of the readout electronics in order to handle the new triggering requirements.

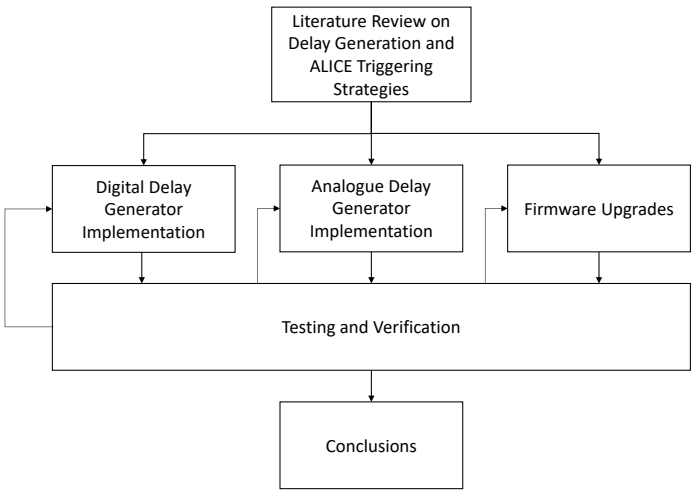


Figure 1.7: The research methodology

Figure 1.7 illustrates the research methodology employed throughout this work. A literature review on the state-of-the-art in delay generation and ALICE triggering strategies has been conducted. Conclusions from this review were then drawn in order to identify the short comings in currently reported delay generators. Two types of delay generators are proposed: one based on a digital implementation of a delay line on an FPGA, and the second based on an analogue implementation of a delay line. The latter involved the design and optimisation of a delay element using various techniques which was later integrated in a delay locked loop architecture. An Application Specific Integrated Circuit (ASIC) was designed in order to verify the performance of the proposed architecture when compared to the post-layout simulations. Finally, firmware upgrades had to be performed to the RCB such that the new triggering mechanisms could be handled. After the implementation of the two types of delay generators and the firmware upgrades, testing and verification was performed, and where needed, modifications to the design were performed, such that the targeted specifications were reached.

1.4 Thesis Organisation

This dissertation is divided into seven chapters which discuss in detail the research goals and the achievements of this PhD. The work presented in this dissertation was published in five peer reviewed conference proceedings papers and a journal paper. This chapter gives an introduction to the requirements of the read-out electronics for the Run3 modifications to be carried out within ALICE. In addition, it presents the motivation of the research described in this dissertation together with its major objectives.

Chapter 2 provides a brief background on the state-of-the-art of delay generation. In particular, it focuses on the design and implementation of both analogue and digital delay elements, while identifying shortcomings and proposes improvements at both architectural and circuit levels. This is followed by Chapter 3, which is a study on the implementation of a purely digital delay generator on an FPGA. While a delay element with the required delay and resolution has been successfully designed and tested, this architecture suffers from the generation of a variable offset, which is not easy to manage and predict. Thus, the fourth chapter describes an analogue implementation of a delay generator architecture, based on a delay locked loop. Significant contributions have been made in optimising the delay element, in order to enhance the linearity as much as possible. This has been done through three different techniques. An ASIC has been developed and its testing methodology is reported in this chapter together with the main results. The penultimate chapter describes the upgrades performed on the firmware of the RCB such that the detector can handle the new trigger scheme during Run 3. Chapter 6 outlines the main conclusions and the potential future directions of this research work.

2. Background and Literature Review

Precise delay generation is an important research area as delay generators can be found in a number of applications ranging from high-energy physics to time-based analogue-to-digital converters. The delay line is at the heart of the delay generator and its delay can be controlled either via analogue or digital means. This chapter presents the necessary background theory and an extensive literature review on the state-of-the-art in delay generation.

2.1 Delay Lines

Precise delay generation is an active research area due to the employment of delay generators in high-energy physics, time-of-flight experiments, time-to-digital and digital-to-time converters [8, 9], clock-data recovery [10], data synchronisation [11], and time interval measurement circuits [12]. The core element of the delay generator is the delay line, which is a device capable of delaying an input signal by a predefined value. This delay can either be fixed or variable. In the case of variable delay lines, the delay can be tuned either by an analogue or a digital mechanism. Delay lines have the following important characteristics which determine their performance [13–15]:

- **Delay Step**: The finest incremental time delay step that can be produced by the delay line;
- **Delay Range**: The maximum time by which a signal can be delayed;
- **Linearity**: The ability to provide equal and uniform delay steps;
- **Monotonicity**: The gradient of the delay transfer characteristic has the same sign over the entire range, that is, the delay increases or decreases in the same direction with an increase in the control voltage or control word;
- **Jitter**: Variation in the delay produced due to noise, which has a direct effect on the smallest delay step that may be generated with high fidelity.

Two main types of delay lines are reported in literature: optical delay lines and electronic delay lines. While optical delay lines achieve high-resolution delay steps (in the sub-picosecond region) together with linear delay increments, they suffer from a limited delay range. The delay range is typically in the order of a few 100 ps. To increase the range, a number of delay lines can be cascaded at the expense of an increase in cost and area [13, 14, 16]. On the other hand, electronic delay lines, typically implemented in a CMOS technology, offer a larger delay range, with reduced cost and complexity. However, two issues need to be taken care of: jitter performance and non-linearity in large delay ranges. The challenge is to design delay lines which provide long delay-ranges with linear and high resolution increments. This is due to the fact that CMOS delay lines cannot be easily cascaded like their optical counterparts, due to an increase in the parasitic effects. This can be mitigated through a monolitic implementation of the circuitry on a single chip [13]. Optical delay lines are unfeasible for the application considered in this work as they are expensive and occupy a lot of area. Thus, this literature review will focus and consider only electronic delay lines.

2.2 Delay Line Architectures

Delay lines can be classified into two architectures: tapped-delay lines and single output delay lines [13]. The tapped-delay line architecture utilises a number of identical delay elements which are connected in series. After each stage, the output is tapped via a switch. The delay elements can either be designed using logic gates or flip-flops limiting the delay resolution to the propagation time of a single delay element, which depends on the CMOS technology used. The delay range is approximately equal to the delay resolution multiplied by the number of stages. This type of architecture has a number of advantages, such as its simple topology leading to efficient integration and the fine resolution that it can achieve [13]. In the single output delay line, a chain of delay elements are cascaded with a single output. In this case, the time delay may be adjusted either by means of an analogue or digital signal, depending on the type of delay element used.

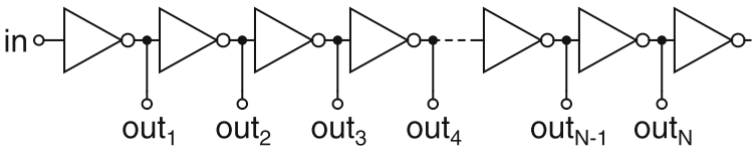


Figure 2.1: Typical inverter chain delay line [17]

2.2.1 Inverter Chain Tapped Output Delay Line

Figure 2.1 shows an inverter chain consisting of N inverters with tapped outputs after each inverter. When a signal is applied at the input, it is produced at out_i with a delay D_i [17]:

$$D_i = i\tau_u \tag{2.1}$$

where i is the output tapping number and τ_u is the propagation delay of one inverter that can be approximated by its drive strength (overdrive voltage). This depends

on the equivalent drive resistance, R_{eq}, and the load capacitance C_L [18].

$$\tau_u = \ln(2) R_{eq} C_L \tag{2.2}$$

The load capacitance consists of the output capacitance of the inverter, the input capacitance of the subsequent stage and the capacitance of the interconnects. The value of R_{eq} depends on the sizing of the transistors, and whether the transistor is NMOS or PMOS. This leads to a different propagation delay for rising and falling edges. The equivalent resistance of a transistor, when used with full-swing signals is given by:

$$R_{eq} \approx \frac{3}{4} \frac{V_{dd}(1 - 7\lambda V_{dd}/9)}{\beta((V_{dd} - V_t)V_{dssat} - V_{dssat}^2/2)} \tag{2.3}$$

where β is the current factor given by:

$$\beta = k' \frac{W}{L} \tag{2.4}$$

and k' is the process transconductance parameter, V_{dssat} is the overdrive voltage, λ is the channel length modulation factor and V_t is the threshold voltage. These are all technology parameters which vary for PMOS and NMOS transistors [18].

The channel length modulation factor λ is proportional to L and V_{DS} is proportional to L. The equivalent resistance, R_{eq}, is therefore inversely proportional to the transistor aspect ratio W/L, where W is the channel width and L is the channel length. The driving inverter and the load inverter are sized equally such that identical delay elements are created. Thus, C_L is approximately proportional to the transistor area. This implies that, from equation (2.2), τ_u is proportional to L^2 and is therefore independent of the channel width. Therefore, the larger the length, the larger the delay. Usually, the minimum length is chosen such that the shortest propagation delay is obtained. Since local process variations can hinder

the operation of the delay element and are directly affected by W, the channel width can be used as an additional degree-of-freedom such that these effects are reduced.

From Figure 2.1, it can be noted that the output after the last inverter is not tapped. This is due to the fact that the last inverter is not loaded, and therefore its load capacitance is smaller. For this reason, it is typically not used and is referred to as a dummy inverter. Its purpose is simply to provide the correct C_L for the previous inverter. While this architecture is simple, it has an obvious disadvantage. The odd outputs are the inverse of the input signal, which may be undesirable. There are two solutions that can be used to solve this, and these are introduced in the following sections.

2.2.2 Non-Inverting Output Delay Line based on Cascaded Inverters

The easiest way to implement a non-inverting delay element is to cascade two inverters which will act a single delay element, as illustrated in Figure 2.2. If both inverters are equally sized, the unit delay will be twice the unit delay, τ_u, which will result in a degradation of the time delay resolution by a factor of 2. If the inverters are not equally sized, the unit delay will increase further. This is due to the fact that the load capacitance will increase proportionally with the width and consequently, by equation (2.2), the delay will also increase.

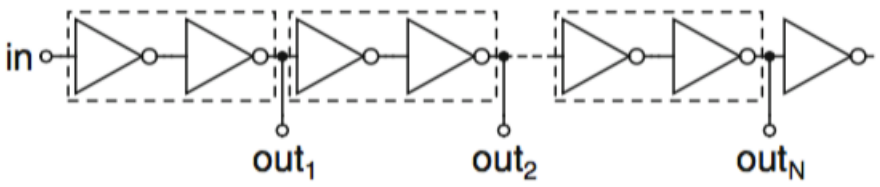


Figure 2.2: Non-inverting tapped delay line [17]

2.2.3 Differential Delay Line

A differential delay line is presented in Figure 2.3, where two parallel inverter chains are used. Cross-coupled inverters are used so as to provide the same non-inverted outputs on the same side of the delay line, while also limiting the delay to almost that of a single inverter. These cross-coupled inverters also serve a different purpose. Due to process variations, the time delay of an inverter may vary, and therefore the time delay of the two independent inverter chains will vary. The cross-coupled inverters thus serve as a latch, forcing both nodes to switch at the same time and will keep the edges in both inverter chains synchronised [17].

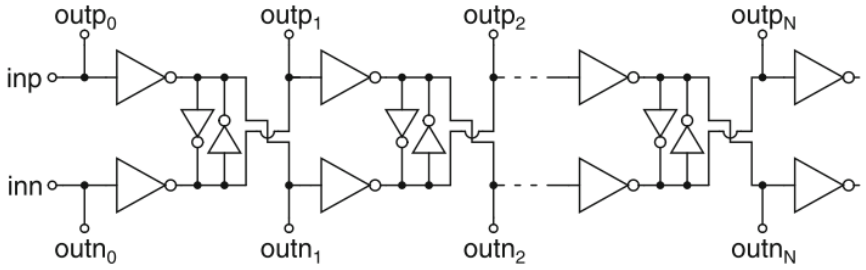


Figure 2.3: Differential delay line architecture [17]

However, the use of cross-coupled inverters leads to an increased capacitative load, which will therefore increase the unit delay, τ_u. Although simulations show that the unit delay is only slightly increased, timing errors due to noise and process variations and an increase in power consumption are undesired effects. Moreover, the sizing of the cross-coupled inverters needs to be taken into consideration. If they are too small, their effect on the edges of the signal slopes would be negligible, whereas if they are too large, the aforementioned effects will dominate and will in turn degrade the performance [17].

2.3 Delay Elements

Delay lines are based on delay elements that can be implemented in various methods, each of which having their respective advantages and disadvantages. The purpose of a delay element is to generate an identical and delayed waveform of an input signal [19]. There are two methods for controlling the delay; either by using a digital word, or through an analogue signal which can be either a voltage or a current. The latter is suitable for sub-picosecond to picosecond delay control. The delay can also be controlled in a digital manner by using discrete voltage or capacitance changes in Digitally Controlled Delay Elements (DCDE). In either case, the design of the delay element is crucial as it can hinder the performance of the circuit [14].

While delay elements can be controlled in an analogue or digital manner, their architecture varies according to whether they use passive delay elements or active delay elements. Passive delay elements utilise passive devices (resistors, inductors, and capacitors) and have numerous advantages over their active counterparts. Such advantages include reduced sensitivity to environmental variations, a linear response, less distortion in the output signal, while also having a higher bandwidth and better accuracy [13].

Active delay elements, on the other hand, utilise components such as diodes and transistors. These types of devices can be programmable and can achieve finer delay steps than passive delay elements. Such delay elements can be further split into two categories; coarse delay elements, which are useful in generating fixed, quantised and long delays, and fine delay elements which are able to produce small and precise delay steps through an analogue control voltage or current [13].

Figure 2.4 presents a taxonomy diagram of the main delay element techniques reported in literature and that will be discussed in the following sections. Delay elements can be split into two broad groups: analogue signal controlled and digital

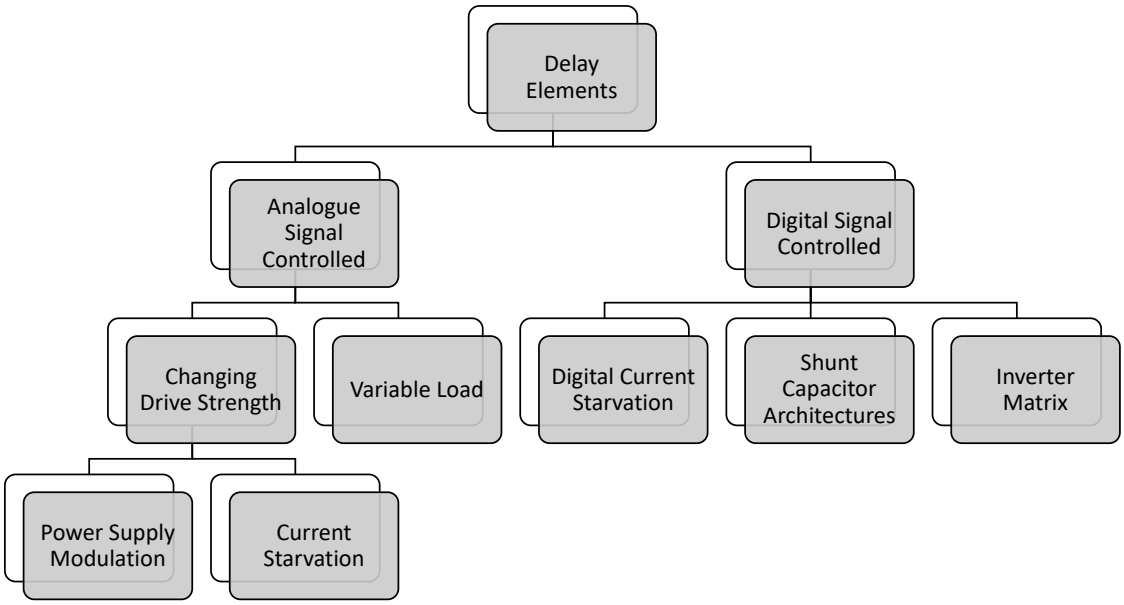


Figure 2.4: Taxonomy diagram of the main delay elements

signal controlled. Each category can be further split according to the different methodology employed.

2.3.1 Analogue Signal Controlled Delay Elements

The delay of a CMOS delay element depends on the resistance and the capacitance time constant. Thus, by controlling the effective on resistance or effective capacitance, the delay can be tuned. Logic-gate based delay elements can be tuned through two strategies; either by changing the drive strength of the logic gate that is driving a load capacitance and input of another logic gate, or through the addition of a variable load connected between two successive logic gates. There are two methods for changing the drive strength, namely via the modulation of the power supply voltage, and through current-starving [13, 17, 20].

2.3.1.1 Power Supply Modulation

By modulating the power supply of a delay element, the delay may be changed. For example, by increasing the operating voltage, the transistors will conduct more

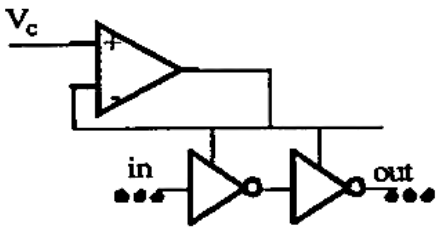


Figure 2.5: Delay element utilising supply modulation [21]

current and therefore a load capacitance is charged or discharged faster which in turn leads to a shorter delay. One advantage of this method is that an increase in supply voltage will lead to steeper edges which will reduce sensitivity to Power, Voltage, and Temperature (PVT) variations [17].

An architecture that regulates the supply voltage has been presented in [21] and is illustrated in Figure 2.5. In this architecture, a control voltage, V_c, is used so as to regulate the supply voltage provided to the inverters. By changing the supply voltage, the effective resistance is changed and therefore the current is adjusted, leading to a change in the delay [13, 21]. This architecture, however, has two disadvantages. The first disadvantage relates to the use of the adjustable voltage source, which would require that the source provides a significant amount of current [17]. The second disadvantage is that a high delay resolution cannot be achieved using this technique [13, 17, 21, 22].

2.3.1.2 Current Starving Techniques

Current starving is another method of changing the drive strength, thereby modulating the delay [14, 17]. Figure 2.6 shows a voltage controlled delay element, implemented as a two-stage inverter. The main inverter (M_1 and M_2) is biased through M_3 and M_4. A control voltage, V_c, is applied and this changes the biasing current of the inverter accordingly. The main disadvantage with this architecture, however, is that although the delay range achieved is large, the delay is highly non-linear as it is inversely proportional to the square of the control voltage, V_c.

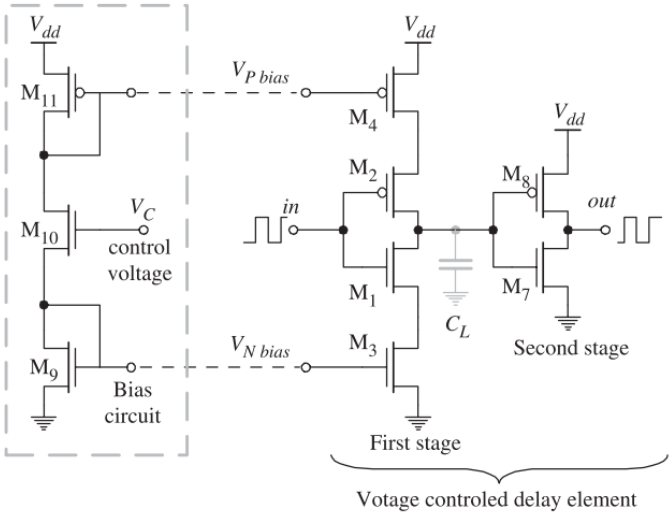


Figure 2.6: Current starved inverter architecture [23]

In fact, the delay can be defined as [23]:

$$t_p = \frac{C_L V_{dd}}{K'_p \frac{W}{L}(V_c - V_t)^2} \tag{2.5}$$

where C_L is the load capacitance, V_{dd}, V_t are the supply voltage and threshold voltage, respectively, K'_p is the technology transconductance parameter, and $\frac{W}{L}$ is the aspect ratio.

A non-linear delay leads to a reduction in the quality of the performance of the variable delay line [24]. Nonetheless, the response may be linearised by utilising a symmetric load configuration [23]. Other methods of the current-starving method have been proposed in literature. Such methods include the use of a transmission-gate element connected to the output of a logic gate [25], using an RC-based differentiator to drive the PMOS transistor of a CMOS inverter [26], and a neuron-MOS mechanism using an electrically floating gate electrode [27].

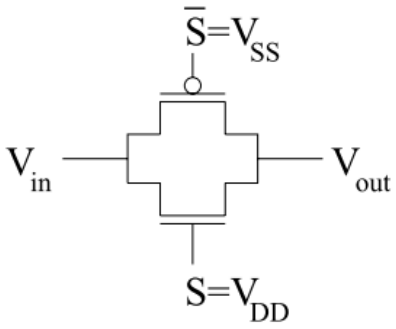


Figure 2.7: Transmission gate [25]

2.3.1.3 Transmission Gate Based Delay Elements

A parallel connection between a PMOS and an NMOS transistor, controlled by complementary signals, make up a transmission gate. The transmission gate is presented in Figure 2.7 [25]. Keeping the transmission gate always on, will result in a delay element. The propagation delay, t_p, of the transmission gate is dependent on the charging or discharging of C_L, through R_{eq}. The equation for the delay has the general form of equation (2.2) and is given by equation (2.6) [25].

$$t_p = \ln(2)\frac{2V_{dd}}{K'_n(\frac{W}{L})_n(V_{DD} - V_{TN})^2 + K'_p(\frac{W}{L})_p(V_{DD} - |V_{TP}|)^2}C_L \tag{2.6}$$

The delay may be increased by increasing the length of the transistors, which will therefore linearly increase the equivalent resistance, R_{eq}, and it may be decreased by increasing the width, thus decreasing R_{eq}. This however is limited by the diffusion (junction) capacitance as by reducing the width, this parameter increases, which will lead to an increase of the load capacitance [19, 25].

Transmission gates can be cascaded to increase the propagation delay, however it was shown in [28] that the delay increases quadratically with the number of transmission gates in the chain. The delay of n cascaded transmission gates may be approximated by the Elmore approximation [29] and will result in equation 2.7.

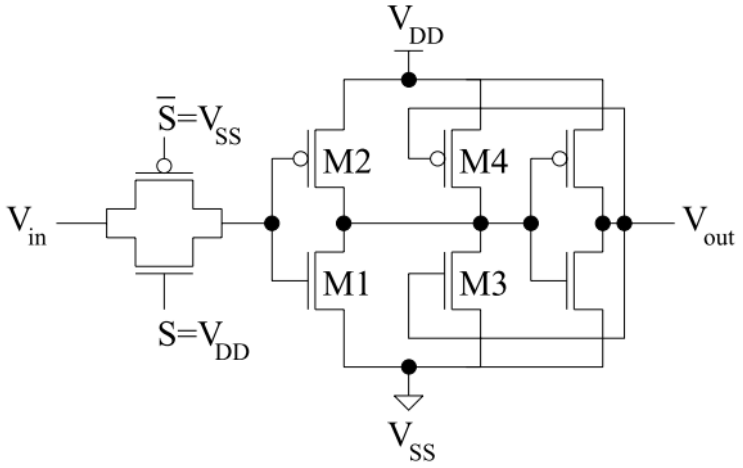


Figure 2.8: Transmission gate in cascade with a schmitt trigger [25]

$$t_{p_{\text{cascade}}} = \ln(2) R_{eq} C_L \frac{n(n+1)}{2} \tag{2.7}$$

While the transmission gate is advantageous with respect to power and area consumption it suffers in maintaining signal integrity, the time it takes for the output tor transition between 10% and 90% of V_{DD}. The problem becomes further prevalent when cascading transmission gates as the signal integrity deteriorates quadratically [19, 25].

Since the transmission gate suffers from poor signal integrity, it may be cascaded with a circuit called a Schmitt Trigger. The purpose of this circuit is to generate a fast, clean signal output from a noisy or slow varying input signal. This circuit not only aids in noise suppression, but the steep output helps in maintaining low power consumption [19, 25]. While the delay of the element may be changed by altering the W/L ratio of the transmission gate, in this case the delay may also be altered by changing the switching thresholds of the Schmitt Trigger [19]. The signal integrity of this type of delay element is good due to the positive feedback in the circuit, and it is unaffected by the delay value. The architecture is illustrated in Figure 2.8.

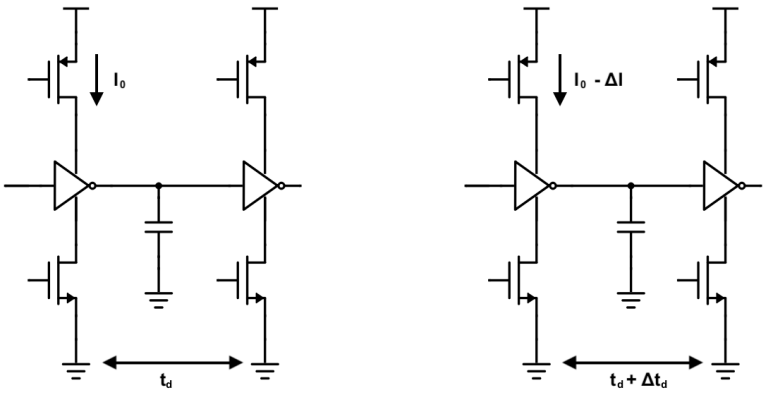


Figure 2.9: Current-controlled delay element (Adapted from [30])

2.3.1.4 Current Controlled Delay Elements

Figure 2.9 illustrates a simple current-controlled delay element, which exhibits non-linear delay characteristics. The delay time (t_{d0}) of this delay element can be expressed as [30]:

$$t_{d0} = \frac{C_L V_t}{I_0} \tag{2.8}$$

where C_L is the load capacitance, V_t is the threshold voltage, and I_0 is the control current. If the control current is decreased by ΔI, the time delay should increase by a corresponding Δt_d, as expressed through equation 2.9.

$$t_{d0} + \Delta t_d = \frac{C_L V_T}{I_0 - \Delta I} \tag{2.9}$$

Combining equations 2.8 and 2.9 yields:

$$\frac{I_0 - \Delta I}{I_0} = \frac{t_{d0}}{t_{d0} + \Delta t_d} \tag{2.10}$$

It can be seen from equation 2.10 that the controlling current has a direct effect on the delay produced as it affects how fast the output capacitance is charged and discharged [13, 30].

2.3.1.5 Inverter Based Delay Elements

It was previously seen that two inverters can be cascaded to create a delay element. The propagation delay of this type of architecture is dependent on the time it takes to charge or discharge its load capacitance, which implies that the delay will not be linear. The propagation delay can therefore be approximated by:

$$t_p = \frac{1}{2}(t_{p_{LH}} + t_{p_{HL}}) \tag{2.11}$$

where $t_{p_{LH}}$ and $t_{p_{HL}}$ are the time it takes for a transition from low to high, and high to low, respectively. However, this equation is not valid for devices with small geometrical dimensions, when $V_{DD} >> V_T$ and velocity saturation effects are taken into account, as the average current flowing through the transistor would be proportional to V_{DD} not V_{DD}^2. The delay can then be approximated by [25, 28]:

$$t_p \approx \frac{C_L}{2}\left(\frac{1}{K_p} + \frac{1}{K_n}\right) \tag{2.12}$$

The advantage of this type of delay element is the generally small power consumption. The power consumption depends on three factors; the static power consumption due to leakage current (P_{stat}), the dynamic power consumption due to the charging and discharging of the load capacitance (P_{cap}), and the dynamic power consumption due to the non-zero rise and fall times which result in a short-circuit (P_{sc}).

Therefore, the total power consumption can be represented by:

$$P_{tot} = P_{stat} + P_{cap} + P_{sc} \tag{2.13}$$

$$= I_{leak}V_{DD} + C_L V_{DD}^2 f + \frac{t_r + t_f}{2} V_{DD} Ipeak f \tag{2.14}$$

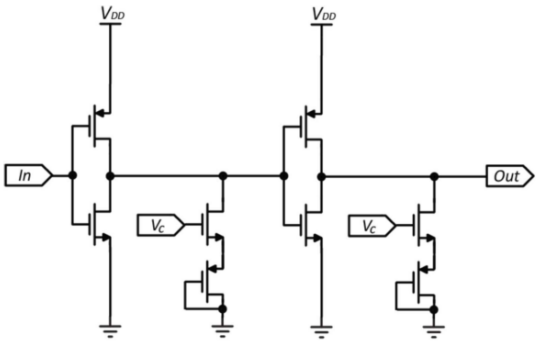


Figure 2.10: Diode-connected transistors used as a delay element [22]

where I_{leak} is the leakage current, f is the switching frequency, and I_{peak} is the peak current.

The small power consumption comes at the cost of area, where four transistors are required for each delay element, excluding any extra transistors which can be added to the pull-up and pull-down networks [25].

2.3.1.6 Diode Connected Transistors

In [22] the authors propose a delay element that uses diode-connected transistors and the architecture is shown in Figure 2.10. It can be seen that the two inverters are loaded with a diode-connected PMOS transistor such that a variable propagation delay can be obtained, through the control of the charging/discharging current through the MOS diode, by V_c. This type of architecture allows for a very linear delay-voltage characteristic [22].

2.3.1.7 Thyristor-Based Delay Elements

Another architecture which can be used is based on Thyristor Delay element [25,31], shown in Figure 2.11. This architecture has a number of advantages, particularly the good robustness against environmental variations, due to the fact that this delay element is current-controlled. A long delay range can be achieved using this type of delay element. The delay range of this thyristor delay element depends

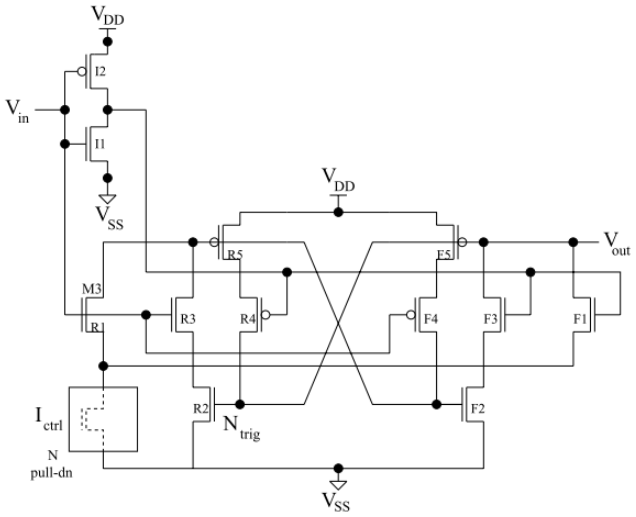


Figure 2.11: Thyristor based delay-element [25]

on the N-pull down network, which is the controlled current source. In 0.8 μm technology, the delay range achieved was from 2.6 ns to 76.3 ms. This architecture also has the advantage of having very good signal integrity, due to the positive feedback, which in turn leads to very small power consumption. The disadvantage with this system is the relatively large area requirement [25].

2.3.2 Digitally Controlled Delay Elements

Digitally Controlled Delay Elements (DCDEs) are based upon logic gates. There exist four popular techniques for designing a variable, digitally controlled delay element: The shunt-capacitor technique, the current starved technique, the inverter matrix, and the variable resistor technique (also called the Differential Delay Cell) [13, 32]. Also known as Programmable Delay Elements, DCDEs, are utilised in various applications such as Delay Locked Loops (DLLs), Phase Locked Loops (PLLs), Controlled Oscillators, and Analogue-to-Digital Converters. Programmable Delay Elements can be of two types: Partially Programmable, where only one edge of the signal can be controlled, and Fully Programmable, where both edges of the input signal can be manipulated [33].

2.3.2.1 Digitally Controlled Current Starved Inverter

Figure 2.12 presents a basic current-starved inverter architecture that can be digitally programmed. An input signal, D_{in}, is applied to an inverter that is to be controlled through the input vector, $b_0...b_{N-1}$. The delay control part consists of a discharge network (DN) and a charging network (CN). The inverter at the output is used such that rail-to-rail voltages can be obtained as well as having an exact replica, albeit a delayed version, of the input signal. The transistors in the charge/discharge networks are sized in a binary weighted fashion, such that binary incremental delays can be achieved [14, 33].

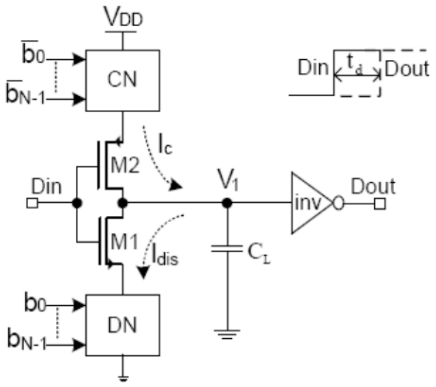


Figure 2.12: Digitally controlled current starved inverter architecture [33]

Figure 2.13 illustrates a typical implementation of the aforementioned architecture. The charge and discharge transistor networks ($M_{p0}, M_{p1}...$ and $M_{n0}, M_{n1}...$) control the effective resistance of the sources of M_1 and M_2. On the rising edge of the input signal, the parasitic drain capacitance of M1, C_1, is discharged according to the current through M_1, which is dependant on the resistance connected to the source of this transistor. The opposite occurs when a falling edge is present on D_{in}. In this case, the parasitic drain capacitance is charged towards V_{DD} [33].

While this structure is simple, the delay achieved is also dependant on the parasitic capacitance, C_P, present on the sources of M_1 and M_2. On the rising edge of the input voltage, charge sharing of the parasitic capacitances at the drain and source

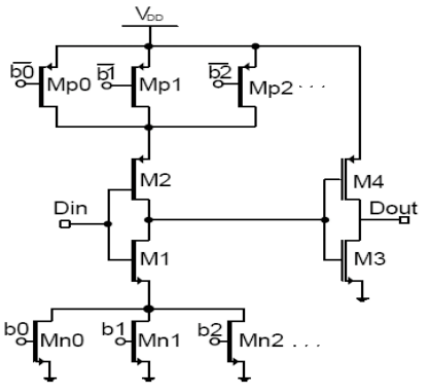


Figure 2.13: Implementation of a digitally controlled current starved inverter architecture [33]

terminals of M_1 occur. When the input signal is low, the capacitance at the source terminal of M_1 discharges to ground [33].

The final voltage, V_{CF}, on the parasitic capacitance, C_1, can be derived from the principle of charge conservation and is given by:

$$V_{CF} = \frac{C_1}{C_1 + C_P} V_{DD} \tag{2.15}$$

Equation 2.15 assumes that C_1 is initially charged to V_{DD}. C_P, on the other hand, varies according to the input combinations of the control transistors, which will in turn lead to a non-monotonic delay [14, 33].

Through the architecture described in Figure 2.13, it can be seen that the input vector is effectively changing the resistance of the PMOS and NMOS transistors of the first inverter. Not only that, but the parasitic capacitance at these nodes also changes as the transistors are switched on and off.

The problem with non-monotonic delay behaviour may be described further through Figure 2.14, which shows a digitally controlled current starved inverter delay element and its response to an input signal. The transistors M_{N_0} and M_{N_1} serve as delay controllers, having different W/L ratios and are placed at the source of M_1.

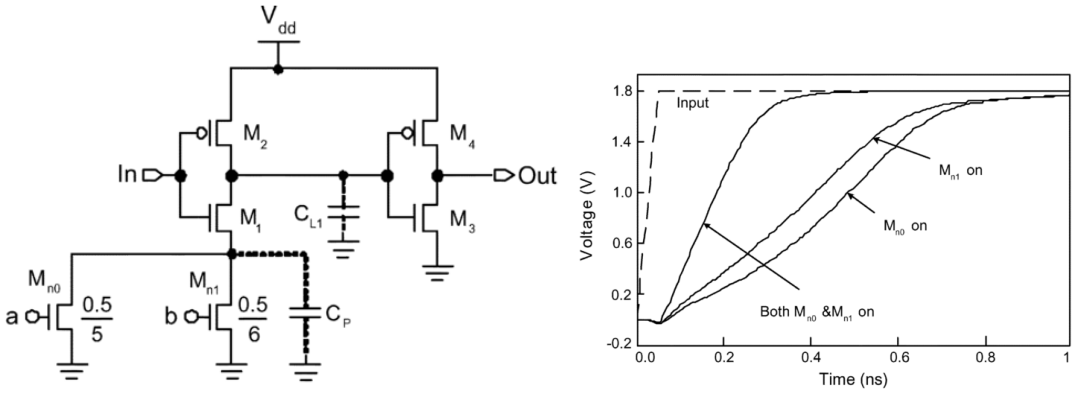


Figure 2.14: Delay behaviour of a current-starved inverter [14]

The circuit can have a total of three different delays, since at least one of the control transistors needs to be turned on. From the transient response of the circuit, it can be seen that the delay between the input and the output is larger when M_{N_0} is on then when M_{N_1} is on. Even though the W/L ratio of M_{N_0} is larger, the delay of the waveform is not lower as expected due to the lower equivalent resistance. This is attributed to the parasitic capacitance, C_p, at the source of M_1. When transistor M_1 is switched on, the load capacitance C_L charge shares with C_p, therefore the delay is smaller than expected as C_L is initially discharging through C_p [14].

2.3.2.2 Shunt Capacitor Architectures

Figure 2.15 shows the basic Shunt Capacitor architecture. In this delay element, a control voltage, V_{ctr}, is used to control the charge-flow through a pass transistor. The transistor is therefore acting as a capacitor. V_{ctr} can be either an analogue signal (coming from a charge pump) or a digital signal. To use the shunt-capacitor architecture in a digital manner, a number of differently sized NMOS transistors, with the drain and source connected together are used and an input vector can be used to control the delay [9]. The equivalent circuit is illustrated in Figure 2.16. Instead of using MOS capacitors, another technique that may be used is to employ transmission gates as switchable capacitors by leaving one node floating, as shown

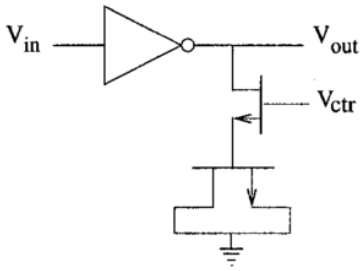


Figure 2.15: Basic shunt capacitor architecture

in [34]. This is useful to lower the power consumption and area.

The time constant, τ_{SCI}, of the shunt-capacitor architecture can be estimated by multiplying the NMOS/CMOS channel resistance, R, with the total capacitance at the inverter output. Thus:

$$t_{delay} \propto \tau_{SCI} \approx RC_{TOT} \tag{2.16}$$

$$= R(C_{inv} + (C_1 b_1 + C_2 b_2 + ... + C_N b_N)) \tag{2.17}$$

C_{inv} is the total parasitic capacitance at the inverter's output. $C_1, ..., C_N$ are the MOS capacitors and $b_1, ..., b_N$ is the input control vector. It can be seen from equation 2.17 that this architecture does not suffer from the non-monotonic behaviour as the current-starved inverter architecture [9]. The shunt-capacitor technique has been seen to deliver a linear response and generates lower jitter when compared to the current-starved inverter architecture [35].

The capacitance of a MOSFET device connected in this manner is given by:

$$C_g = \frac{\varepsilon_{\text{ox}} WL}{t_{\text{ox}}} = C_{\text{ox}} WL \tag{2.18}$$

where $C_g = C_{gb} + C_{gs} + C_{gd}$

From equation 2.18, it can be seen that the capacitance is determined by the WL

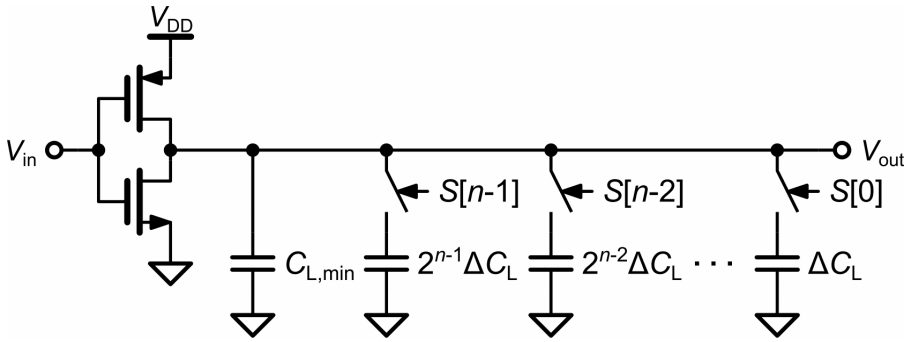


Figure 2.16: Typical implementation of a digitally controlled shunt capacitor inverter delay element [34]

product. The delay, however, is not only determined by this product but is also dependent on the parasitic effects at the drain and source regions. To achieve a linear delay, a unit size MOS capacitor is used, and instead of increasing the WL product to achieve binary delays, the number of capacitors is increased. Thus for a 5-bit input vector, the time delay T_d would be equal to:

$$T_d = R_{\text{channel}}(C_p + C(S_1 + 2S_2 + 4S_3 + 8S_4 + 16S_5)) \tag{2.19}$$

2.3.2.3 Inverter Matrix

The inverter matrix is another architecture that can produce small delay steps. The circuit consists of an array of tri-state inverters. The delay of the circuit is adjusted by enabling or disabling the required number of inverters. This method was implemented by Abas et al. [12] in 0.18 µm technology and linear delay steps of 2 ps in the delay region from 84 to 200 ps were achieved, with a total delay range of approximately 400 ps. Figure 2.17 illustrates this architecture.

2.4 Conclusion

This chapter has introduced the fundamental concepts of delay elements, which are essential in the implementation of delay lines. Two types of delay elements were

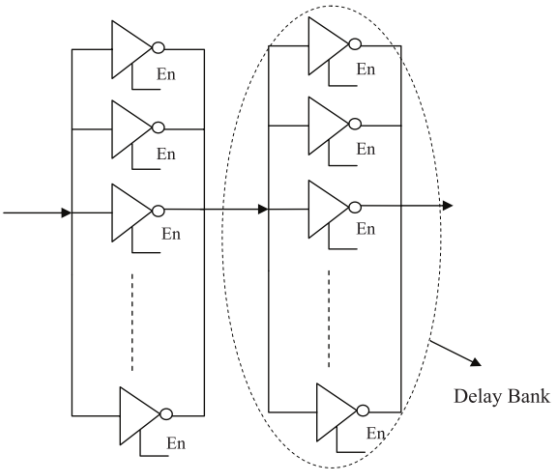


Figure 2.17: Inverter matrix architecture [12]

discussed: analogue signal controlled delay elements, and digitally controlled delay elements. Some analogue controlled delay elements have fixed delays, such as the transmission gate delay elements and the inverter based delay elements. In these cases, the delay depends mainly on the dimensions of such devices. The delay however may be controlled either through power supply modulation, or through the use of additional circuitry. Power supply modulation is not useful for the delay generator architecture being proposed in this work, since a delay resolution of 2 ns is required and cannot be achieved using this technique. On the other hand, in current-starvation architectures the controlling current is modulated to increase or decrease the delay, where the required resolution can be achieved. However, this type of delay element has a non-linear response. Nonetheless, there exist techniques to linearise the delay response [23]. Another method is to add diode connected transistors to the inverter, and this generates very linear delays. Thyristor-based delay elements can also be used, when long delay ranges are required.

Digitally controlled delay elements are another option to generate delays. Such delay elements can also produce high-resolution delay steps, however they suffer more from non-linearity than their analogue counterparts. This is due to the parasitic capacitances of the digitally controlled transistors, which changes with the switch-

ing of the transistors. Techniques such as the shunt capacitor architecture, where the number of capacitors is increased aid to obtain a more linear delay response, however the aforementioned problem is still prevalent.

For this reason, analogue controlled delay elements are better suited when integrated in a system. In particular, they can offer high-resolution delay steps, low jitter, good PVT compensation, and good stability. On the other hand, digitally controlled delay elements are process portable, easier to design, simpler in their design, and may utilise lower power [13].

Delay Element Architecture	**Control Mechanism**	**Linearity**	**Range**	**Area Requirements**
Power Supply Modulation	Analogue (Drive Strength)	NA	NA	Large
Transmission Gate	Analogue	NA	NA	Low
Cascaded Inverter Based	Analogue	Depends	Depends	Low
Current Starved Inverter	Analogue/Digital	Non-Linear	Large	Low
Diode Connected	Analogue	Linear	Low	Low
Thyristor-Based	Analogue (Current Control)	Non-Linear	Large	Large
Shunt-Capacitor	Digital	Non-Linear	Large	Depends on Capacitor Values
Inverter Matrix	Digital	Linear	Low	Large

Table 2.1: Comparison between different delay element architectures

A comparative table is presented in Table 2.1, which shows the different delay element architectures with their respective characteristics. From this table it can be seen that while the current starved inverter architecture is a highly non-linear delay element, it can offer a large range which is desirable. Additionally, a hybrid current-starved shunt-capacitor architecture may be used to further increase the delay range. Thus this work proposes the use of a hybrid current-starved and shunt-capacitor architecture in order to obtain the required delay range. Mathematical optimisation techniques will be used to enhance the linearity of the current-starved inverter, while also obtaining rail-to-rail and symmetric operation. Furthermore, a switched capacitor bank will be used to further increase the delay. In addition, a purely digital approach, based on a shift-register architecture, will be implemented and analysed as a first approach to ensure that the required delay range and resolution are indeed achievable and to provide a benchmark for comparison.

3. Digital Delay Line on an FPGA

This chapter deals with the design, analysis, and testing of a shift-register based delay line, implemented on an FPGA that has a range of 525 ns*, with a resolution of* 1 ns*. This approach has been implemented as a first implementation to ensure that the required delay range and resolution can be obtained. A brief background on the two types of delay lines currently employed at CERN is initially given. This is then followed by the design and implementation of the digital delay line and results from this architecture are presented. A mathematical model is derived, and this is simulated via MATLAB and C++ code. Physical results illustrate that noise generates an unpredictable offset, and solutions are proposed in order to mitigate this error. Parts of this chapter have been published by the author in [15].*

3.1 Background on Delay Lines used at CERN

In this section a review of the electronic delay lines currently used at CERN is given. The remotely programmable Fan-In/Fan-Out module currently used by HMPID is described in the first section while the second section describes a remotely programmable radiation-hardened delay line used by the LHCb calorimeter.

3.1.1 Fan-In/Fan-Out Module currently used by HMPID

HMPID, together with other detectors, currently make use of a fan-in/fan-out module so as to distribute the L0 trigger to the 14 panels that make up HMPID. In addition, the 14 readout control boards (RCBs) of HMPID generate a BUSY signal that is used to provide feedback to the Central Trigger Processor (CTP), and therefore it is the module's responsibility to act as fan-in and distribute one BUSY signal to the CTP.

The module is implemented as a 6U VME64x [36] board, and all logic and control are integrated on an ACTEL PROASIC3E FPGA. The board is shown in Figure 3.1. The device can be configured as either fan-in or fan-out through jumpers on the board itself. 25 LVDS cables can be connected to the board, with typical configurations being 1 input/24 output and 24 input/1 output. Each line can be remotely enabled or disabled. In addition a delay can be introduced on each output. The module is capable of generating a maximum delay of 2.5 µs with a resolution of 25 ns.

3.1.2 Delay Line used by LHCb

High energy physics experiments commonly employ the use of delay lines as synchronisation is critical for such applications. A desired delay is programmed by the user such that a delay can be introduced to compensate for any latency that has been introduced by cables or fibres. This section describes a delay line based on a Delay Locked Loop (DLL), programmed through the Serial-Peripheral Interface (SPI) protocol, and which can be configured to delay the 25 ns LHC Clock in steps of 1 ns [37].

A block diagram of the delay chip that was designed is illustrated in Figure 3.2. On startup, a 1.2 ms wide reset signal is generated along with a glitch suppressor such that any Single Event Transients (SET) glitches, up to 8 ns, are filtered out.

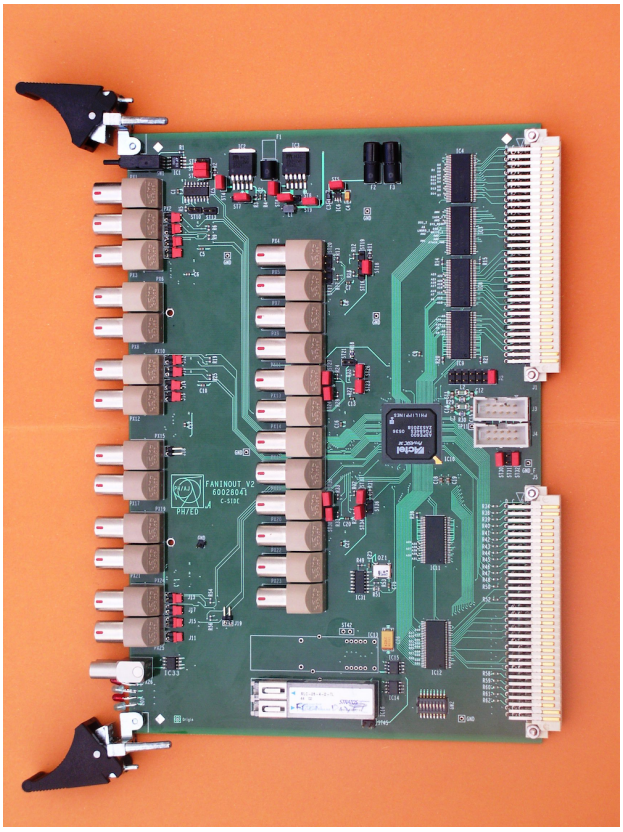


Figure 3.1: Fan-in/fan-out module currently in use by HMPID

The SPI slave interface is used so as to be able to generate the signals required for reading from and writing to the registers of the DLL channels, while also allowing the user to reset the charge pumps through software. Given the amount of radiation present in the area of the chip, the SPI slave machine is also Single Event Upset (SEU) tolerant. Three independent LVDS clock signal outputs are used for each DLL channel, enabling the user to configure the delay in steps of 1 ns, within the range 0 ns and 24 ns. A 16-bit Triple Modular Redundancy (TMR) register is used so as to store the configuration and protect against single event upsets. The DLL is implemented in a fully-differential manner, as this results in lower switching mode noise than their single mode counterparts. This is essential in this case since switching noise affects the analogue shaper used by the detector for readout. For this reason, the ADC Clock Generator is placed far from the analogue shaper.

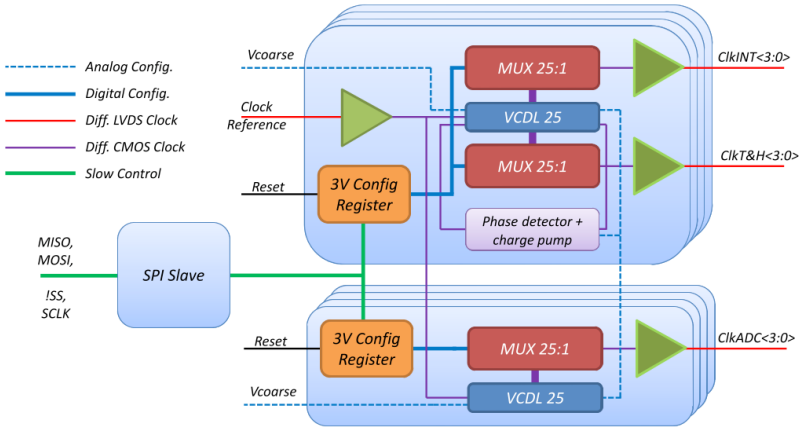


Figure 3.2: Block Diagram of Delay Chip designed for LHCb [37]

The reference differential clock signal is fed through a Voltage Controlled Delay Line (VCDL) and a multiplexer selects the desired output. The last stage involves the use of a LVDS driver to convert the voltage to the required levels. The main DLL block is similar but it also converts the input Clock Reference signal from LVDS to CMOS levels which is then fed to a phase comparator generating the fine control voltage. The external signal *Vcoarse* is a bias voltage and is used in conjunction with *Vcontrol* to ensure that the introduced delay by each VCDL stage is equal to 1 ns.

The Voltage Controlled Delay Line (VCDL) consists of 25 cascaded adjustable Delay Elements (DE). The adjustable DEs are designed as current starved inverters. This type of inverter can apply a delay to the input signal through the input signals *Vcontrol* and *Vcoarse*. Its construction is based on the current starved inverter architecture, where the current passes through the MOS transistors, acting as adjustable resistors, which in turn varies the delay of the output signal. The type of architecture utilised here is advantageous as a large delay range can be achieved while maintaining slew rate symmetry. On the other hand, this architecture requires a larger area, and requires an additional control of the second signal. To minimise the effect of the latter, a voltage *Vcoarse* is applied so as to compensate

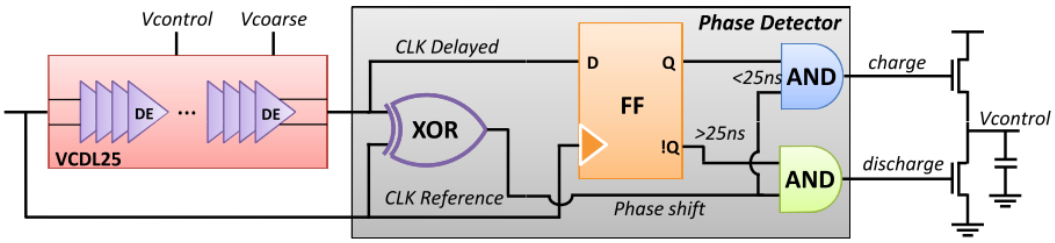


Figure 3.3: Schematic of Phase Detector as implemented and used by the LHCb calorimeter [37]

for any process or environmental variations. The charge pump of each DLL channel will then provide *Vcontrol*, which will result in the PMOS transistor voltage compensating for mismatch, environmental variations and/or noise perturbations.

The phase detector, shown in Figure 3.3, consists of a flip-flop, which detects whether the delayed clock signal is leading or lagging the reference clock, and an XOR function between the delayed and reference clock such that the amplitude of the charge/discharge pulse can be determined.

A 25:1 multiplexer is also implemented in the circuit, which was designed specifically such that small delay errors are reduced. The multiplexer has been implemented using two levels of 5:1 line tristate multiplexers such that the clock signal path is as short as only 2 multiplexers. This will therefore achieve low latency, and therefore environmental variations will not introduce variations in the delay.

This implementation has successfully achieved a delay line, capable of shifting the phase of the LHC clock (25 ns) in steps of 1 ns. However, the addition of process variations means that the delay resolution is not always equal to 1 ns. A measure called Differential Non-Linearity (DNL) is used to measure the difference between the expected delay and the measured delay. The mean of the DNL was found to be equal to 23 ps, which differed from the simulated value of 6 ps. This was attributed to the lack of symmetry in the ADC clock delay lines. When removing the systematic effects from the results, the resulting mean DNL was 13 ps [37].

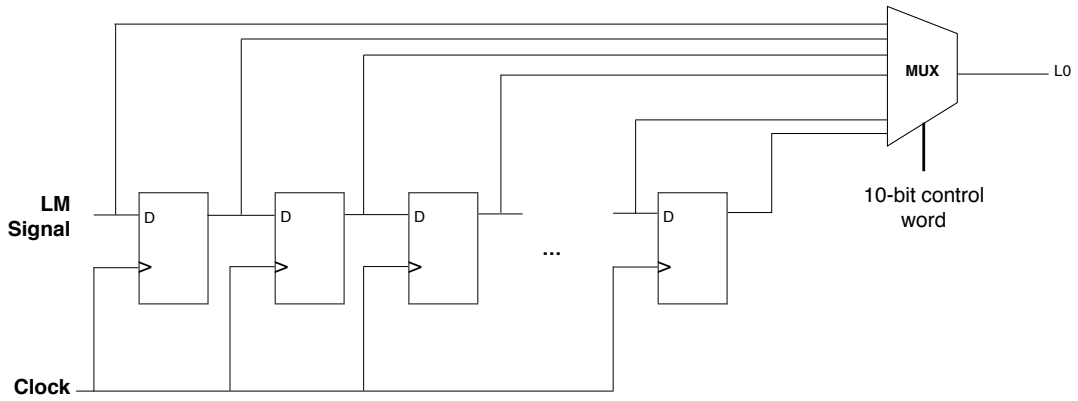


Figure 3.4: Simplified schematic of the tapped shift-register based delay generator

3.2 Digital Implementation of a Delay Line - Design and Analysis

While the radiation hardened programmable delay used by the LHCb calorimeter is capable of generating a 1 ns delay with low jitter and DNL, its range is limited to the period of the LHC clock (40 MHz, 25 ns). On the other hand, the delay line that is currently being used by HMPID has a coarse resolution of 25 ns, with a larger delay range of 2.5 µs. The latter architecture was chosen mainly due to its simplicity, but the resolution had to be improved to at least 2 ns, such that it can conform to the new specifications. Figure 3.4 illustrates the chosen design.

The design consists of a 525-bit wide shift register, operated using a 1 GHz clock, with a tap after each delay element fed into a multiplexer. A 10-bit control signal is used to select the desired delay. Theoretically, this architecture is capable of generating a maximum delay of 525 ns with a resolution of 1 ns.

An analytical model is required to study the behaviour of the delay generator architecture shown in Figure 3.4, when this is operated by a jittery clock. Starting from an ideal scenario, this architecture can be modelled by equation 3.1, where T_d is the total delay that can be obtained with N delay elements, T_{prop} is the propagation delay of the flip-flops and any other I/O Buffers, and T_{MUX} is the

propagation delay introduced by the multiplexer. Equation 3.1 assumes an ideal clock with a period of $T_{\text{clk, ideal}}$. Even though T_{MUX} can be significantly large, it can be ignored for simplification purposes, since it only contributes to the systematic error in the delay generated.

$$T_d = T_{\text{MUX}} + T_{\text{prop}} + (N-1)T_{\text{clk, ideal}} \quad (N \geq 1) \tag{3.1}$$

In practice however, clock jitter will have an effect on the output delay as it will cause variations in the periodic time of the clock. The real clock can be modelled by equation 3.2, where ΔF is the frequency deviation at a given instance. This may also be expressed in the form of Equation 3.3.

$$T_{\text{clock,real}} = \frac{1}{F_{\text{clk,ideal}} + \Delta F} \tag{3.2}$$

$$T_{\text{clk,real}} = \frac{1}{F_{\text{clk,ideal}}} + \varepsilon'_{\text{clk}} \tag{3.3}$$

$\varepsilon'_{\text{clk}}$ is the instantaneous clock error which can be related to the instantaneous frequency deviation ΔF and the ideal clock frequency, through the combination of equations 3.2 and 3.3.

$$\varepsilon'_{\text{clk}} = \frac{-\Delta F}{F^2_{\text{clk,ideal}} + \Delta F F_{\text{clk,ideal}}} \tag{3.4}$$

Thus, for every clock cycle, the clock will induce an error ε'_i in each stage i of the delay line, such that Eq. 3.1 can be expressed as:

$$T_{d,real} = T_{\text{prop}} + (N-1)T_{\text{clk,ideal}} + \varepsilon_N \tag{3.5}$$

ε_N is the total error at stage N of the delay line, that is, $\varepsilon_N = \sum_{i=1}^{N} \varepsilon'_i$, where ε_i

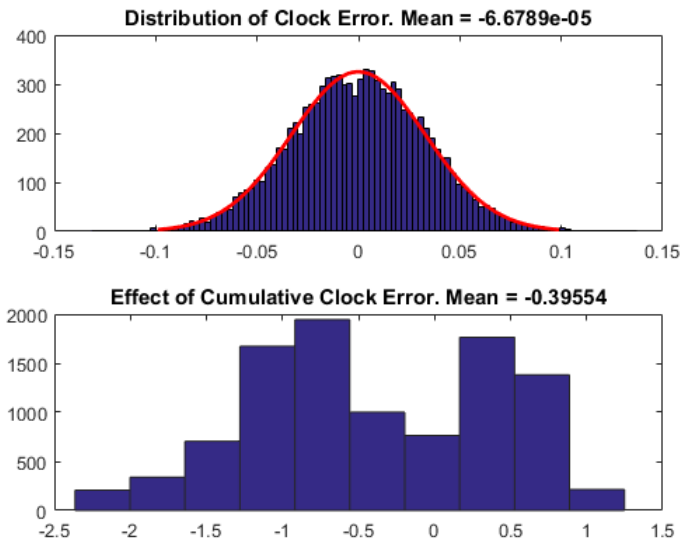


Figure 3.5: Distribution of the clock error (top) and the effect of clock error accumulation on the distribution of delay error (bottom)

is the relative error between the ideal clock and the real clock. This implies that even though ε_i' may follow a known probability distribution with a particular mean and standard deviation, the resultant distribution of the delay will be transformed due to clock error accumulation. For instance, as illustrated in Fig. 3.5, if the distribution of ε_i' is a standard distribution with zero mean, ε_N would have a multi-modal distribution with a different mean, which will be translated to an offset in the transfer characteristic of the delay generator.

3.2.1 MATLAB model for error

A simple MATLAB script was written to further investigate this phenomenon. The code, presented in Appendix A, generates 10000 normally distributed random numbers with a standard deviation of 0.3 and mean of 0, while also generating 10000 ideal clock rising edges. The generated numbers are used to modulate the period of the ideal clock, thus introducing jitter. From the generated periods, the location of the rising edges can be calculated. In turn, other parameters can be

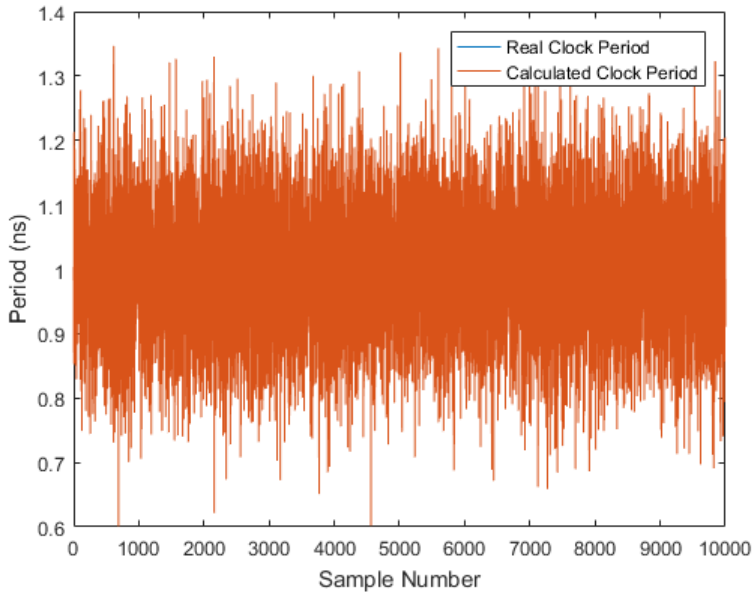


Figure 3.6: Variations in the periodic time of the generated clock

calculated as well, such as the real clock frequency and the frequency deviation. The following are some of the results obtained.

Figure 3.6 shows the variations in the periodic time of the clock. Two plots are shown, the first illustrating the real clock period, as modulated by the random number generator, and the second shows the calculated clock period. This was calculated according to equation 3.4. As can be seen from Figure 3.6, the calculated clock period corresponds directly to the real clock period and therefore this confirms the equation.

Figure 3.7 illustrates the distribution of the clock error as generated by the script, as well as the cumulative clock error. The latter was calculated in two ways. The first method was to sum the generated period error for each stage, as shown in Listing 3.1. The second method involves the subtraction of the rising edges of the real clock, from those of the ideal clock. This resulted in the same error, thus showing that the error is indeed cumulative. From Figure 3.7, it can be seen that while the distribution of the generated clock error follows a normal distribution with

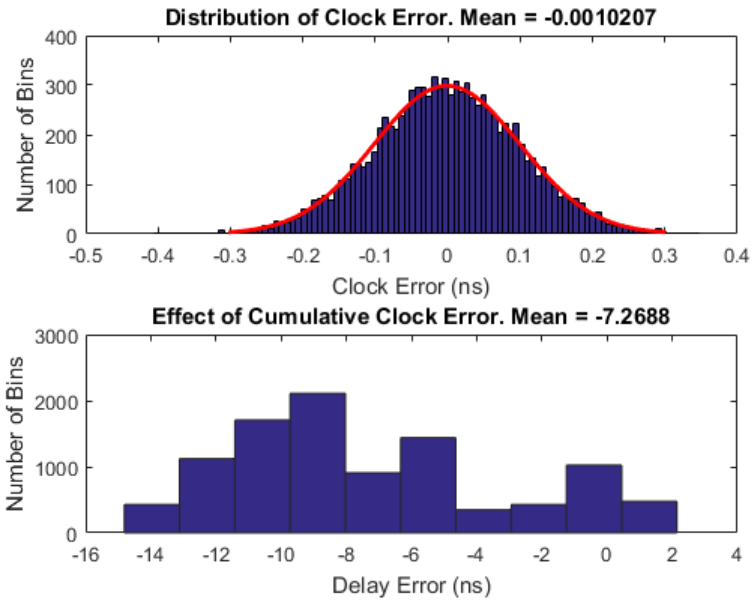


Figure 3.7: The distribution of the clock error (in ns) and its effect when this error is accumulated

mean equal to zero and standard deviation of 0.3, the cumulative effect changes the distribution altogether. A multi-modal distribution with a different mean can now be observed. This also affects the distribution of the clock frequency as seen by the architecture. It can be observed in Figure 3.8 that the mean of the clock frequency has shifted to 1.012 GHz.

Listing 3.1: Summation of the Period Error

```
for i = 1:length(R)
    cumulative_error_eachstage(i) = sum(R(1:i));
end
```

The end result of this analysis shows that this architecture will generate an error that is unpredictable. This will result in the generation of an offset corresponding to ε_N, in equation 3.5. Figure 3.13 shows the variation of the offset in the transfer characteristic of the delay generator for different values of clock jitter. This delay offset is random in nature and thus difficult to predict.

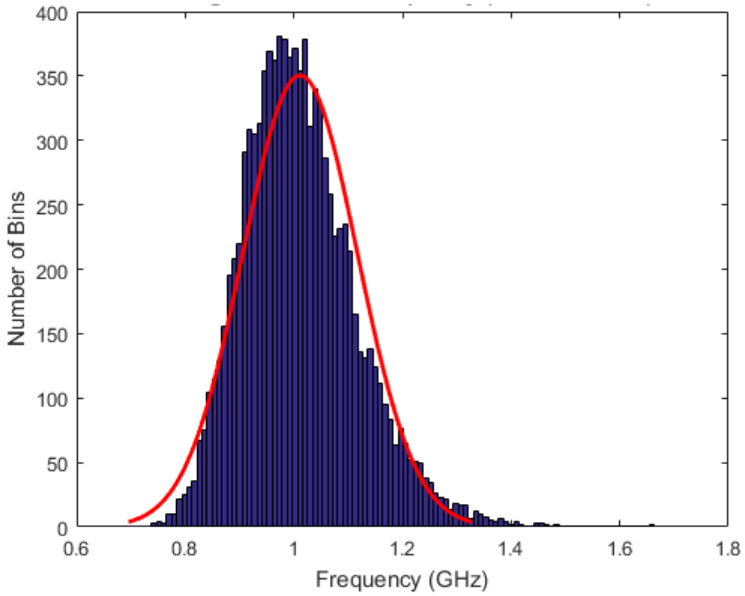
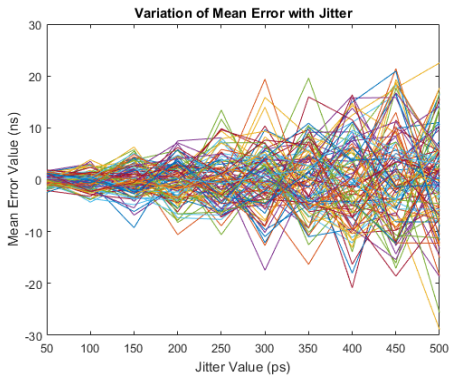


Figure 3.8: Histogram of clock frequency

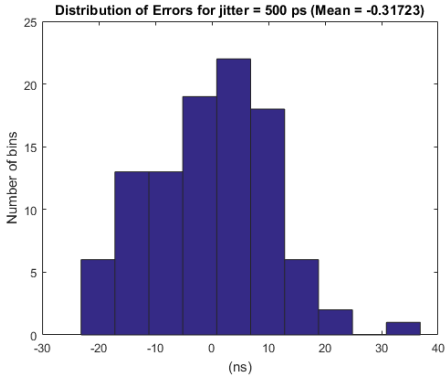
Another MATLAB script was written to show the effect of clock jitter on the generation of the offset, and to study its effect. The script, generates 10000 random numbers with varying standard deviation (from 0.05 ns to 0.5 ns) in steps of 0.05 ns. The error is calculated by comparing the real clock signal with the ideal clock signal, and the mean value is taken and plotted. A selection of the results obtained is shown in Figure 3.9. The randomness of the error can be clearly seen in Figure 3.9a. The distributions of the error for different values of jitter are also shown in the Figure 3.9a, together with their means. The means are plotted in Figure 3.10, and continue to show the unpredictability of this value.

3.2.2 Conclusion

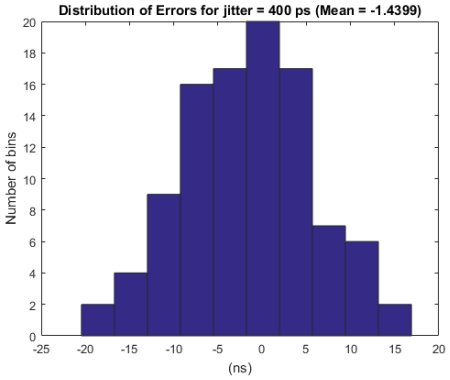
This section has introduced the analytical model of a shift-register based architecture, when used as a delay generator. It was seen that when introducing clock jitter, a random offset is generated. This analytical model also shows that the main advantage of using the shift-register architecture is that as the number of taps in-



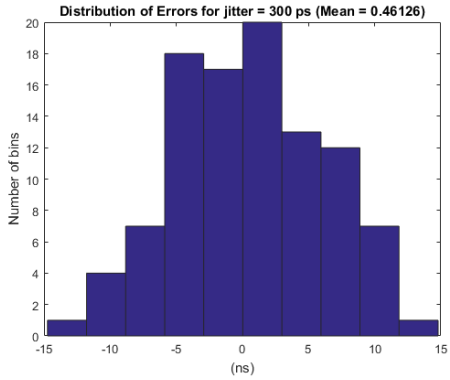
(a) Variation of Mean Error for Different Jitter Values (Simulation Run for 100 times)



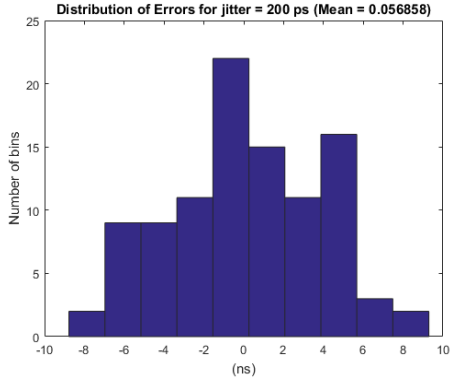
(b) Distribution of Errors induced by a clock with jitter of 500 ps



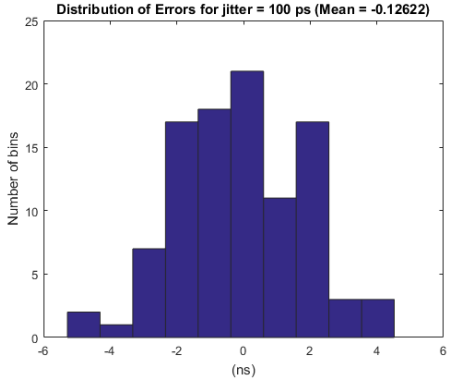
(c) Distribution of Errors induced by a clock with jitter of 400 ps



(d) Distribution of Errors induced by a clock with jitter of 300 ps



(e) Distribution of Errors induced by a clock with jitter of 200 ps



(f) Distribution of Errors induced by a clock with jitter of 100 ps

Figure 3.9: Effect of Clock Jitter on Generation of Offset

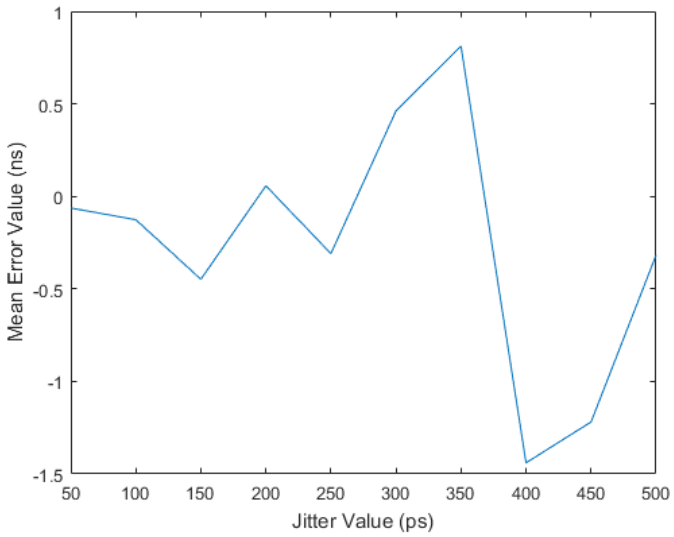


Figure 3.10: Means of the generated offset plotted against the jitter, as extracted from MATLAB model

creases, the average delay between each stage will approach the periodic time of the ideal clock. This means that given a sufficient number of stages, the average delay error will tend to zero. This effect can be predicted by equation 3.6.

$$\mu_{T_d} = \frac{T_{\text{prop}}}{N} + T_{\text{clk,ideal}} - \frac{T_{\text{clk,ideal}}}{N} + \frac{\varepsilon_N}{N} \tag{3.6}$$

3.3 Simulation of Model

This section presents the methodology for simulation of the chosen architecture. In particular, a model for D flip flop was written in C++ that mimics exactly the operation of a D Flip-Flop in a Xilinx Virtex-5 FPGA.

A C++ programme that simulates the tapped output delay generator architecture shown in Figure 3.4 was developed, in order to verify how the system will perform when fed by a jittery clock. The programme consists of a source generator and a model for the D flip-Flop.

3.3.1 Source Generator

The *Source Generator* class provides functions and variables to the *Clock Generator* and *Data Generator* classes. It contains variables related to the sampling frequency, generation frequency, and signal start delay. The class contains methods to set each of these parameters, as well as another method to generate data. The *genSource* method accepts as a parameter the current *timeStep*, which is the sample number. The output $\text{generator}_{\text{out}}$ is calculated as follows:

$$\text{generator}_{\text{out}} = \begin{cases} \text{generator}_{\text{out}} \oplus 1 & \text{mod}(\frac{f_s}{2f}) = 0 \\ \text{generator}_{\text{out}} & \text{otherwise} \end{cases} \tag{3.7}$$

where f_s is the sampling frequency, and f is the source generation frequency.

3.3.2 Clock Generator and Data Generator

The *Clock Generator* class consists of two overloaded methods called *genSource* each of which capable of accepting different parameters. The first method accepts an integer called *timeStep*, and calls the *genSource* method from the Source Generator class, while the second method accepts two other parameters called *nextEvent* and *lastEvent*. The purpose of these two parameters is to work out when the next event would occur when jitter is introduced. The code for this method is given in Listing 3.2.

Listing 3.2: Generation of a clock event under the presence of jitter

```
int clockGen::genSource(int timeStep, int lastevent, int nextevent){
    int halfway = (nextevent - lastevent)/2;
    if (timeStep == lastevent || timeStep == lastevent + halfway){
        m_prev ^= 1;
    }
    return m_prev;
}
```

The construction of the *Data Generator* is similar to the clock generator but can only generate data without any jitter.

3.3.3 DFF Model

The DFF model takes as its parameters the sampling frequency, and the propagation delay, C_2Q. The latter denotes the time it takes for the input to be seen at the output when triggered by a rising edge of the clock. The DFF detects a rising edge of the clock and samples the input data. When a rising edge occurs, the DFF starts counting until it reaches the value of C_2Q, and then it propagates the input to the output.

3.3.4 Results from Model

Simulations were performed with a sampling resolution of 1 ps and a simulation time of 2 μs. The results from the model with an ideal clock are presented in Figure 3.11, where the delay value is plotted against the tap number. As expected, the response of the system is perfectly linear, with an average delay value of 1 ns. The offset, -0.7 ns, stems from the design of the D flip-flop model where the C_2Q propagation time is equal to 0.3 ns (for $N \geq 1$ in Equation 3.5). The offset is negative due to the subtraction of $T_{\text{clk,ideal}}$.

A clock with finite jitter was then introduced in the model and simulations were performed for different cycle-cycle jitter values. A standard distribution based random number generator was used with its mean set to zero and a standard deviation of one third the maximum jitter required. Figure 3.12 shows the transfer characteristic of the delay generator for a clock jitter equal to ± 250 ps, where the time delay together with the best line of fit are plotted. As can be seen, since the clock error is small when compared to the the periodic time of the clock, the response of the system is still linear. The method of least squares was used to

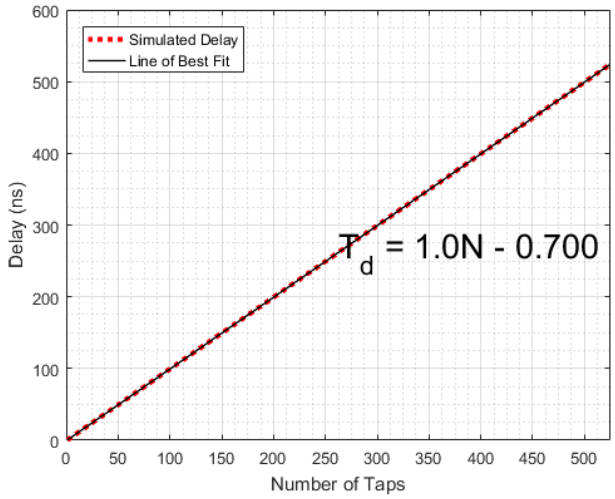


Figure 3.11: Simulated Ideal Case where the Average Delay is equal to 1 ns, with an offset of -0.7 ns

calculate the y-intercept of the line of best-fit. The gradient of this line was forced to 1 such that the contribution of the clock error will only affect the y-intercept, as per equation 3.5. While an offset of -0.7 ns was predicted for the ideal case, this has changed to -0.519 ns. This change of 0.181 ns is exactly equal to the mean of the cumulative clock error, ε_N.

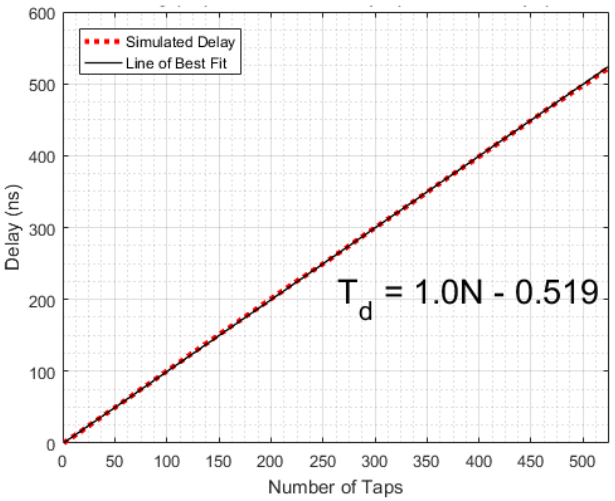


Figure 3.12: Simulation of the delay generator with a clock jitter of $\pm$ 250 ps resulting in an offset of -0.519 ns

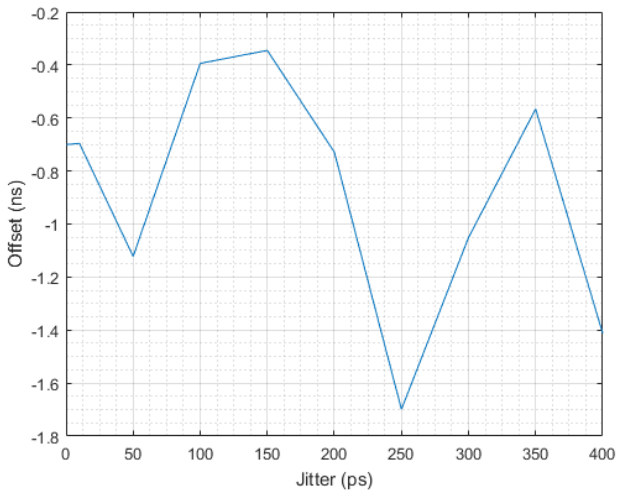


Figure 3.13: Variation of the offset in the transfer characteristic of the delay generator for different values of clock jitter, as extracted from C++ model

3.4 Implementation, Testing, and Results

The delay generator that is illustrated in Figure 3.4 has a simple design, produces linear delay steps and can easily be implemented on the Xilinx Virtex-5 FPGA. Most importantly, the delay generator is reliable and does not carry any stability issues since it is an open loop architecture.

The delay generator was therefore based on this architecture. The implementation, illustrated in Figure 3.14, consists of three modules: the clock manager, the delay module, and the output multiplexer. The clock manager consists of a Xilinx digital clock manager (DCM) followed by an all digital phase locked loop. This module takes the differential 200 MHz clock and multiplies it to obtain a 1 GHz clock. This clock is then used to drive the 525-bit shift register present in the delay module, thus obtaining the required delay resolution. The last component in the architecture is the output multiplexer that taps the shift register such that the required delay can be selected, through a 10-bit control word.

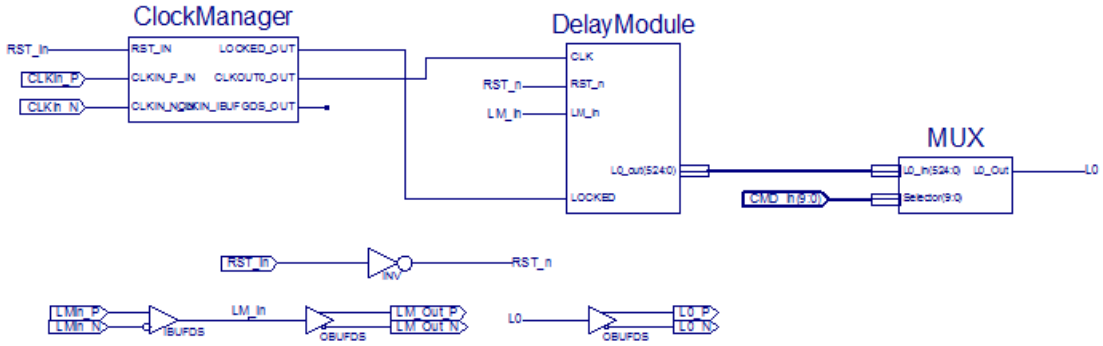


Figure 3.14: High level view of the architecture

3.4.1 Floorplanning

Since timing is critical for this architecture, floorplanning was used such that the design can adhere to the timing constraints in place. In particular, this technique was used to limit the shift register to a single clock region, thus eliminating the risk of setup and hold violations that may arise. This was done by splitting the shift register into blocks of 25, and assigning a *pblock* for each of these groups. The floorplan layout is illustrated in Figure 3.15.

3.4.2 VHDL Test Bench

A VHDL test bench was developed in order to simulate the delay generator while considering a post place and route FPGA model. The test bench consists of a process that generates a command between 0 and 525, and sends it to the multiplexer. This process also generates the LM signal, to be delayed. The outputs are saved through the TEXTIO library. Once a simulation is finished, the results are loaded by a MATLAB script for analysis.

The results from the VHDL testbench are illustrated in Figure 3.16. In this case, the offset introduced is equal to -1.357 ns. Even though there are no errors related to the clock jitter, another type of error has been introduced between each stage. This is the systematic error coming from the physical placement of the shift register

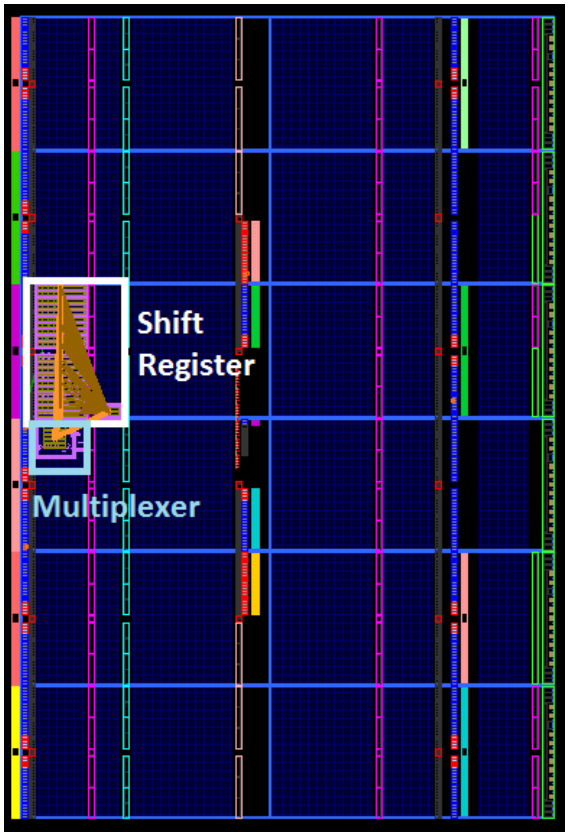


Figure 3.15: Floorplan of delay generator as implemented on a Virtex-5 FPGA

on the FPGA, buffers/drivers and the multiplexer delay.

3.4.3 Physical Test Bench

Two Xilinx FPGA boards were used to test the system (Figure 3.17). The first board emulates the generation of the LM trigger signal, with a frequency of 200 kHz and sends the signal along with the required delay. This was sent through the high-speed VHDCi port on the board. The FPGA board is controlled through the serial RS232 protocol, via MATLAB.

A PCB was designed such that the VHDCi port of the second board can be used to output both the received signal (LM) and the delayed signal (L0). The design of the PCB is illustrated in Figure 3.18.

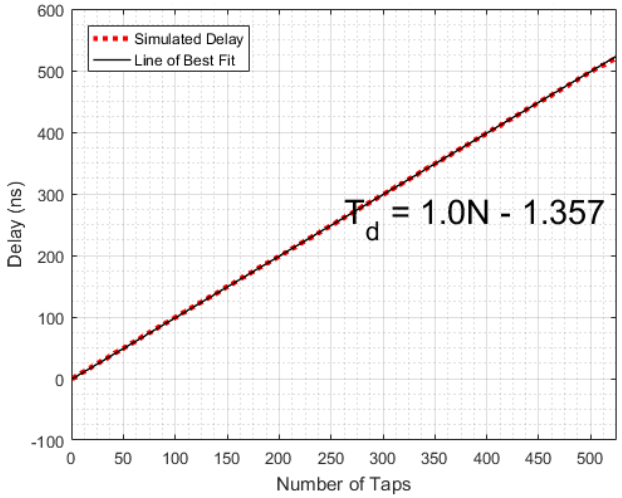


Figure 3.16: Results from VHDL Testbench after Place and Route Procedure

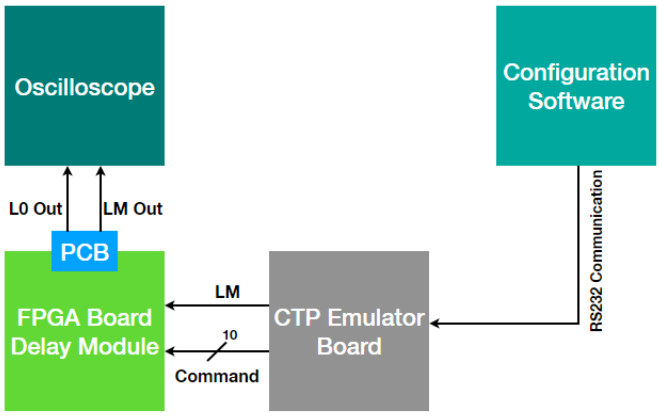


Figure 3.17: Testbench Setup

The LM trigger signal and the delayed signal L0 were then read by a Tektronix TDS820 Oscilloscope and the delay was calculated. Since the mathematical model of Equation 3.5 only describes the shift register architecture, it was necessary to remove the delay contribution added by any input and output buffers, look up tables and the output multiplexer.

While an offset of -1.357 ns has been predicted by the VHDL test bench simulation, in this case the offset has increased to 1.957 ns. This is due to the fact that apart from the systematic offset that has been introduced by the architecture, the clock

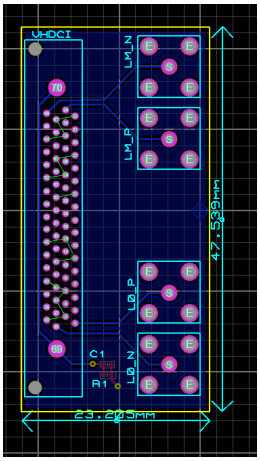


Figure 3.18: PCB Layout of VHDCI to SMA Converter

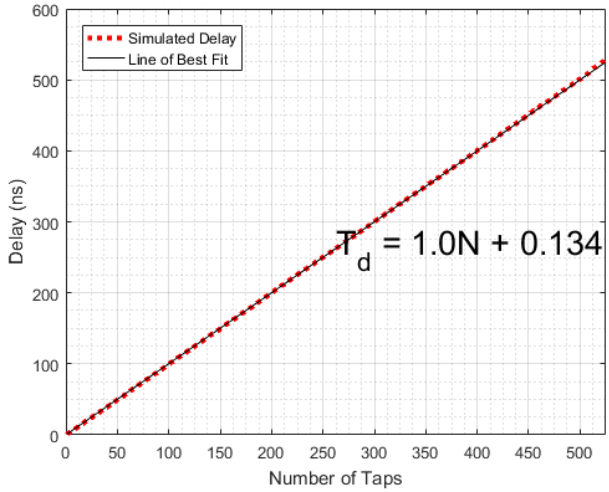


Figure 3.19: Error contribution due to the clock jitter only

error will now have an effect. To see the contribution of the clock signal only on the output, the simulation results can be subtracted from those obtained from the VHDL test bench, thereby excluding any systematic errors. This will result in an offset equal to 0.134 ns (Figure 3.19).

3.5 Comparison with Previous Works

Table 3.1 presents a comparison between the delay generator currently used by HMPID [38], a DLL-based delay generator used by the LHCb calorimeter [37] and this work. It can be seen that this work is the most suitable for use by HMPID as it satisfies both the delay range and resolution requirements. In addition its DNL is lower than that of the delay generator used by LHCb.

Table 3.1: A comparison with other delay generators within the LHC

Work	Delay Range	Resolution	DNL
Current Module [38]	2.5 μs	25 ns	N/A
DLL Based [37]	25 ns	1 ns	23 ps
This Work	525 ns	1 ns	5 ps (simulated)

3.6 Offset Compensation Mechanism

From the previous discussion, it was seen that an offset is generated, due to the realisation of the circuit on the FPGA (Figure 3.20). To overcome this, two methods are proposed:

1. **Hard-Coded Compensation:** This method involves directly subtracting the received command to compensate for the error in the delay. Figure 3.20 illustrates the error for each tap number. By rounding the error, the command value can be adjusted accordingly. Although this method is simple, and can be implemented on either the FPGA or on the control software, the main disadvantage is that the error may change with temperature and voltage variations.

2. **Online Compensation:** This method involves the use of a feedback path mechanism to automatically adjust the command value that has been sent. A Time-to-Digital Converter (TDC) is required to count the produced delay and compare it with the required delay, and then compensate accordingly.

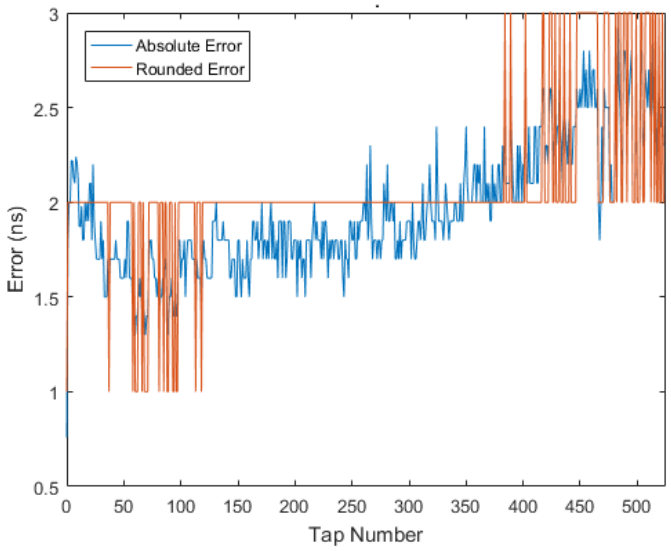


Figure 3.20: Error in the measured delay, absolute and rounded error

The online compensation method was chosen to be implemented for two reasons; the first being the ability to compensate as much as possible for voltage and temperature variations, and the second reason to be able to provide real-time feedback to the user.

Apart from describing the chosen architecture, this section will provide a brief background on Time to Digital Converters, and their implementation on FPGAs.

3.6.1 Brief Background and State of the Art

The precise measurement of time intervals is frequently needed in many applications such as High Energy and Nuclear Physics [22, 39, 40]. Particle identification using precise time measurements is used in the time-of-flight technique, and its efficiency is directly related to the accuracy of the time measurement [41]. Tapped Delay lines are frequently used to measure picosecond time resolutions [40]. A Time Interval Meter (TIM) is used to convert a time interval into a digital word. The term Time-to-Digital Converter (TDC) can be used interchangeably [42].

The most important parameters of a TDC are [42]:

- Measurement Range (MR)
- Precision Error
- Linearity, measured through Differential Non-Linearity (DNL) and Integral Non-Linearity (INL)
- Resolution
- Dead Time
- Readout Speed

Measurement methods can be split into coarse measurements, and fine resolution measurements. The simplest method of coarse measurement utilises a counter, driven by a reference clock. The resolution of this type of counter depends on the operating frequency of the clock. While using a ripple-counter can be a simple solution to perform time interval measurement, it poses several problems. The first relates to the use of a high-frequency to achieve high-resolution readout. In this case both a stable clock and fast electronics are required. Moreover, with an m bit counter, a maximum range of 2^m can be achieved. This can simply be solved by using a linear feedback shift register, using an XOR-gate as feedback. However this technique generates pseudo-random code, which would require a separate converter. A counter in free-running mode can also be used to calculate the time interval. In this case, the counter is sampled when a START or STOP signal is received. An overflow counter is required to detect overflows [42].

Fine measurements, as their name implies, allow for better accuracy than their coarse counterparts. Various methods allow for fine measurement of a time delay, and they can utilise either analogue and/or digital techniques. Techniques such as Time-to-Amplitude conversion are hybrid, and they utilise a charge pump followed

by an analogue to digital converter. This can achieve very high resolutions between 1 ps and 20 ps [42]. The Vernier method, is a purely digital approach. The basic configuration of the Vernier converter utilises two startable oscillators, having frequencies that are slightly different from each other. The resolution that can be achieved with this method depends on the time difference between the two clock periods. A conversion successfully occurs when the edges of the clocks coincide. A resolution as low as 1 ps has been obtained in literature [43, 44]. Another technique utilises a tapped-delay line architecture, such as the one previously described. The START signal is applied to the D input of a D Flip-Flop, and the STOP pulse is applied to an active low clock input of the flip-flop. The measured time interval would be equal to the propagation time multiplied by the number of flip-flops which have a high output, meaning that the START pulse has successfully propagated [42].

As with delay lines, there is a trade-off between the time interval range that can be read and resolution. To overcome this, interpolation methods are used. The Nutt method is one such example, and it has been successfully implemented on an FPGA [39, 40]. While other techniques have been implemented on an FPGA [45–49] this method is simple, low-cost, and provides both a high resolution and a large range. Figure 3.21 illustrates the basic concept of the Nutt Method.

The time interval consists of 3 parts:

$$T = \Delta t_1 + \Delta T_{12} + \Delta t_2 \tag{3.8}$$

While the time interval ΔT_{12} can be coarsely measured using a simple counter, a finer resolution is required to measure the other two time intervals. To achieve this, a time stretcher based on a simple shift-register has been employed in [42]. The stretched signal is then counted and a factor is used to count the interval. To measure this interval Aloisio et al. [39, 40] designed two fine time converters, one

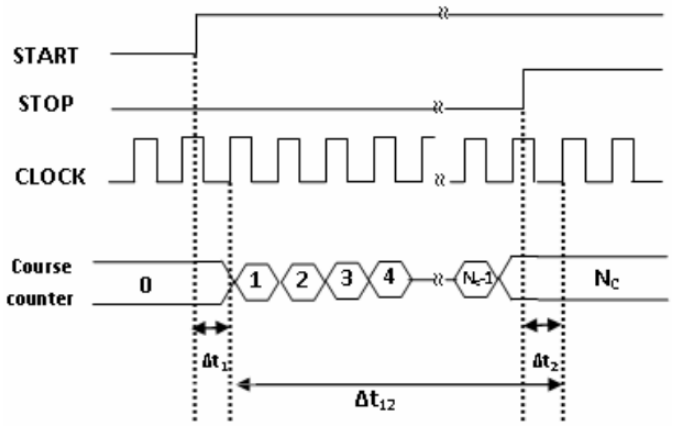


Figure 3.21: Time Interval Measurement with the Nutt Method [39]

based on tapped delay lines while the other is based on Vernier delay lines. The coarse delay line is based on a free-running counter that saves a snapshot when a START signal is applied. The same occurs when the STOP signal arrives. The design makes use of a 550 MHz oscillator, with 4 clock phases such that a better resolution of the time interval is achieved. The phase difference between the START and STOP signals can then be calculated. Through this method, a coarse resolution of 454 ps was obtained. The combination of coarse and fine measurements resulted in a resolution of 60 ps [39, 40].

3.6.2 Design and Implementation

Since a systematic offset has been generated, it is necessary to measure the delay between the input LM signal and the output L0 signal. As such a time-to-digital converter has been developed. The design for the TDC is based on that of Aloisio et al. [39], and is illustrated in Figure 3.22. The design consists of a free-running counter, a phase state machine, and an integrator. To increase the resolution of this TDC, four clock phases are used.

On arrival of an LM signal, a snapshot is taken of the coarse counter and the value is saved by the integrator block. When the L0 feedback signal is received, the

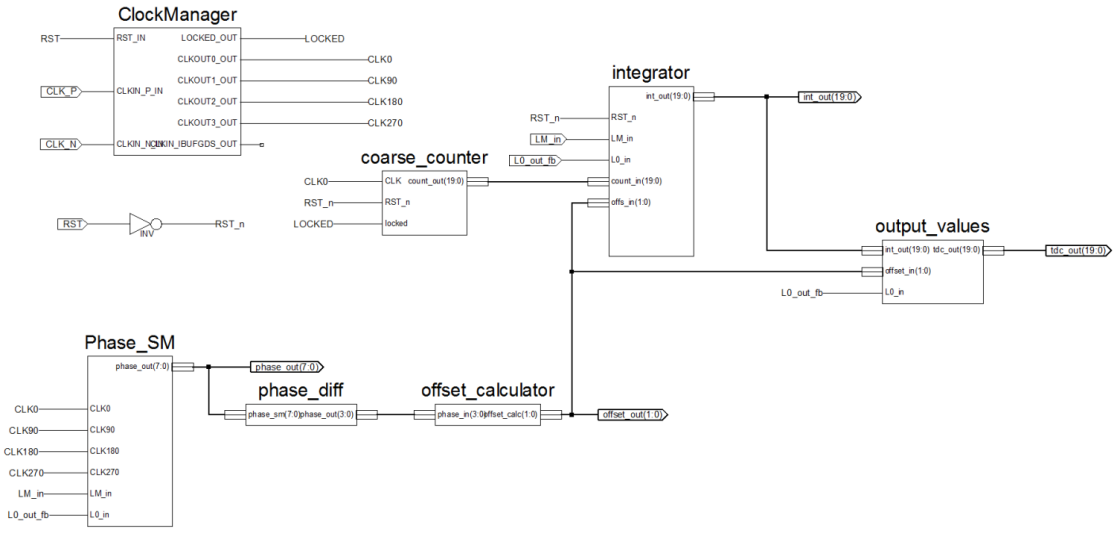


Figure 3.22: Design and Implementation of Time-to-Digital Converter

value of the coarse counter previously saved, is subtracted from the current value, thus obtaining a coarse delay with resolution of 2 ns. To increase the resolution, the *Clock Manager* block generates four phases of the 500 MHz clock, with a 90° spacing. With every LM signal, a snapshot of the four clock phases is taken. The same happens when an L0 signal is received. The phases are encoded, and the phase difference is calculated. The *output values* block adds an offset to the value from the integrator, to obtain the required 1 ns resolution.

3.6.3 Testing and Results

A simple test bench was written to generate an LM signal, together with a delayed L0 signal. Thus, the TDC would be tested for different delay values. The main results are illustrated in Figure 3.23. The signals are split into three groups. The *Trigger Inputs* group consists of the LM and L0 signals. The delay between these 2 signals is equal to 275 ns. The second group is the readout from the TDC. Signal *tdc_out* is a 20-bit vector containing the delay value between LM and L0, to the nearest 1 ns. The result corresponds directly to the actual delay value. The other outputs are the actual phase offset value, the sampled clock phases when LM and

L0 occurred, and the value of the coarse integration, to the nearest 2 ns.

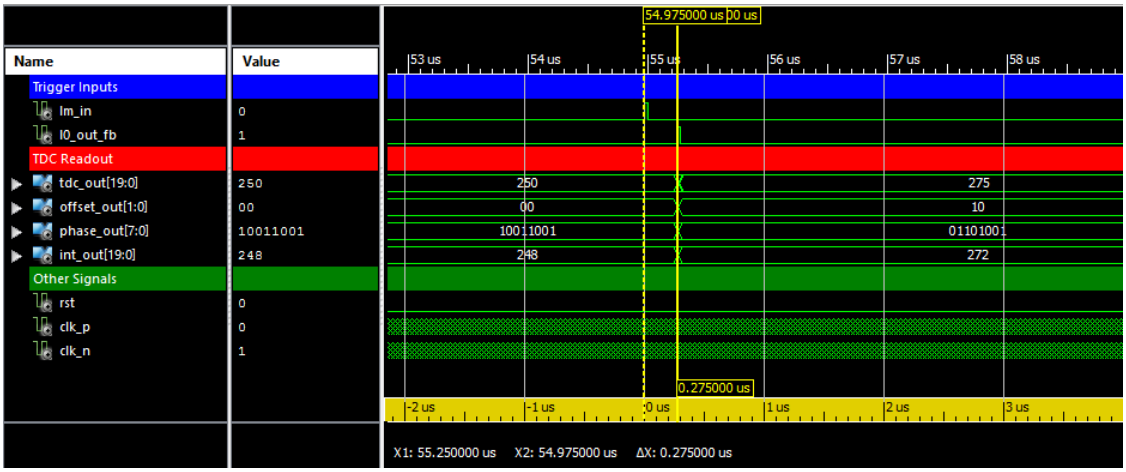


Figure 3.23: Results from FPGA implementation of Time-to-Digital Converter

3.7 Conclusion

This chapter has given an account on the design of a digital delay generator implemented on an FPGA, having a delay range of 525 ns with a resolution of 1 ns. This digital delay generator is based on a shift-register architecture, and was successfully implemented on a Xilinx Virtex-5 FPGA. This architecture has been analysed using a simple mathematical model, and a simulator was written in C++ and MATLAB to verify the operation of the architecture. In comparison to the delay generator currently in use by HMPID [38], the shift-register based architecture is capable of generating a controllable delay with a finer resolution (1 ns against 25 ns), albeit with a lower delay range which was 2.5 µs. Although the delay range is lower, this is much higher than the delay-range of the DLL-based delay line used by LHCb [37], where the maximum delay range is 25 ns, with a resolution of 1 ns.

It was noted that, although the delay is linear, variations in the clock signal will result in an unpredictable offset. This can prove problematic, and therefore the chapter has introduced solutions to this issue. Two methods were proposed, with the first being the introduction of a hard-coded compensation mechanism, where

the sent command is subtracted to compensate for the error in the delay. The second method is to introduce a Time-to-Digital converter in the feedback of the delay line such that on the fly compensation can occur. This method is capable of compensating for any voltage and temperature variations while also providing feedback to the user regarding the generated delay with an accuracy of 500 ps. While this result is coarser than what has been achieved in literature, it is sufficient for this application.

4. ASIC Design, Simulation and Testing

Following the design and testing of an FPGA based purely digital delay generator, this chapter introduces a hybrid analogue/digital delay generator implemented around a DLL architecture. The work in this section has been published by the author in [50–52] and [53].

4.1 Background - Delay Locked Loops

A DLL is a closed-loop system that is able to generate precise delays. In fact, the output signal has a precise phase relationship with the input (or reference) signal [13,54]. The DLL has numerous advantages. In particular a DLL is unconditionally stable and can compensate for PVT variations and noise, with a jitter performance in the picosecond range [13, 55].

Figure 4.1 shows a typical implementation of a DLL. The architecture consists of a phase detector, a charge pump, a loop filter, and a VCDL. Since the only state variable in the loop is the phase, only a first order loop is required [21]. The purpose of the PD is to compare the phase of the feedback signal with that of the reference signal and generate an error signal. This error is then converted to a charge through the charge pump. The loop filter converts the charge to a control

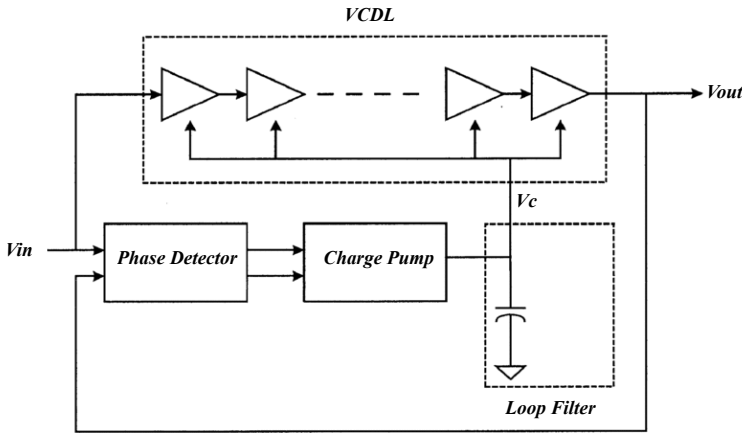


Figure 4.1: Implementation of a delay locked loop [57]

voltage, and consequently, tunes the delay of the VCDL. Since the system employs negative feedback, the phase error is eventually minimised to zero. This process also ensures that the delay generated by the VCDL is equal to one cycle of the reference signal [56].

Jia and Milor [56] identify four performance metrics that characterise a DLL: the lock range, the locking time, the jitter performance, and the static phase error. A wide lock range allows the DLL to lock to a large range of input frequencies. The number of delay elements in the VCDL has a direct effect on the lock range. In other words, the maximum lock range of the DLL is set by the maximum delay that can be generated by the delay line. This can be achieved either by utilising a large number of delay elements, each of which having a comparatively short unit delay, or by using a smaller number of delay elements with larger unit delays. The former technique utilises more power and area than the second technique. On the other hand, the latter may suffer from jitter degradation as the transitions are slower [13, 56, 58–60]. The locking time of a DLL is a measure of the time it requires to minimise the phase error to zero, thus locking with the input signal. The static phase error is the steady state error between the input and output signals of the

DLL. This error arises from the limited resolution of the CP and PD. Finally, jitter is the random deviation in the output delay time from the desired value.

The locking time, lock range, and jitter are related to each other through the bandwidth of the loop. A low bandwidth will result in an attenuation of the jitter at the cost of degrading the lock time and the lock range. A trade-off therefore exists between these parameters [56]. The finest change in the generated delay time is defined as the delay step, and it is controlled by the control voltage, V_c. This voltage modulates the charging and the discharging current feeding the output capacitance of the delay element and as a result changing the delay.

A DLL may also be implemented in a digital manner, however its delay resolution is not as fine as that attained by its analogue counterpart, and also occupies more area. It has worse jitter performance, and its response is less linear. The digital implementation does however has some advantages particularly in the simplicity of the design, and power consumption. A digital DLL typically consists of a PD, an up-down counter, some form of finite state machine, and a digitally controlled delay line [13, 54, 56].

4.2 Architecture Overview

The proposed delay generator architecture is illustrated in Figure 4.2 and it mainly consists of a clock generator, a programmable digital frequency divider and a DLL. The clock generator is used to generate a square wave of 500 MHz, which corresponds to a periodic time of 2 ns. The fully-programmable digital divider is fed by this clock generator such that the input frequency to the DLL can be changed in steps of 2 ns.

On start-up, the clock generator is connected to the DLL via the divider (switch position A). In a DLL, the delay provided by the VCDL at lock state is equal to

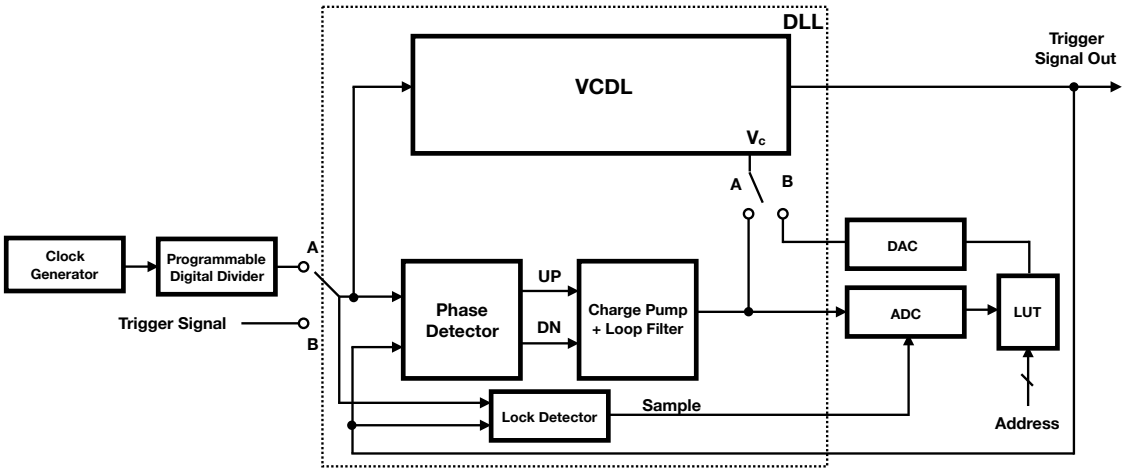


Figure 4.2: Block diagram of the proposed delay generator architecture

the periodic time of the input signal. During the initial calibration of the delay generator, the resultant control voltage (V_c) for different input frequencies generated by the programmable frequency divider is digitally converted and stored in a Look-up Table (LUT) by means of an ADC. This process is used to generate a delay transfer characteristic for digitally controlling the VCDL via a DAC and provide the required trigger signal delay in open loop (switch position B). The DLL based calibration is essential in order to ensure that the delay generated is insensitive to process, voltage, and temperature variations.

After the calibration stage is completed, the DLL feedback loop is opened (switching to position B) such that the LM trigger signal (which is a non-periodic signal) can be delayed by the required amount. The control voltage is obtained from the LUT by converting the digital value generated remotely to an analogue voltage.

4.3 Technology Considerations

The first step in designing an ASIC is to choose the correct technology with which to implement the design. The following technologies were available from the Europractice IC [61] service, at time of designing the ASIC:

- AustriaMicroSystems (AMS) - The C35 0.35 µm technology from AMS is popular in designing low-power digital, analogue, and mixed-signal circuits. This process can have either 3 or 4 metal layers for routing, depending on the chosen technology option.

- United Microelectronics Corporation (UMC) - The L180 0.18 µm process from UMC is mainly suitable for mixed mode and RF circuits. This technology process can have up to 6 metals for routing.

- Taiwan Semiconductor Manufacturing Company Ltd (TSMC) - The TSMC 0.18 µm technology is suitable for high-voltage, analogue, digital, and mixed-signal circuits. In addition this technology provides 6 metal layers which are suitable for routing complex digital designs.

- X-FAB - The XH018 0.18 µm technology from X-FAB is suitable for analogue, and mixed-signal circuits. Depending on the chosen process, the XH018 technology allows up to 5 metal layers for routing.

While all the four technologies mentioned above are suitable for implementing the delay generator architecture previously described, the costs vary significantly. Table 4.1 compares the cost of fabricating a 6 mm^2 ASIC, using the above technologies. Although the cheapest way to implement the ASIC is to use AMS technology, the minimum feature size is larger than the others (0.35 µm versus 0.18 µm). In addition, the XH018 process includes a number of analogue libraries, such as ADCs and DACs, which make it more suitable for the design and fabrication of the delay generator.

Technology	**Price (€)**
AMS	3,480
UMC	24,940 (25 mm^2 block)
TSMC	25,500 (Estimate)
X-FAB	15,050

Table 4.1: Cost comparison of suitable technologies for a 6 mm^2 ASIC

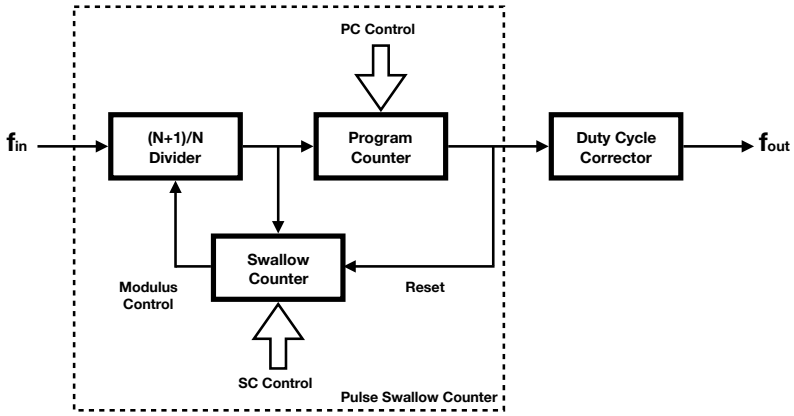


Figure 4.3: Fully Programmable Digital Divider

4.4 Fully Programmable Digital Frequency Divider

The first stage in the proposed architecture is the programmable digital divider, and is essential such that the DLL can be calibrated to achieve the required delay. The divider (illustrated in Figure 4.3) consists of two elements; the pulse-swallow counter, and the duty cycle corrector. The pulse-swallow counter architecture consists of three blocks; the dual-modulus prescaler, the program counter, and the swallow counter.

The dual-modulus prescaler is a counter that divides the input frequency by either N or $N+1$ according to the modulus control input. The swallow counter is a circuit that divides the input signal by the constant division ratio, S. This counter acts as a controller for the modulus of the prescaler, while also having a reset input. The program counter is also a divider but it has a constant division ratio of P. When this up-counter reaches P, it sends a reset signal to the swallow counter [62].

A divider with this architecture can achieve a division ratio, D,

$$D = NP + S \tag{4.1}$$

where P is greater or equal to S.

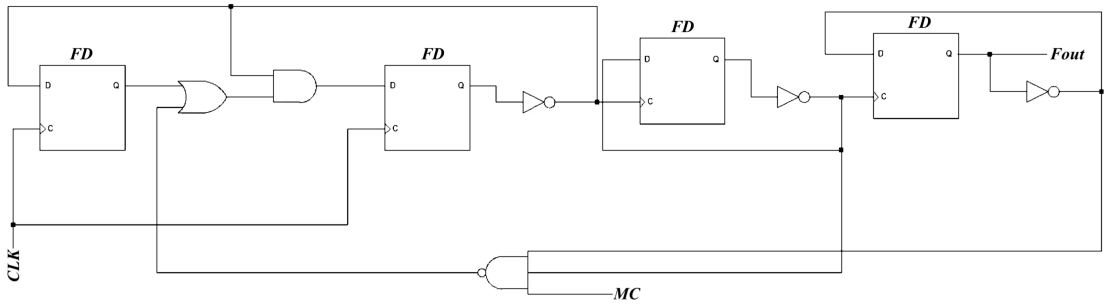


Figure 4.4: Implementation of 8/9 divider in Xilinx ISE

Contrary to typical implementations of such digital dividers, where only the swallow counter is programmed for channel selection, in this work the divider was designed such that both the program and swallow counters can be programmed such that a wide division ratio can be achieved that caters for all the input frequencies required to obtain the desired delay, while keeping a small value for the (N+1)/N divider.

The last block in the fully programmable digital frequency divider is the duty cycle corrector block, which increases the duty cycle of the generated frequency to approximately 50%. This was specifically done such that the output signal has a wide duty cycle that can be fed to the DLL.

4.4.1 Implementation

The fully programmable digital divider was implemented in VHDL following the architecture presented in Figure 4.3. The prescaler was implemented by instantiating Xilinx primitives, as illustrated in Figure 4.4. The program and swallow counters were implemented as up counters in VHDL, with the code for the swallow counter illustrated in Listing 4.1.

Listing 4.1: Implementation of Swallow Counter in VHDL

```
main : process(Fin, RST_in)
        variable count : std_logic_vector(3 downto 0) := "0001";
begin
```

```
        if(RST_in = '1') then
                count := "0001";
                MCout <= '1';
        elsif(rising_edge(Fin)) then
                if (count < Control_in - 1) then
                        count := count + 1;
                        MCout <= '1';
                else MCout <= '0';
                end if;
        end if;
end process;
```

To convert the divider into layout, it was first synthesised with Genus. A tcl script was written to simplify the loading of libraries and to automate the process of synthesis. This script automatically compiles, elaborates, synthesises, and optimises the design. A verilog file is generated that contains the synthesised design. This file was modified slightly to add a buffer (BUHDX0) in the feedback path of the divider. Innovus was then used to translate the synthesised verilog file into layout. The power rings and pins were added with the Innovus layout editor and the router was used to add power stripes. The components were automatically placed and routed using the "placer" and "nanoroute" functions. Filler cells (decoupling capacitors) were finally added to increase the density of the layout. The final layout is illustrated in Figure 4.5.

4.5 Delay Element

Figure 4.6 illustrates a typical implementation of the current starved inverter architecture. The input waveform to be delayed is applied at the V_{in} terminal, while the output appears at V_{out}. The delay is dependent on the charging/discharging current of the capacitor, C_L, and the current is controlled through the voltages V_{cp} and V_{cn}. The relationship between the delay and the tuning voltage may be

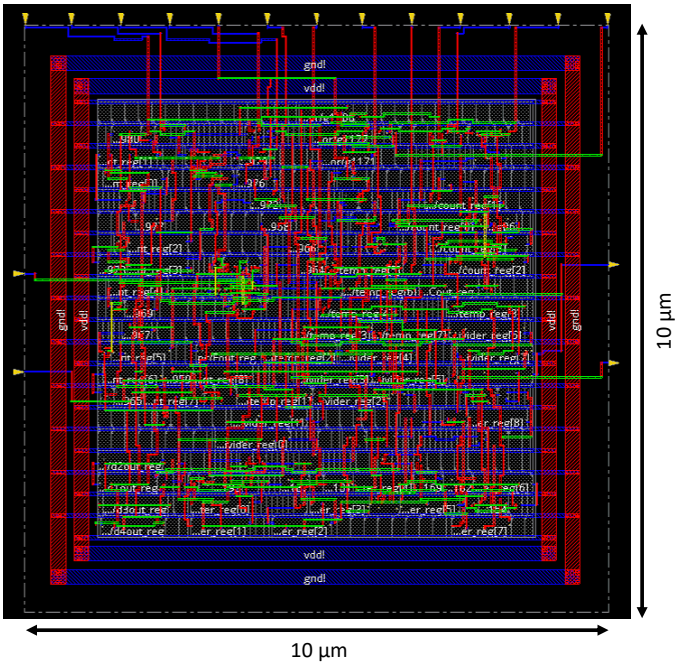


Figure 4.5: Layout of the divider in Innovus

modelled by equation 4.2 [63, 64] where C_L represents the capacitive load of the inverter and I_c is the charging/discharging current through the capacitive load. This equation shows that the delay may be varied by change the values of either C_L, I_{cp}, or V_{DD}.

$$T_d \propto \frac{C_L}{I_c} V_{DD} \tag{4.2}$$

Two problems are immediately evident with this architecture. First, since typically only the current is varied, the relationship between the delay and the tuning voltage is highly non-linear. Secondly, the input range is limited from the threshold voltage of the control transistors, V_t, to the supply voltage, V_{DD}. The constant of proportionality of equation 4.2 depends on the values of the load capacitance and the charging/discharging current. For large capacitances, the discharge current would be limited by the sizing of the control transistors. This is equivalent to having a constant current discharging a capacitor where, if it assumed that the dis-

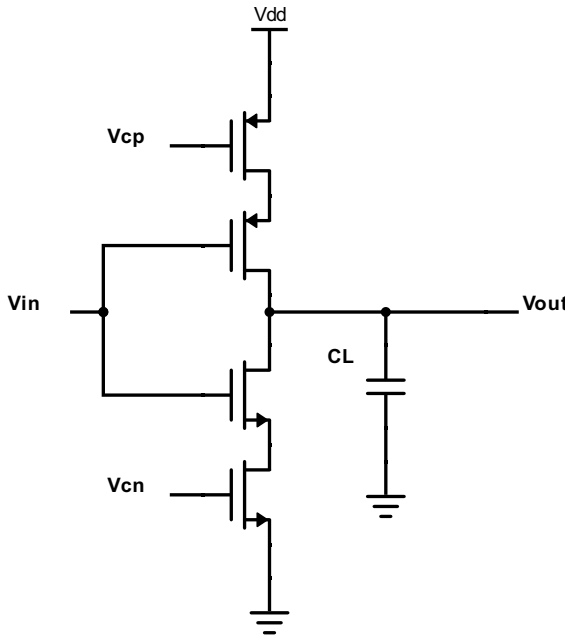


Figure 4.6: Typical implementation of the current starved inverter architecture

charge current is initially constant, the voltage across the capacitor, V_{out}, decreases linearly.

Thus, for the ideal case

$$I_{cp} = -C_L \frac{dV}{dt} \tag{4.3}$$

$$\int_0^{T_d} dt = \frac{C}{I_{cp}} \int_{\frac{V_{DD}}{2}}^{V_{DD}} dV \tag{4.4}$$

$$T_d = \frac{C_L}{I_{cp}} \frac{V_{DD}}{2} \tag{4.5}$$

In reality, however, the value of V_{out} does not start from V_{DD}, but from $V_{DD} + \delta V$, where δV is the contribution due to charge injection due to the parasitic capacitances between the gates and drains of the transistors forming the inverter. This therefore implies that the expression of the time delay consists of two parts; the time it takes for the output voltage to reach V_{DD} from $V_{DD} + \delta V$, and the time

to discharge from V_{DD} to $\frac{V_{DD}}{2}$, as shown in equation 4.6.

$$T_d = \frac{C_L}{I_{cp}} \left(\frac{V_{DD}}{2} + \delta V \right) \tag{4.6}$$

The value δV may be obtained from

$$\delta V = \frac{C_p}{C_L} dV_{in} \tag{4.7}$$

where C_p is the total parasitic capacitance between gate and drain of the two inverter transistors, and dV_{in} is the change in input voltage until it reaches its final value.

To overcome the linearity problem, the work in [64] presents a low power linear delay element circuit, shown in Figure 4.7. The circuit is based on a current-starved inverter architecture but can obtain both an improved linearity in the delay response and rail-to-rail operation. This is achieved through the addition of transistors $M_4 - M_8$. If the tuning voltage, V_{in}, is applied directly through transistor M_6, the delay response of the circuit would be highly non-linear and non-monotonic. For this reason, an inverting common-source amplifier is added consisting of M_7 and M_8 and this helps in achieving a monotonic and quasi-linear relationship in the delay response of the circuit.

The current, i_2, through transistor M_2 determines the delay time of the circuit, and it consists of the currents i_3 and i_4. When $V_{in} < V_{tn}$, M_3 works in the sub-threshold region, while M_6 works in saturation. The delay in this case is therefore primarily dependent on the current through M_6. For values of V_{in} in the region $V_{tn} < V_{in} < V_{DD} - V_{tp}$, both M_3 and M_6 are on and a linear delay-voltage characteristic may be achieved. When $V_{in} > V_{DD} - V_{tp}$, M_7 is switched off and thus the current through M_6 would be saturated. In this case, the delay would depend on the M_3. Transistor sizing optimisation was performed by means of a parameteric sweep, in order to

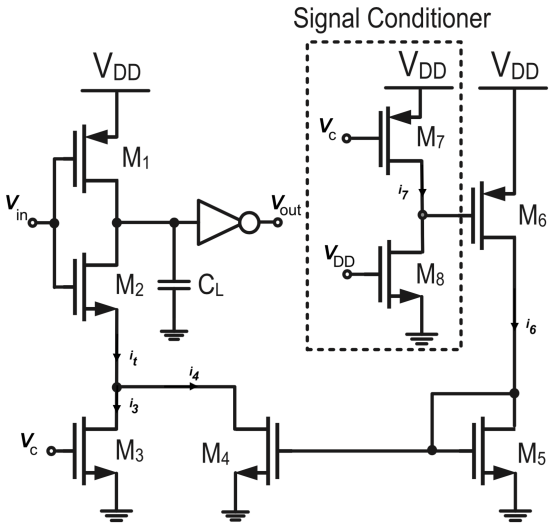


Figure 4.7: Linear delay element circuit [64]

obtain the most linear voltage-delay characteristic, particularly when both M_3 and M_6 are turned on [64].

In particular, the sizing of M_3 and M_4, were chosen to be as small as possible such that the maximum voltage-to-delay gain is achieved. The size of M_5 and M_6 was minimised to reduce the current, and the sizes of the transistors of the inverting common-source amplifier (M_7 and M_8) were chosen specifically to maximise the gain without affecting the linearity. As such, M_7 is chosen as large and M_8 as small. However, if M_8 is chosen too small, there would be a large voltage swing on the gate of M_6 and this will have a negative effect on linearity [64].

4.5.1 Preliminary Analysis and Verification

An analytical model for the current of the delay element shown in Figure 4.7 is presented here. This model is valid for the range $V_{tn} \leq V_{in} \leq V_{dd} - V_{tp}$. Both a first-order and a second-order model are presented in this section. In addition, an approximation method is used in order to linearise the delay transfer characteristic of the circuit. Let i_3 be the current through M_3, i_4 be the current through M_4, and

so on. The total current, i_t, is equal to:

$$i_t = i_3 + i_4 \tag{4.8}$$

$$= \frac{K'_n}{2}\frac{W_3}{L_3}(V_{in} - V_{tn})^2 + \frac{\frac{W_4}{L_4}}{\frac{W_5}{L_5}} i_5 \tag{4.9}$$

The current $i_6 = i_5$ is equal to

$$i_6 = \frac{K'_p}{2}\frac{W_6}{L_6}(V_{DD} - V_x - V_{tp})^2 \tag{4.10}$$

where

$$V_x = i_7 r_{ds8} = \frac{K'_p}{2}\frac{W_7}{L_7}(V_{DD} - V_{in} - V_{tp})^2 r_{ds8} \tag{4.11}$$

and r_{ds8} is the drain to source resistance for M_8. To simplify some terms,

$$\mu_1 = \frac{K'_n}{2}\frac{W_3}{L_3} \tag{4.12}$$

$$\mu_2 = \frac{\frac{W_4}{L_4}}{\frac{W_5}{L_5}}\frac{K'_p}{2}\frac{W_6}{L_6} \tag{4.13}$$

This means that i_t is equivalent to:

$$i_t = \mu_1(V_{in} - V_{tn})^2 + \mu_2(V_{DD} - V_x - V_{tp})^2 \tag{4.14}$$

Equation 4.14 may be expanded to a fourth order polynomial and therefore the current equation may be expressed as:

$$i_t = \alpha_4 V_{in}^4 + \alpha_3 V_{in}^3 + \alpha_2 V_{in}^2 + \alpha_1 V_{in} + \alpha_0 \tag{4.15}$$

where $\alpha_4, \alpha_3, \alpha_2, \alpha_1, \alpha_0$ are functions of the transistor aspect ratios $\frac{W}{L}$, the process parameters K'_p and K'_n, the supply voltage and the threshold voltages of the NMOS

and PMOS transistors, V_{tn} and V_{tp}, respectively. This implies that the delay would be equal to

$$T_d \propto \frac{1}{i_t} = \frac{1}{\alpha_4 V_{in}^4 + \alpha_3 V_{in}^3 + \alpha_2 V_{in}^2 + \alpha_1 V_{in} + \alpha_0} \tag{4.16}$$

The above equation assumes that r_{ds8} has a constant value. In reality, this is not true since it depends on the current i_8 and the drain-source voltage $V_{ds8} = V_x$. Thus a better model of V_x should be taken. Since the current through M_7 is equal to the current through M_8, then

$$i_7 = \frac{K'_p}{2}\frac{W_7}{L_7}(V_{dd} - V_{in} - V_{tp})^2 \tag{4.17}$$

$$= K'_n \frac{W_8}{L_8}((V_{dd} - V_{tn})V_x - \frac{V_x^2}{2}) \tag{4.18}$$

From equation 4.17 and equation 4.18, V_x, can be found.

$$V_x = V_N - \sqrt{(V_N)^2 - 2\frac{K_p}{K_n}(V_{dd} - V_{in} - V_{tp})^2} \tag{4.19}$$

where $V_N = V_{dd} - V_{tn}$, $K_p = K'_p \frac{W_7}{L_7}$ and $K_n = K'_n \frac{W_8}{L_8}$. Equation 4.16 shows that the time delay that can be achieved is a highly non-linear function. However linearity may be increased through the choice of values of the coefficients of V_{in}, $\alpha_4, \alpha_3, \alpha_2, \alpha_1$, and α_0.

Since the equation is an inverse fourth order polynomial, this is not a trivial task. As such, an approximation technique was taken to transform equation 4.16 into a simple second order polynomial equation. This equation allows for two degrees of freedom in terms of the coefficients of V_{in} and V_{in}^2. The former would enable to control the range, while the latter would permit to control the linearity. As such, the objective is to minimise as much possible the coefficient of V_{in}^2.

The Lagrange Polynomial approximation [65] was used to transform equation 4.16

into a second order polynomial equation. The Lagrange Polynomial has the form

$$f(V_{in}) = T_d(V_0)\frac{(V_{in}-V_1)(V_{in}-V_2)}{(V_0-V_1)(V_0-V_2)} + T_d(V_1)\frac{(V_{in}-V_0)(V_{in}-V_2)}{(V_1-V_0)(V_1-V_2)} + T_d(V_2)\frac{(V_{in}-V_0)(V_{in}-V_1)}{(V_2-V_0)(V_2-V_1)} \quad (4.20)$$

where $T_d(V_{0-2})$ is equation 4.16 evaluated at $V_0 = V_{tn}, V_2 = V_{DD} - V_{tp}$, and $V_1 = 0.5 * (V_0 + V_2)$.

Thus, equation 4.16 may be rewritten in the form of:

$$T_d \approx AV_{in}^2 + BV_{in} + C \quad (4.21)$$

where the coefficients A, B, and C can be directly related to the parameters of the transistors. A closer examination of the above coefficients shows that the coefficient A, can be minimised through an increase in the dimensions of transistor M_5, a decrease in the sizing of M_3, and an increase in r_{ds8} (decrease in $\frac{W_8}{L_8}$). This however will have an effect on the second coefficient, B. When the aforementioned parameters are changed, the gradient (and therefore the range) changes significantly. This is due to the fact that the denominator of the three coefficients is a function of the transistor ratios $M_3 - M_8$. This therefore implies that in this architecture there is a trade-off between delay range, resolution, and linearity.

The circuit illustrated in Figure 4.7 was implemented and simulated in Cadence using the 0.18 μm X-FAB technology, with the transistor aspect ratios, as implemented in [64] (refer to Table 4.2). A scaling factor was added to the dimensions such that any channel-length modulation effects are minimised. In addition, each individual transistor was characterised to find the process parameters K'_n and K'_p together with the transistors' respective threshold voltages.

An input pulse was applied to the CLK terminal while varying the tuning voltage, V_{in}. The delay was calculated by measuring the time difference between the input

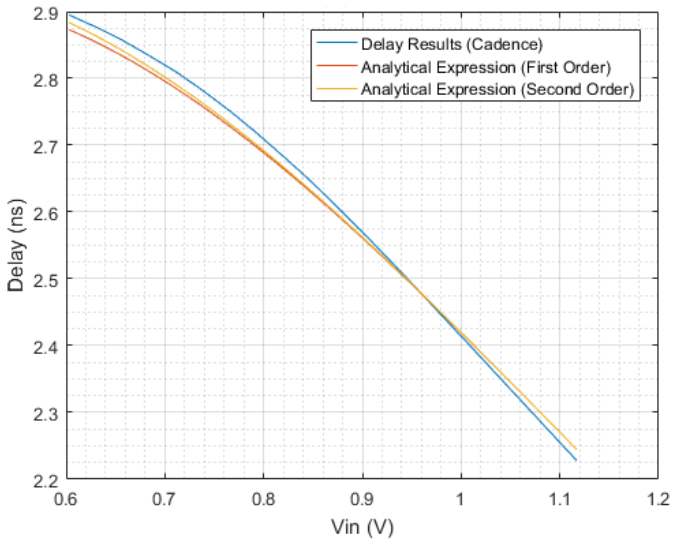


Figure 4.8: Comparison of the simulated results with those obtained from the first order and second order analytical models

Table 4.2: Transistor Aspect Ratios

Transistor Name	Width (μm)	Length (μm)
M1	6.9	0.18
M2	3.5	0.18
M3	0.225	0.25
M4	0.69	0.18
M5	2.08	0.18
M6	2.21	0.18
M7	3.18	0.18
M8	1.94	0.18

and output waveforms, when the voltage reaches 50% of the final value. A MATLAB script was written in order to test the analytical model. The script contains the equations for the currents in each branch of the circuit. Through a DC analysis, with an input voltage of 1 V, it was found that there is a good relationship between the values, thus showing that the equations describing the current are correct. A voltage sweep on the tuning voltage, in the range V_{tn} to $V_{DD} - V_{tp}$ was then performed and the delay results are illustrated in Figure 4.8, where the delay is shown in blue. The first order analysis is illustrated in orange, and the improved analysis is illustrated in yellow. It can be seen that there is a good relationship between the analytical results and the simulated delay. There is, however, a discrepancy between the analytical results and the obtained result. This can be attributed to the parasitic effects.

The Lagrangian interpolation method was used to approximate the delay curve by a second-order polynomial. The results are shown in Figure 4.9, where the simulated delay results from Cadence are plotted in blue, and the approximation is plotted in orange. The error is also plotted in the subplot, where the error illustrated reaches a maximum of 0.94%. This therefore shows that the proposed model can be used to effectively approximate the delay transfer characteristic with a second-order polynomial. To this end, different values for the transistors aspect ratios can be modified to improve linearity and range. As stated previously, the coefficient of the term in V_{in}^2, can be minimised through an increase in r_{ds8}, an increase in the sizing of $M3$, or a decrease in the sizing of M_5. The aspect ratio of M_8 was decreased to 700 nm/180 nm, which yields an r_{ds} value of around 1.7 kΩ. A sweep on the dimensions of M_3 and M_5 was then performed with the coefficient in V_{in}^2 worked out on each iteration. The results are presented in Figure 4.10. From this plot, the values of the aspect ratios of M_3 and M_5, that yields the lowest value of the coefficient of V_{in}^2 can be selected. These aspect ratios were calculated to be 0.9 and 20.8, respectively.

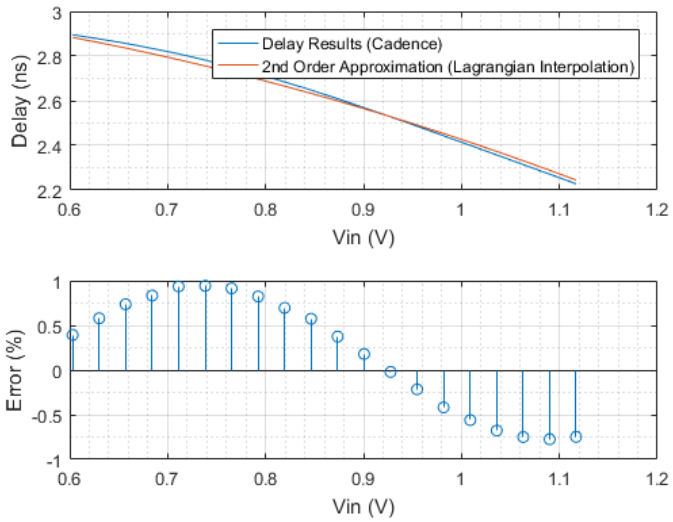


Figure 4.9: Comparison of the simulated delay variation with that obtained from a second order Lagrangian polynomial

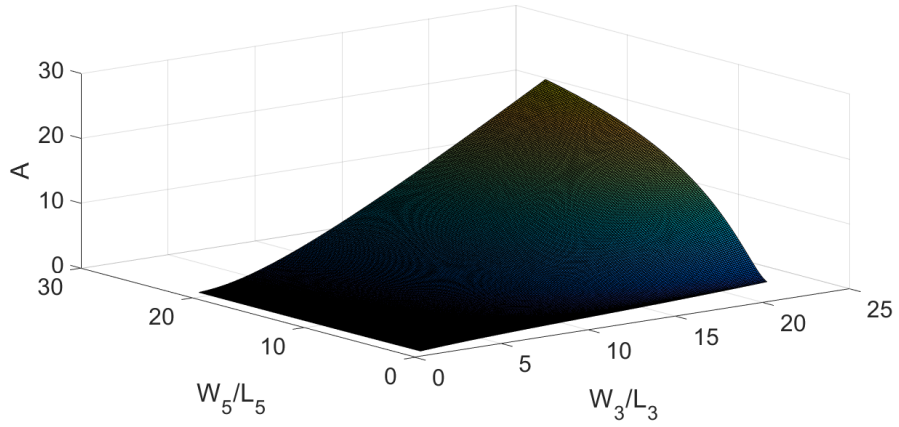


Figure 4.10: Variation of the quadratic coefficient A with the aspect ratio of transistors M_3 and M_5

Transient simulations were performed in order to obtain the variation of the delay with the input tuning voltage for different supply voltages. The results are reported in Figure 4.11 and are compared to those estimated using the Lagrangian Interpolation analytical model. There is good agreement between the plots, which shows that the Lagrangian Interpolation for equation 4.21 can be successfully used to set the transistor aspect ratios in order to obtain the most linear response.

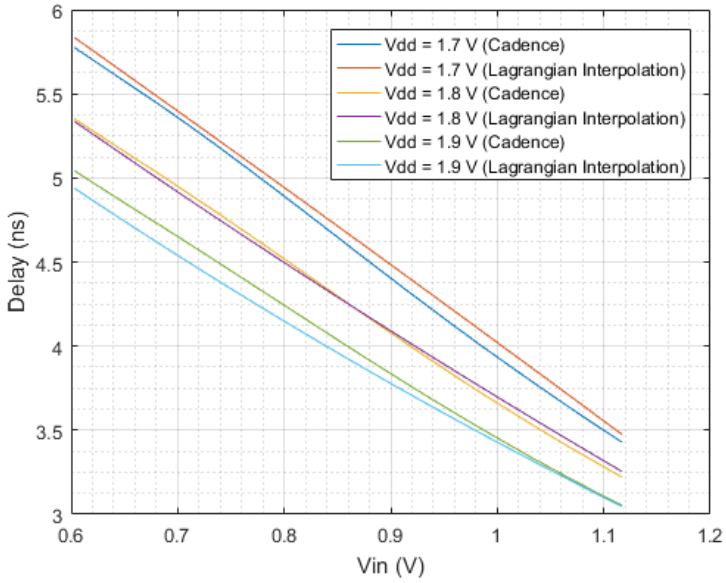


Figure 4.11: Plot of the delay versus the input tuning voltage for the optimised linear delay element circuit

4.6 Improved Delay Element and Delay Model

The previous section has shown that for a control voltage range $V_{tn} < V_c < V_{DD} - V_{tp}$, a linear delay transfer characteristic may be achieved by setting the aspect ratios of transistors M_3, M_5, and M_8 to 0.9, 20.8, and 3.9, respectively. Nonetheless, since the analytical model, used for the design and optimisation, is not completely valid for rail-to-rail operation, simulations show that the linearity worsens for $V_c < V_t$. Furthermore, this circuit is able to delay only the falling edge of the input signal, thus limiting the delay generated by the pulse width of the input.

To overcome this issue, the circuit illustrated in Figure 4.12 is being proposed. Transistors $M_2 - M_7$ form the delay element [64], which is controlled by the control voltage V_c. The current from this architecture is then mirrored to the balanced current-starved inverter, formed by transistors $M_{10} - M_{13}$. This results in a delay element that is capable of rail-to-rail operation with a quasi-linear response. The delay can be finely tuned via V_c and a coarse control is provided via switches EN_1 and EN_2, which increases the effective capacitive load at the output node, thereby

extending the programmable range. In addition, this circuit also enables an increase in the delay range via proper scaling of the current mirror ratios M_{10}/M_1 and M_{13}/M_8. This is particularly useful for limiting the size of the on-chip capacitors. For a symmetrical current-starved inverter, the charging and discharging current should be equal. This means that $i_{charge} = \frac{R10}{R1} i_1$ must be equal to $i_{discharge} = \frac{R13}{R8} i_1$. The aspect ratios of M_{11} and M_{12} should be as large as possible, in order for the charging and discharging currents to be fully controlled by M_{10} and M_{13}.

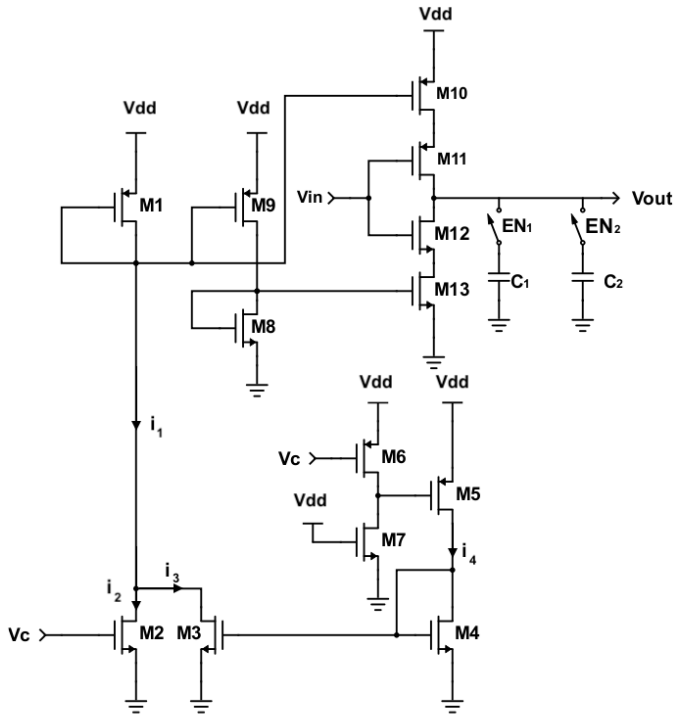


Figure 4.12: Improved linear delay element circuit with extended programmable range and symmetric operation

The time-delay response of the circuit is therefore a piece-wise equation given by equation 4.22.

$$T_d \propto \begin{cases} \frac{1}{i_3} & V_c < V_{tn} \\ \frac{1}{i_2 + i_3} & V_c \geq V_{tn} \end{cases} \tag{4.22}$$

Assuming that M_2 and M_3 remain in pinch-off, the piece-wise expression for the current in M_1, which is inversely proportional to the delay, is given by:

$$i_1 = \begin{cases} i_3 & V_c < V_{tn} \\ i_2 + i_3 & V_c \geq V_{tn} \end{cases} \tag{4.23}$$

where i_x is the current through transistor M_x and x is the transistor identifier. When M_2 and M_3 operate in pinch-off, the currents are given by:

$$i_2 = \frac{K'_n}{2}\frac{W_2}{L_2}(V_c - V_{tn})^2 \tag{4.24}$$

$$i_3 = \frac{\frac{W_3}{L_3}}{\frac{W_4}{L_4}} i_4 \tag{4.25}$$

$i_4 = i_5$ is given by

$$i_4 = \frac{K'_p}{2}\frac{W_5}{L_5}(V_{sg5} - V_{tp})^2 \tag{4.26}$$

where

$$V_{sg5} = V_N - \sqrt{(V_N)^2 - 2\frac{K_p}{K_n}(V_{DD} - V_c - V_{tp})^2} \tag{4.27}$$

and $V_N = V_{DD} - V_{tn}$, $K_p = K'_p\frac{W_6}{L_6}$ and $K_n = K'_n\frac{W_7}{L_7}$. This means that equation 4.23 may be represented as:

$$i_1 = \begin{cases} \alpha_3 V_c^2 + \alpha_2 V_c + \alpha_1 V_c^{1/2} + \alpha_0 & V_c < V_{tn} \\ \alpha_7 V_c^2 + \alpha_6 V_c + \alpha_5 V_c^{1/2} + \alpha_4 & V_c \geq V_{tn} \end{cases} \tag{4.28}$$

where parameters α_0 to α_7 are functions of the transistor aspect ratios, process parameters K'_n and K'_p, the supply voltage, and the threshold voltages of the NMOS and PMOS transistors. The delay that can be generated may still be modelled as a piece-wise equation of equation 4.6. The relationships involved are highly complex, making it difficult to design the delay cell by means of a closed form equation.

To be able to simplify this equation, a second approximation technique based on the Newton Polynomial method [66] was employed. This technique allows its transformation into a piece-wise second-order polynomial with good accuracy. Therefore, the time delay equation can now be approximated by:

$$T_d \approx \begin{cases} A_1 V_c^2 + B_1 V_c + C_1 & V_c < V_{tn} \\ A_2 V_c^2 + B_2 V_c + C_2 & V_c \geq V_{tn} \end{cases} \tag{4.29}$$

where A_x, B_x, C_x are model parameters depending on the transistor aspect ratios, process parameters, supply voltage, and the threshold voltages. A closer examination of the coefficients in equation 4.29 show that the linearity depends mainly on the aspect ratios of transistors M_6 and M_7 while the range depends on the aspect ratio of transistors M_3, M_4 and M_5. Once the desired range has been obtained, the aspect ratio of M_2 was chosen, such that its effect would not limit the linearity to the range $V_{tn} \leq V_c \leq V_{DD}$. This method results in a quasi-linear relationship between the delay and control voltage while ensuring the desired range. With this polynomial, the aspect ratios of the transistors may be optimised to obtain a quasi-linear response without requiring lengthy parametric simulations.

4.6.1 Verification of Model and Simulations

The circuit shown in Figure 4.12 was implemented and simulated in Cadence using the 0.18 µm X-FAB technology. The optimised transistor aspect ratios are presented in Table 4.3. To limit channel length modulation effects, the minimum length size was set to 500 nm. A 1 pF double metal-insulator-metal capacitor (cdmm4) was used and covers an area of 20×20 µm^2. This capacitor was chosen because it provides the largest capacitance per unit area (2 fF/µm^2) and it has a small voltage coefficient of 3 ppmV^{-1}.

For the analytical model to take into consideration any effects related to transis-

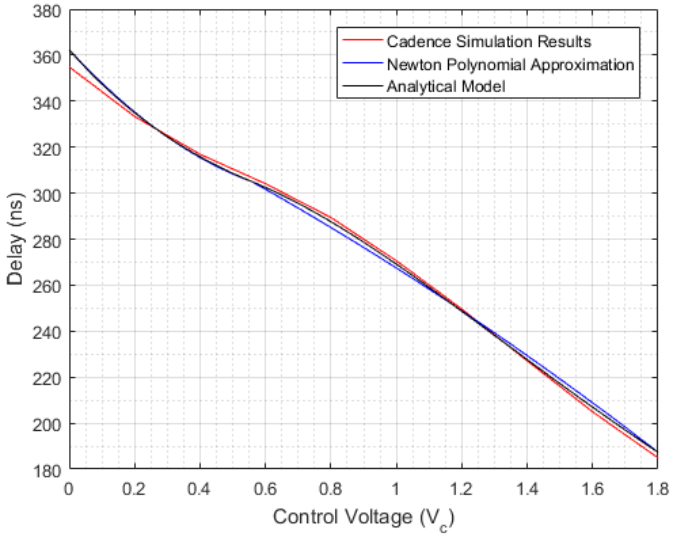


Figure 4.13: Comparison of the simulated results with those obtained from the analytical model and the Newton Polynomial approximation method

tor mismatches, each transistor was characterised in order to obtain the different parameters K_n' and K_p' together with the respective threshold voltages. The delay was calculated by applying a pulse to the V_{in} input, and calculating the time it takes for the output to reach $0.5V_{dd}$, for different control voltages V_c. The Newton Polynomial method was used to approximate the delay equation by two piece-wise second-order polynomials. The results are illustrated in Figure 4.13, where the analytical model is plotted in black, the approximation model is plotted in blue and the results obtained from the simulator in red. It can be seen that through this method, a good approximation may be obtained, through which the transistor aspect ratios may be found in order to obtain the most linear delay. Since a piece-wise second-order polynomial describes the response of the delay element, the coefficient of V_c^2 can be minimised for the case when $V_c < V_{tn}$, where there is no dependency on the aspect ratio of M_2. M_2 can then be chosen such that its effect for the case when $V_{tn} \leq V_c \leq V_{DD}$, is minimised as much as possible. This will ensure that the delay response remains quasi-linear.

To further test the linearity of the circuit, a sinusoidal input, with a frequency

Table 4.3: Transistor aspect ratios of the linear delay generator.

Transistor Name	Aspect Ratio
M_2	0.1
M_3	12
M_4	12
M_5	1
M_6	5.9
M_7	4.85

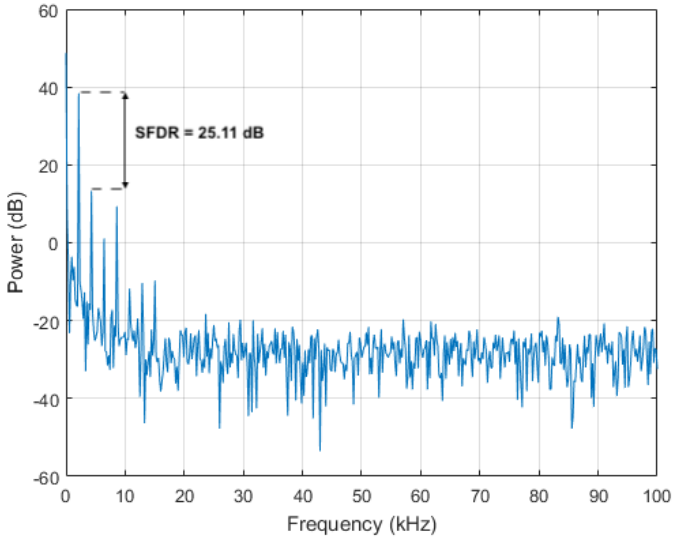


Figure 4.14: Frequency spectrum of the simulated time delay signal

of 2.148 kHz was applied to V_c, and an input square wave of 200 kHz was applied to V_{in}, with a sampling frequency of 2 GHz, and a simulation time of 5.12 ms. The delay between the input and the output was then calculated via a MATLAB script. The spectrum of the delay is plotted in Figure 4.14. The Spurious-Free Dynamic Range (SFDR) of the delay is equal to 25.11 dB, and the Signal-to-Noise-and-Distortion Ratio (SNDR) is equal to 23.04 dB. Although these values are less than those achieved in [64], the proposed delay element circuit has a wider delay range (168 ns in this work compared to 1.4 ns in [64]).

To confirm that the delay element with extended range works as expected, a 500 fF capacitor, and a 1 pF capacitor, were used. Each of the capacitors was connected to a transmission gate to be able to include or exclude them from the circuit. This

allows for an increase in the delay range. The results are presented in Figure 4.15 for the different combinations of the capacitors. As expected, since the capacitance is multiplied by the value of the inverse of the current, then both the gradient and the y-intercept will change. This is because the value of the capacitance is multiplied by the inverse of the current, which approximates a straight line.

Temperature analysis and Monte Carlo analysis were carried out to test the robustness of the delay element for C_L =1 pF. The results from the temperature analysis are presented in Figure 4.16, where it can be observed that the delay varies with the operating temperature. Linearity suffers at temperatures in the range of $-40\,^{\circ}\mathrm{C}$ to $-20\,^{\circ}\mathrm{C}$, particularly near the threshold voltage of M_2. Monte Carlo simulations were performed for process and mismatch variations with 200 points for all values of V_c. The results show that for $V_c = 0\,\mathrm{V}$ the mean delay between the rising edge at the input and the falling edge at the output has a mean delay of 374.3 ns with a standard deviation of 51.61 ns. On the other hand, the mean delay between the falling edge at the input and the rising edge at the output has a mean delay of 370.3 ns with a standard deviation of 38.8 ns. This shows that there is a slight discrepancy between the charge and discharge paths of the current due to process variations and that the process variations affect more the NMOS side than the PMOS side. This is further evident when the results from mismatch contributions are taken into consideration. In fact, the largest error contribution comes from transistors M_3, and M_{13}, at 39% each. M_8 and process variations are the next largest contributors at 35% each.

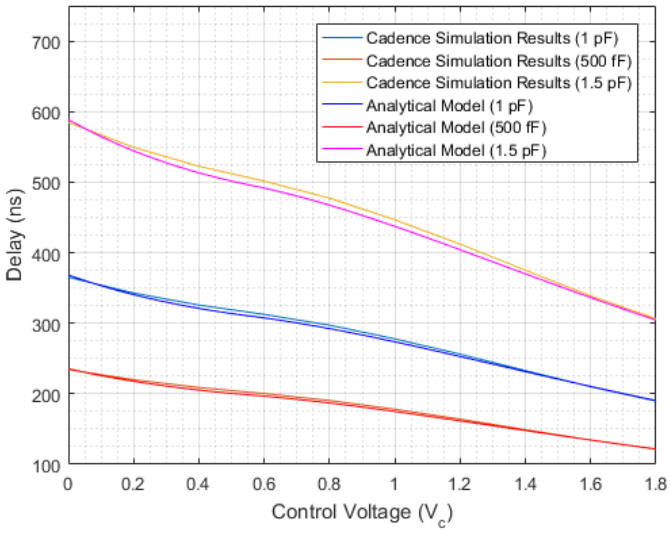


Figure 4.15: Extended delay range achieved via the proposed programmable banked capacitor architecture

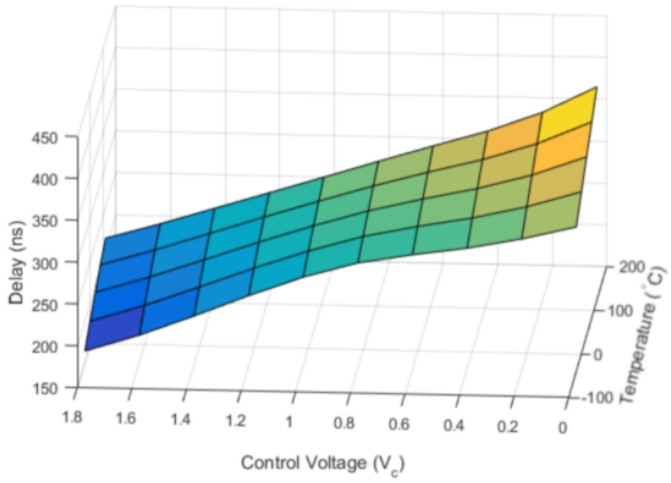


Figure 4.16: Simulated delay variation with control voltage across a temperature sweep from $-40\,^{\circ}\mathrm{C}$ to $120\,^{\circ}\mathrm{C}$

4.7 Further Optimisation

There are two methods in which optimisation can take place; either simulation-based, or equation-based. The former performs optimisation on circuit parameters based on the simulation results. It takes care of any parasitic components, if modelled, and can be used for any type of circuit with minimum setup. Unfortunately this method is more computationally expensive, as with each iteration the simulation needs to be re-evaluated. The second method is the equation-based approach. While the equations need to be derived for each type of circuit topology, this is done only once and the equation can be optimised using numerical solvers.

Optimisation algorithms in integrated circuit design may be split into three main categories: Bio-inspired optimisation which encompass evolutionary algorithms and swarm intelligence algorithms, deterministic algorithms, and other optimisation techniques that do not fit in the previous two categories such as simulated annealing, convex optimisation, and greedy algorithms [67].

There are two most commonly used techniques for optimisation of analogue circuits through bio-inspired techniques; Particle Swarm Optimisation (PSO), and Genetic Algorithm (GA). The PSO algorithm involves the use of an initial swarm (a random set of generated particles) which move in the design search space towards the required optimal solution. The information is shared between each member of the swarm. The GA, on the other hand, is built upon the principle of "survival of the fittest" and solutions are generated based upon experience and environment [68].

The PSO and GA are similar in the sense that they are both population-based search approaches and utilise information sharing between particles. In [68] it was shown that the PSO is more computationally efficient when compared to the GA, even though the former yields similar results to the GA approach. Thus, the PSO is a good algorithm to be able to optimise the aspect ratios of the transistors in a circuit.

A flowchart of the PSO algorithm is depicted in Figure 4.17. Initially, the basic swarm is created, consisting of N particles, in the multi-dimensional search space, with D dimensions, where each dimension represents a variable in the circuit model. Each particle is characterised by a position and velocity. Once the particles have been created, and the iteration number set to 1, the objective function is calculated for each particle, and the fitness value of each particle is compared with the best particle value. On each iteration, the position and velocity of each particle are updated according to its own best position, P_{Best}, and the best position in the entire swarm G_{Best}. The iteration number is incremented and the objective function is calculated once again. This continues until the termination criteria (e.g. maximum number of iterations) is met [69].

4.7.1 Improving the Linearity of the Delay Element

There are two methods that can be used to improve the linearity of the rail-to-rail delay element. The first method is to approximate the piecewise equation of equation 2.17 by using either the Lagrange Polynomial or Newton Polynomial methods, as described in [50] and [51], respectively. While these methods can achieve good linearity, the process is cumbersome as this is done manually, while introducing a certain error due to the nature of the approximation techniques used. The second method involves the use of bio-inspired numerical optimisation techniques to find automatically the values of the aspect ratios of the transistors which yield the highest linearity.

4.7.2 Objective Function

To find a measure of the linearity of the time delay equation, the Mean-Square Error (MSE) can be considered. The MSE, σ is given by:

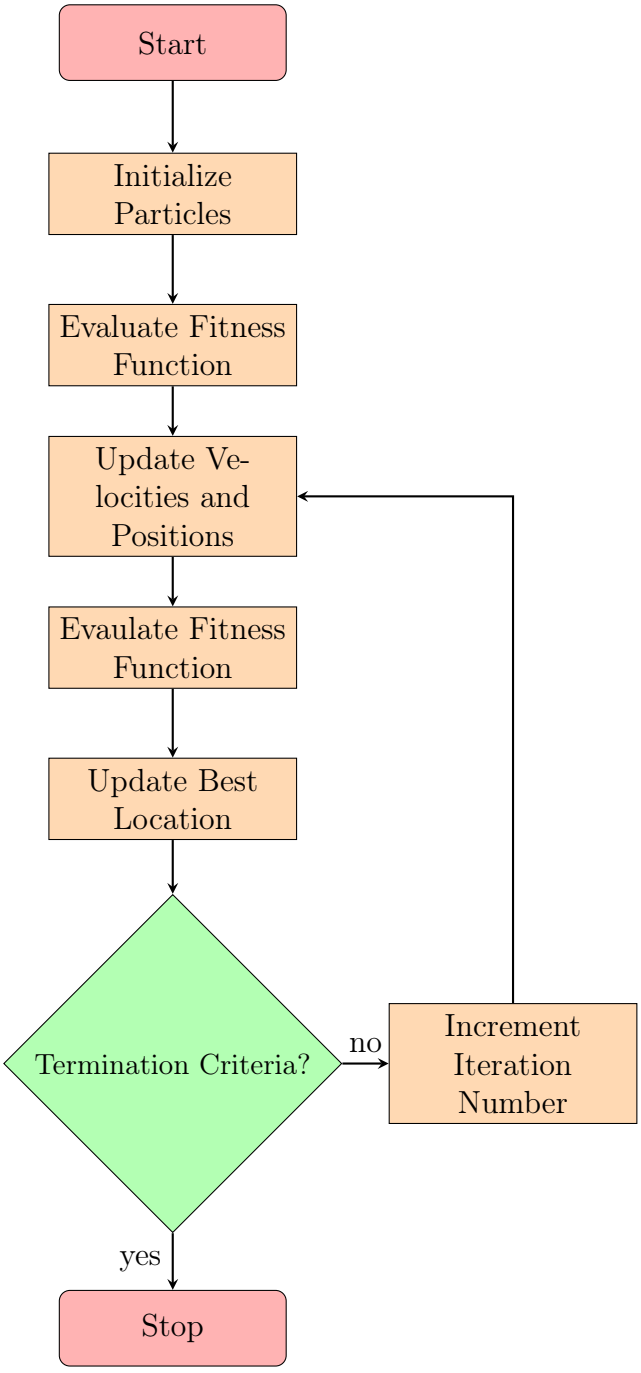


Figure 4.17: Flowchart of the PSO algorithm

$$\sigma = \frac{1}{n}\sum_{i=1}^{n}(T_{d_i} - \hat{T}_{di})^2 \tag{4.30}$$

where n is the number of samples considered, T_d is the time delay of the circuit according to the model, and $\hat{T}_d$ is the perfect linear time delay, given by equation 4.31, where R is the required resolution, and O is the offset. The gradient is negative since the delay is inversely proportional to the current, and hence inversely proportional to V_c.

$$\hat{T}_d = -RV_c + O \tag{4.31}$$

The MSE is the objective function of the optimisation algorithm. In other words, to obtain the most linear delay, the MSE needs to be minimised. For the HMPID application considered in this work, the required delay element should have a resolution of $-90\,\text{ns/V}$ with an offset (delay value at $V_c = 0\,\text{V}$) of around $360\,\text{ns}$.

4.7.3 Variables and their Constraints

In this case, the variables are the aspect ratios of transistors $M_2 - M_7$. To ensure that the values of the sizes of the transistors can be realised, the constraints were chosen specifically to keep the area to a minimum. As such, the lower bound values of the aspect ratio was set to 0.1, while the upper bound value was set to 30.

4.7.4 Implementation of the PSO Algorithm

The PSO algorithm was implemented in MATLAB. Three functions were created for proper code re-usability and readability. The main function consists of the primary PSO algorithm, while the other two functions evaluate the delay model, and the ideal linear delay model. The former takes as its parameters the aspect ratios of the transistors and the control voltage vector, and it works out the time-delay

model according to equation 4.22. The latter takes as its parameters the control voltage vector, and the required gradient and offset, and evaluates equation 4.31.

In the main function, the parameters of the PSO algorithm are defined. Specifically these parameters are the swarm size, N, and the number of dimensions (variables), D. In this scenario, there are six unknown parameters which are the aspect ratios of transistors $M_2 - M_7$. The maximum and minimum inertia weights, w_{max} and w_{min} respectively, are set in this part of the code. The inertia weights are multipliers for the current velocity of the particles such that the new positions may be updated.

To obtain the most reliable results, the main program runs for a number of times. As such, the main function consists of two nested loops, where the first loop takes care of the run, and the second loop is the principal iteration of the PSO algorithm.

The position of the particles is first initialised through equation 4.32, where UB(j) and LB(j) denote the upper bounds and lower bounds of each dimension, j, and r is a random number generated between 0 and 1. The initial velocity of the particles is set by multiplying the initial position by a factor of 0.1. The code is illustrated in Listing 4.2. After the initial particles are set, the delay is worked out according to equation 4.22. There may be certain cases where the model would yield a complex number due to the evaluation of the square root in equation 4.27. In such cases, the script performs a check and if any values are imaginary, they are changed to infinity. This will ensure that any imaginary values will not affect the results, as such values are essentially discarded. The values of the particles are passed to the model function, such that the delay model for each particle can be calculated, and the MSE for each particle is evaluated. For each iteration, the minimum MSE is found and its value saved, together with the initial best particle and best global particle.

$$X_{i,j} = \text{Round}(\text{LB}(j) + r \times (\text{UB}(j) \text{ - } \text{LB}(j))) \tag{4.32}$$

Listing 4.2: Initialisation of Particles

```
for i = 1:n
    for j = 1:m
        x0(i,j) = round(LB(j) + rand()*(UB(j)-LB(j)));
        if x0(i,j) < LB(1,j)
            x0(i,j) = LB(1,j);
        elseif x0(i,j) > UB(1,j)
            x0(i,j) = UB(1,j);
        end
    end
end
x = x0; % Initial Population
v = velocity*x0; % Initial Velocity

fprintf('Delay for every variable\n');

for i = 1:n
    % Work out delay for every variable
     Td(i,:) = Td_model(Vc,x0(i,1),x0(i,2),x0(i,3),x0(i,4),x0(
        i,5),x0(i,6)).*A*cap*prop*Vdd;
    if(isreal(Td(i,:)))
        f0(i,1) = mse_calc(Td_req,Td(i,:));
    else f0(i,1) = inf;
    end
end

% Find the minimum value for f0
fprintf('Finding Minimum\n');
[fmin0,index0] = min(f0);
pbest = x0; % Pbest
gbest = x0(index0,:); %gbest
```

The first step in the inner-loop is to update the inertial weight, w, and then calculate the new velocities vector, V, of the particles according to equation 4.33,

and the position vector, X, of each particle according to equation 4.34. In these equations, k denotes the iteration number, and i represents the particle number. Parameters $r_{1_{i,j}}, r_{2_{i,j}}$ refer to a random number generated in the range 0 to 1, while c_1, c_2 are acceleration factors. This is illustrated in Listing 4.3.

$$V_{i,j}^{k+1} = w \times V_{i,j}^{k} + c_1 \times r_{1_{i,j}} \times (P_{Best_{i,j}}^{k} - X_{i,j}^{k}) + c_2 \times r_{2_{i,j}} \times (G_{Best_j}^{k} - X_{i,j}^{k}) \quad (4.33)$$

$$X_{i,j}^{k+1} = X_{i,j}^{k} + V_{i,j}^{k+1} \quad (4.34)$$

Listing 4.3: Updating the Inertial Weight and calculation of new velocities and position vectors

```
% Update inertia weight factor
        w = wmax - (wmax-wmin)*iter/maxiter;
        % Update Velocities and Positions
        for i = 1:n
            for j = 1:m
                v(i,j) = w*v(i,j)+c1*rand()*(pbest(i,j)-x(i,j)) +
                    c2*rand()*(gbest(1,j)-x(i,j));
                x(i,j) = x(i,j) + v(i,j);
            end
        end
```

With every iteration, the values of the particles are re-checked through the model function, and the MSE is re-calculated. The least MSE value for each particle (denoted by P_{Best}, for population best) and the global best for each swarm (G_{Best}) are stored separately. While the iteration number has not reached the maximum number of iterations, the inner loop restarts. Once the maximum number of iterations has been reached, the run counter of the outer loop is increased, and the algorithm is re-executed. With every iteration, a check is performed to ensure that each particle lies within the upper and lower bounds. When the program is complete, the best MSE value obtained is presented together with the relevant aspect

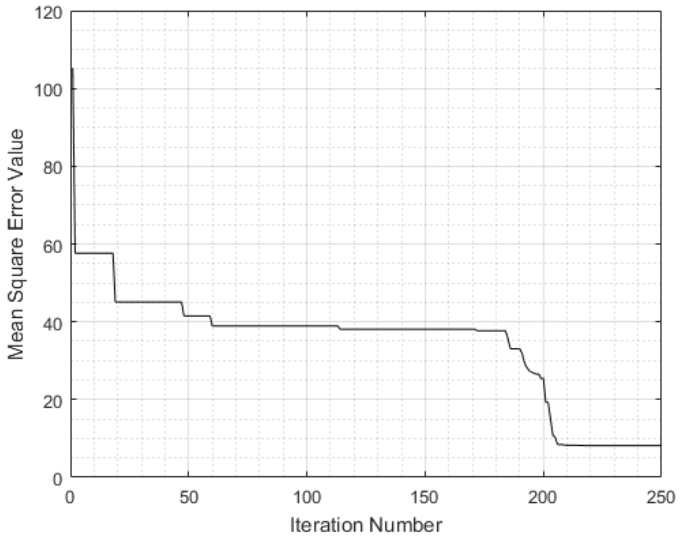


Figure 4.18: Convergence profile of the PSO algorithm for the best run

ratios of the transistors that yielded that result.

The PSO algorithm was executed and the optimal aspect ratios of the transistors were found. The algorithm was executed with six variables, and a swarm population of 100. The minimum and maximum inertia weights were set to 0.4 and 1.4, respectively. Acceleration factors c_1 and c_2 were set to 1.5 and 2, with an initial velocity of 0.1.

The time it takes the program to find the optimal solution depends on the swarm size, number of iterations, and number of runs. In this case, the algorithm was executed with 250 iterations and 100 independent runs, and the optimal solution (being the solution that yields the minimum MSE) took approximately 520 seconds, while running on a Core i7-4790 processor at 3.6 GHz, with 10 GB of memory. Figure 4.18 illustrates the convergence profile of the PSO algorithm for the best run.

The results are presented in Table 4.4. For the values obtained, the MSE is 8.16, while for the Newton Polynomial Approximation technique, presented in the previous section, this value was 12.8. This implies that an improvement in the linearity

Table 4.4: Aspect ratio of transistors

Transistor Identifier	**Aspect Ratio**
M_2	0.1
M_3	17
M_4	30
M_5	1.8
M_6	29.7
M_7	22.2

of the delay element was achieved.

The circuit illustrated in Figure 4.12 was implemented in Cadence, with the optimised transistor aspect ratios found using this algorithm. The delay versus control voltage characteristic obtained from Cadence is presented in Figure 4.19, together with the ideal delay response. It can be seen that there is a good correspondence between the analytical model and the simulation results obtained from Cadence, using a 1 pF load.

To verify the linearity of the delay element, a sinusoidal input was applied to the V_c terminal of the circuit, and the delay between the input and output square waves was calculated through another MATLAB script. The sinusoidal input has a frequency of 2.148 kHz, while the input square wave has a value of 200 kHz. The sampling frequency used was 2 GHz, with a simulation time of 5.125 ms.

The frequency spectrum of the generated delay is plotted in Figure 4.20. The SFDR) of the delay is equal to 29.62 dB. This implies that there is an improvement of 4.5 dB over the work presented in [51]. While this value is still less than that achieved by the authors in [64], where the SFDR was equal to 35 dB, the range of this delay element is much wider (170.4 ns), compared to 1.4 ns achieved in [64]. The simulated SNDR is 25.6 dB, an improvement of 2.56 dB over the work in [51].

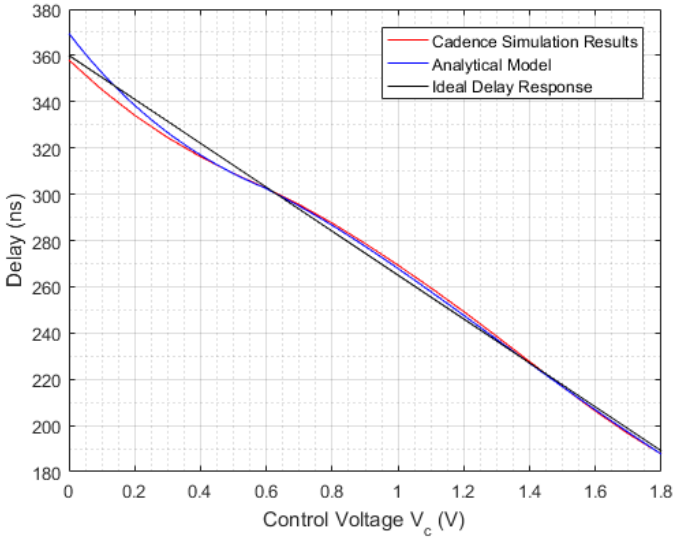


Figure 4.19: Comparison of the simulation results obtained from Cadence with the results generated from the analytical model and the ideal delay response

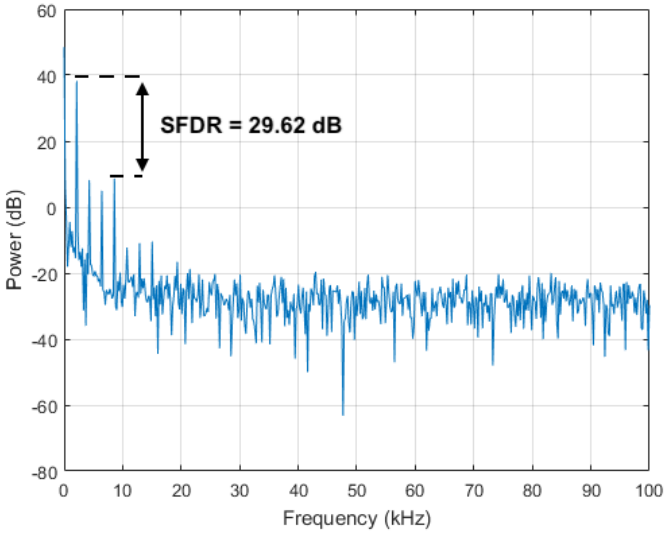


Figure 4.20: Frequency spectrum of the generated delay for $C_L = 1\,\text{pF}$

4.8 Final Delay Element Design

To ensure that the delay line is capable of handling short input pulses, five delay elements were used. This therefore implies that the maximum delay range of each delay element is shorter than that originally stated. As such, the maximum delay of the delay element was chosen to be 100 ns. The transistor aspect ratios were

optimised in order to achieve maximum linearity. It was previously seen that the Particle Swarm Optimisation algorithm can successfully improve linearity by optimising the aspect ratios of transistors $M_2 - M_7$. This was done by reducing the MSE between the time delay equation (equation 4.22) and that of an ideal straight line, leading to the most-linear delay characteristic. The optimised transistor aspect ratios are presented in Table 4.5. This technique has further improved the MSE to 4.54.

Table 4.5: Transistor aspect ratios of the final delay element circuit

Transistor Identifier	**Aspect Ratio**
M_2	0.2
M_3	1
M_4	13.3
M_5	11.7
M_6	5.3
M_7	3.3

To minimise any channel-length modulation effects transistor lengths of 0.5 µm were used for $M_3 - M_7$. These values give a delay element whose delay range varies from 31.59 ns to 83.84 ns, illustrated in Figure 4.21. There is a slight discrepancy between the theoretical results obtained using the values optimised with the Particle Swarm Optimisation, and the simulated values with Cadence. This is due to the fact that to increase the signal integrity, the delay line was loaded with two inverters. Together, these two inverters introduce both an additional load and an additional delay. This therefore has two effects on the delay.

If it were to be assumed that the delay is truly linear, then $T_d = AV_c + B$. The addition of the two inverters, increase the effective delay thereby adding an offset to the parameter B. However, these two inverters add a load to the delay, thus affecting not only B, but also A. This therefore reduces the SFDR, where even though the MSE was reduced, the additional load has in essence amplified the non-linearity. In fact, the SFDR has been reduced to 23 dB (Figure 4.22). The delay response of the delay element, with the contribution of the two inverters is

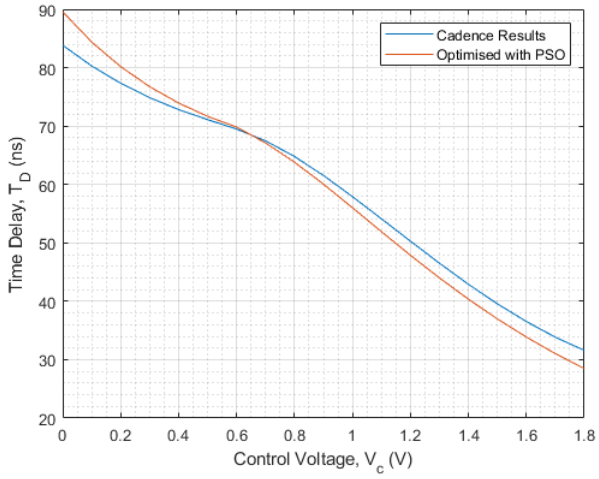


Figure 4.21: Delay response of the delay element before inverters

illustrated in Figure 4.23. This therefore implies that this modification gives a delay element whose delay range varies from 42.96 ns to 107.6 ns, for a load capacitance of 1 pF.

4.9 ADC and DAC

The analogue-to-digital and digital-to-analogue converters are integrated on chip. Both the ADC and DAC feature a 12-bit resolution for high precision and form part of the analogue library packages provided by the X-FAB technology. The successive-approximation ADC is based upon a 12-bit segmented capacitor DAC to reduce the input capacitance with a total area occupation of 512.165×524.825 μm^2. To start the conversion process, a strobe signal of 50 ns is required. During conversion stage, the ADC consumes 966 µA, at a clock frequency of 20 MHz and a bias current of 30 µA. Since the chosen ADC works with a differential input, the DLL control voltage, V_c, is fed through a single-ended to differential converter. A 12-bit DAC is used to convert the digital control voltage value, which is stored in the LUT, into an analogue value which is used by the VCDL to provide the required trigger signal delay in open loop. The chosen DAC occupies an area of

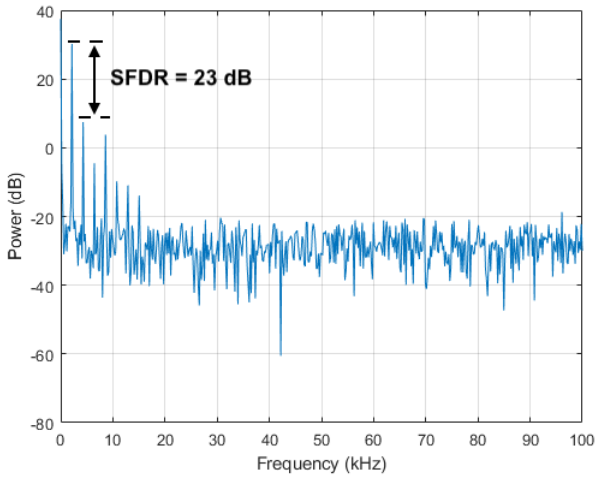


Figure 4.22: Frequency spectrum of the generated delay for $C_L = 1\,\text{pF}$

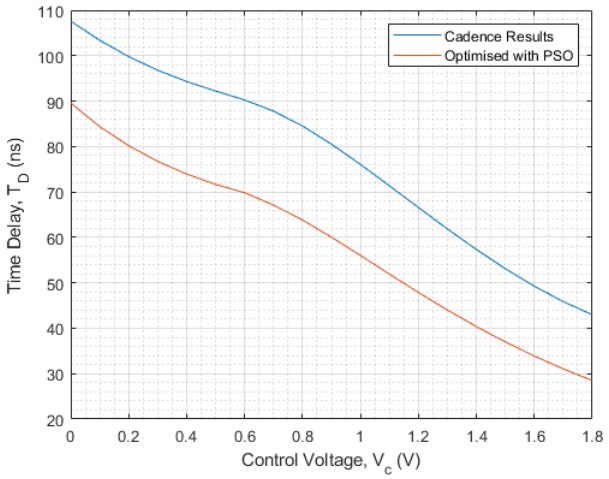


Figure 4.23: Delay response of the delay element after inverters

524.40×357.38 µm^2.

4.10 Lock Detector and Sampling Circuit

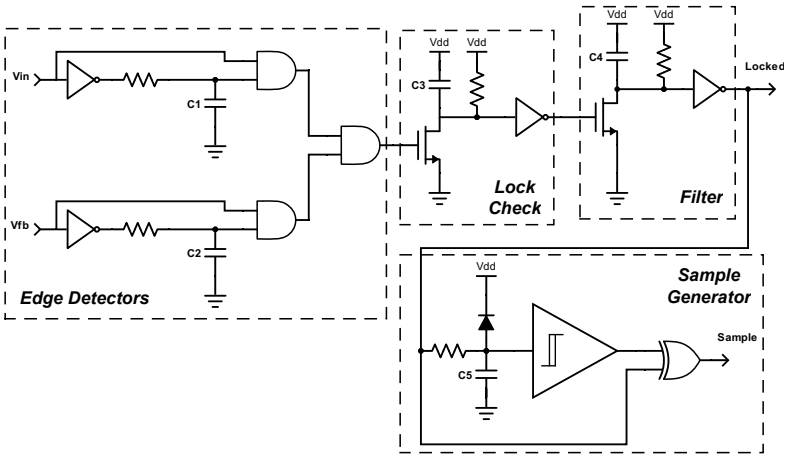


Figure 4.24: Lock detector circuit

The ADC requires a sampling circuit such that the control voltage is sampled when the DLL is locked. The circuit is illustrated in Figure 4.24 and it consists of four stages. The first component is a rising-edge detector for the input square wave and the feedback from the DLL. These two edge triggers generate a pulse of 400 ps at the rising edge of each of the inputs. This value takes into consideration the minimum delay between each signal to ensure that the two signals are locked, together with the propagation delay of the AND gate. Whenever the delay is 200 ps or less, capacitor C_3 of the *Lock Check* stage is fully discharged at each cycle. This generates a pulse of 7 ns at its output irrespective of the delay between the inputs. This signal is fed into the third block, which generates a constant output voltage whenever the pulse is at present at its input. This output is maintained while V_{in} and V_{fb} are in lock. The *Sample Generator* stage generates a 60 ns pulse which is fed to the ADC, such that the control voltage is sampled. Figure 4.25 shows what happens when the two input signals are in lock. When the two waveforms are not in phase, the “Locked" and “Sample" signals output 0 V.

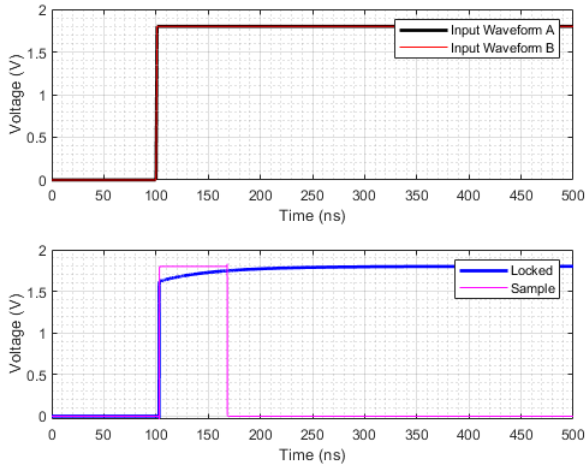


Figure 4.25: Output of Lock Detector and Sample circuit when the input waveforms are in phase

4.11 Layout Considerations and Packaging

The architecture was implemented in Cadence using the X-FAB 0.18 µm technology. The layout of the chip is shown in Figure 4.26, and it occupies a total area of 2.434×2.434 mm^2. The ASIC consists of the complete delay generator, with the fully programmable digital divider, the DLL, and the ADC and DAC. In addition, spare blocks were added to be able to characterise each individual block. These include an additional DLL, a lock detector, a spare delay line, and a spare delay element. The switches were implemented as two transmission gates with their outputs shorted together and an enable signal in the input is used to select the desired input path. Power on Reset (POR) circuits were also added to send a reset pulse to the ADC and the divider.

Although the ASIC is pad-limited, and reducing the number of pins would reduce the overall chip area and cost, this was not possible. This is because spare components were added together with a number of auxiliary I/O connections to allow for thorough testing of the ASIC. Therefore, the final number of inputs and outputs of the ASIC required are:

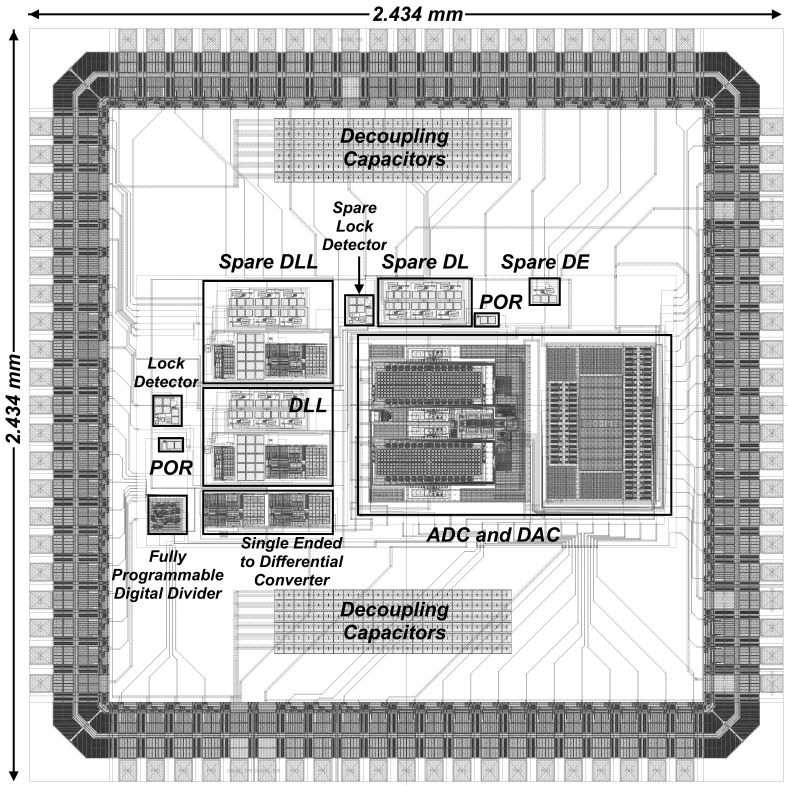


Figure 4.26: Chip layout of the proposed delay generator

- 4 Power and 2 Ground Pins
- 14 input pins and 2 output pins for the digital divider
- 21 input pins for the Delay Generator (including 3 bits for ADC current control, 1 pin for the ADC clock, and 12 bits for DAC input) and 17 outputs which include the 12-bit word from the ADC
- 13 pins for the spare DLL (7 input and 4 output)
- 5 pins for the spare delay line (4 inputs and 1 output)
- 5 pins for the spare delay element (4 inputs and 1 output)

This implies that 83 pins are required to be able to fully characterise the ASIC. The pin list is given in Table 4.6. The required I/O pads were chosen from the X-FAB IO_CELLS_C1V8 library which work with a voltage of 1.8 V. For all the analogue

inputs and outputs, the APR00DPA pads were chosen, while for the digital inputs the ISPA pads were chosen. Special consideration had to be taken for the digital output pads, where the driving strength had to be considered. The BT2SPA output pads, having a 2 mA drive strength, were chosen specifically for the slow outputs, such as the converted digital words. For the fast outputs, such as those from the frequency divider and the DLL outputs, the fast version (BT2PA) was chosen. For the power and ground outputs, the VDDALLPADPA and GNDALLPADPA pads were chosen. These two pads provide an input maximum current rating of 43 mA to the core. The main DLL including the ADC and DAC consume around 6 mW from a 1.8 V supply, thus making these pads more than enough to power the circuit. Nevertheless, for ease of routing, six power pads were used: 2 for analogue voltage, 2 for digital voltage, and 2 for ground.

Due to the fact that the number of pins required is large, the Kyocera CLCC84 package was chosen. This package has a footprint of 2.921×2.921 cm^2. A packaged die complete with bonding is shown in Figure 4.27.

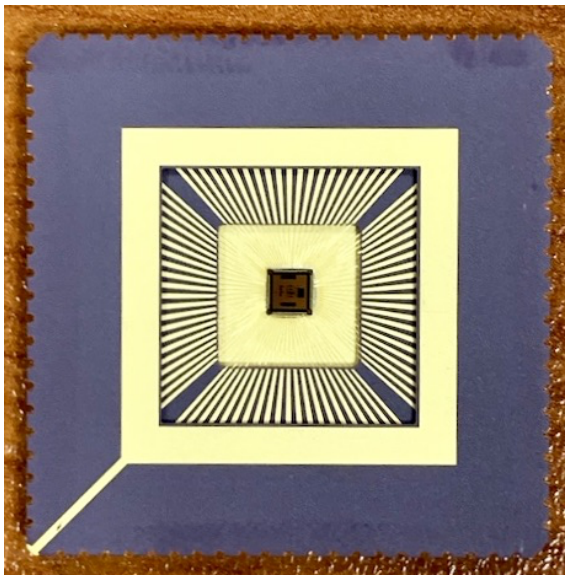


Figure 4.27: Packaged die

Pin number	Pin Name	Pin Description
1	SEL_VC_DAC	Selects whether V_c will come from loop or from DAC
2-10	PC_CONTROL	Input Bus for Program Counter
11	Calibration Clock	500 MHz calibration clock for divider
12-15	SC_CONTROL	Input Bus for Swallow Counter
16,17	GND	Ground
18-20	B_ADC	Internal Current Controller of ADC
21-32	Digital_Out	Digital Output from ADC
33,50	V_{DDD}	1.8 V Digital Power
34	Fout	Frequency Divider Output
35	Fout_DCC	Frequency Divider Output from DCC
36,66	V_{DD}	1.8 V Analogue Power
37-48	DATA_DAC	12-bit Digital Input to DAC
49	EOC	End of Conversion for ADC
51	Vc	Control Voltage Output from Main DLL
52	LockOut	Lock Detector Output
53	DN	Down Signal from Phase Detector
54	UP	Up Signal from Phase Detector
55	SpareDE_Vin	Input Signal to Spare Delay Element
56	SpareDE_Vout	Output Signal from Spare Delay Element
57	SpareDE_EN	Enable 1 pF capacitor of Spare Delay Element
58	SpareDE_EN2	Enable 500 fF capacitor of Spare Delay Element
59	SpareDE_Vc	Input Control Voltage for Spare Delay Element
60	CLK_ADC	20 MHz clock input to ADC
61	SpareDL_Vout	Output Signal from Spare Delay Line
62	SpareDL_EN	Enable 1 pF capacitor of Spare Delay Line
63	SpareDL_EN2	Enable 500 fF capacitor of Spare Delay Line
64	SpareDL_Vc	Input Control Voltage of Spare Delay Line
65	SpareDL_Vin	Input Waveform of Spare Delay Line
67	DLL1_Vout	Output Voltage of Spare DLL
68	DLL1_Vc	Control Voltage Output of Spare DLL
69	DLL1_Sample	Sample Signal from Spare Lock Detector connected to Spare DLL
70	DLL1_LockOut	Lock Out from Spare Lock Detector connected to Spare DLL
71	DLL1_Up	Up signal from phase detector of Spare DLL
72	DLL1_Dn	Down signal from phase detector of Spare DLL
73	DLL1_EN_Cap1	Enable 1 pF capacitor in VCDL of Spare DLL
74	DLL1_EN_Cap2	Enable 500 fF capacitor in VCDL of Spare DLL
75	DLL1_SelAB	Select whether Input A or Input B will be directed to DLL
76	DLL1_Sel_Vc	Select whether the input control voltage will come from Loop or Externally applied
77	DLL1_Vc_in	External Control Voltage of Spare DLL
78	DLL1_VREFA	Reference Input A of Spare DLL
79	DLL1_VREFB	Reference Input B of Spare DLL
80	NC	No Connection
81	EN_cap1	Enable 1 pF capacitor in VCDL of main DLL
82	EN_cap2	Enable 500 fF capacitor in VCDL of main DLL
83	LM_in	LM signal to be delayed by VCDL of main DLL
84	SEL_CLK_LM	Select whether the input to the VCDL comes from divider or from LM_in

Table 4.6: Pin List

4.12 Design of Test Board and Testing Methodology

A Printed Circuit Board (PCB) has been designed, using KiCAD, such that the ASIC could be tested. The PCB consists of four layers, with the top and bottom layers dedicated for routing and the middle layers dedicated to the ground and

power planes, respectively. Most of the components were placed on the top routing layer, except for the 500 MHz clock generator. The reason why this was done is due to the fact that it was necessary to use an LVDS to CMOS converter which has its inputs inverted with respect to the clock generator. Other components on the PCB are the CLCC84 site on which the ASIC is soldered, a through-hole VHDCi connector to be able to connect to the FPGA development board, and various through-hole SMA and other connectors. In addition, there are also sites for voltage-level translators to ensure that the signalling to/from the FPGA is at the correct level, switches to be able to select and de-select capacitors in the delay elements, voltage regulators, LEDs to indicate the presence of power, and de-coupling capacitors to improve noise immunity.

Two LT1965 voltage regulators were used to convert the external 3.3 V supply into the 1.8 V required for the ASIC to operate. These regulators are able to provide separate analogue and digital supplies. Grounding techniques have been used to make sure that high frequency digital noise would not couple with the sensitive analogue signals. This entailed the design of two ground planes, joined together in a single point. This effectively isolates as much as possible the analogue and digital circuitry.

The SI545 crystal oscillator was used to generate the 500 MHz clock signal necessary for the fully programmable digital divider. Due to its high frequency output, only an LVDS output variant was available from the manufacturer. Since the input of the frequency divider is single-ended, an LVDS to 1.8 V low voltage CMOS converter had to be used. The SN65LDS4 high-speed differential line receiver was used for this purpose. Although the manufacturer recommends a maximum operating frequency of 250 MHz, the output signal rise and fall time is equal to 750 ps, for a 10 pF. Post-Layout simulations on the ISPA pad at the input of the divider, show that the pad adds a 7 pF load. This implies that the device is capable of having smaller rise and fall times, thus extending slightly the frequency capability.

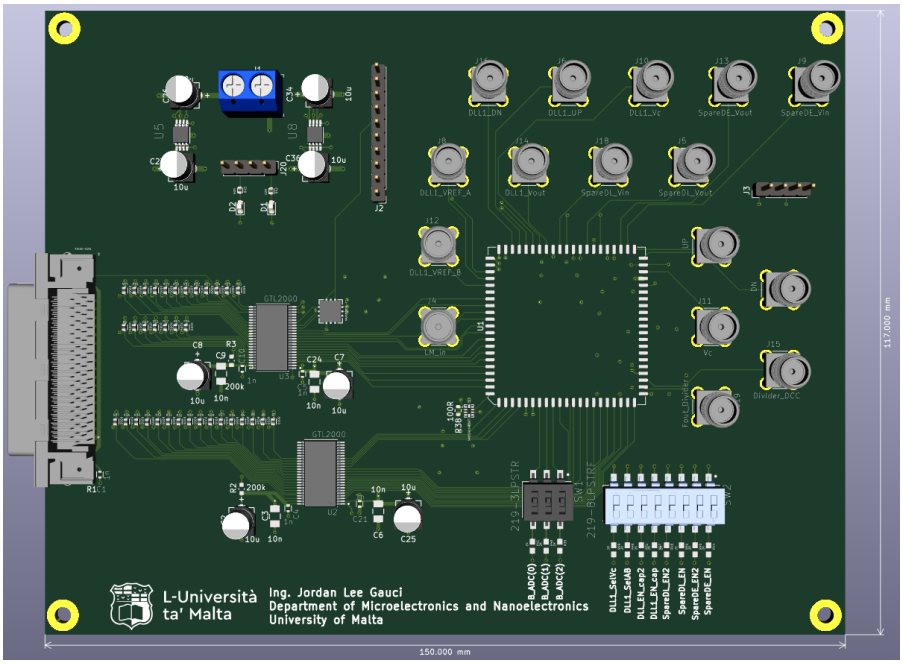


Figure 4.28: 3D Render of the final PCB layout

A problem was discovered during the design of the PCB. The pin from the main DLL output was not connected to its relevant pad. This rendered the on-chip DAC useless, as without an output the operation of the main DLL could not be verified. Thus, the PCB also contains an external Digital-to-Analogue Converter (DAC). The MAXIM 5530 12-bit DAC was used for this purpose, and it was connected to the control voltage pin of the spare delay-line in the ASIC, such that its delay characteristic can be achieved.

The final PCB has dimensions of 150 mm × 117 mm. A 3D render of the PCB is shown in Figure 4.28, and Appendix B shows the layout of the PCB in KiCAD and its schematic together with an illustration of the final PCB design with components soldered, and a list of the components that were used in the PCB to be able to verify the operation of the delay line ASIC.

Since the complete architecture could not be fully characterised, only individual blocks could be tested. The ASIC was tested using the following equipment:

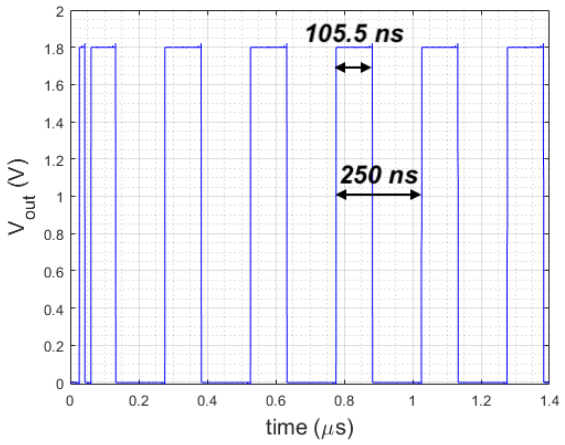


Figure 4.29: Output of the programmable frequency divider for an input frequency of 500 MHz

- 2 Oscilloscope Hameg HMO3522
- 2 Power Supplies Delta ES-030-5
- Multimeter PeakTech 2010DMM
- Signal Generators TTi TG550
- Digilent NEXYS3 Development Board (Xilinx Spartan 6 FPGA)

The sections that follow detail the results obtained from the ASIC, together with any testing methodologies used to verify the correct behaviour of the ASIC. An adequate comparison and reference to post layout simulation results is also made to aid the discussion.

4.13 Post Layout Simulations and ASIC Characterisation

4.13.1 Programmable Digital Divider

To test the fully programmable digital frequency divider, the program counter value was set to 15 while the value of the swallow counter was set to 5. This sets the output frequency of the divider to 4 MHz for a reference frequency of 500 MHz. Figure 4.29 illustrates the results obtained with post-layout simulation showing that, after an initial transient, the output frequency of the divider settles to 4 MHz, with a duty cycle of 42.2%. This is similar to the duty cycle obtained from pre-layout simulation, which is equal to 45.5%.

A simple program was written in VHDL and implemented on the Digilent NEXYS3 Development Board with the Xilinx Spartan6 XC6LX16-CS324 FPGA. The program is capable of receiving commands from MATLAB via the RS232 interface and sends the values of the program and swallow counters to the ASIC via the on-board VHDCi port. A selection of the commands sent, together with their results is presented in this section. The yellow response shows the output of the divider after duty cycle correction, while the blue shows the output from the divider.

In total, four tests were performed on the divider. The program and swallow counters were programmed according to the desired output frequency, and are presented in Table 4.7. Figures 4.30a-4.30d illustrate the different outputs from the fully programmable digital divider. It can be seen that the divider generates the correct frequencies, with some small errors, depending on the settings of the program and swallow counters. The duty cycle obtained is also comparable to those obtained from post-layout simulations, and are tabulated in Table 4.8.

Desired Frequency	Program Counter	Swallow Counter
2 MHz	31	2
1.9763 MHz	31	5
3.968 MHz	15	6
4 MHz	15	5

Table 4.7: Divider Configuration according to the desired output frequencies

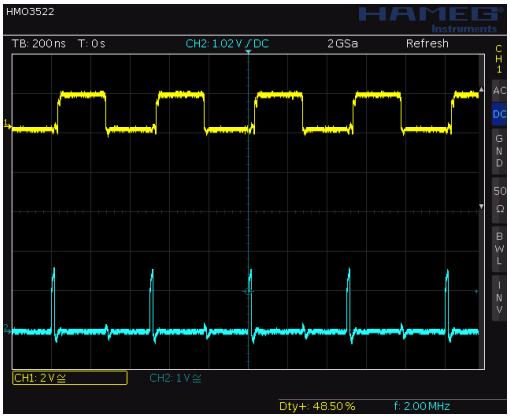
Desired Frequency	Output Frequency	Duty Cycle (Simulated)	Duty Cycle (Measured)
2 MHz	2 MHz	48.4%	48.5%
1.9763 MHz	1.977 MHz	46%	45.94%
3.968 MHz	3.96 MHz	34.7%	34.32%
4 MHz	4 MHz	42.2%	41.68%

Table 4.8: Main measured results from the divider comparing the desired and output frequencies together with the simulated duty cycle and actual duty cycle achieved

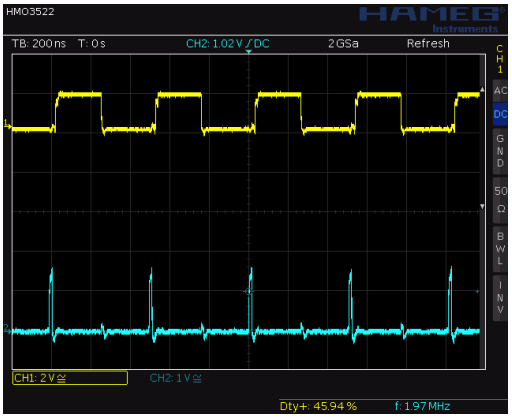
4.13.2 Delay Element

The delay element was tested by applying an input square wave to the V_{in} terminal of the delay element, and measuring the delay between the input signal and output signal with an oscilloscope. Although the oscilloscope used does not have automatic delay measurement, the mathematical functions of the oscilloscope were used to automatically calculate the delay. This process was performed three times for the different capacitor values, 0.5 pF, 1 pF, and 1.5 pF. To remove any unwanted jitter from the results, which may introduce an error in the results, the averaging function with 16 samples on the oscilloscope was used. The results from the delay element are illustrated in Figure 4.31.

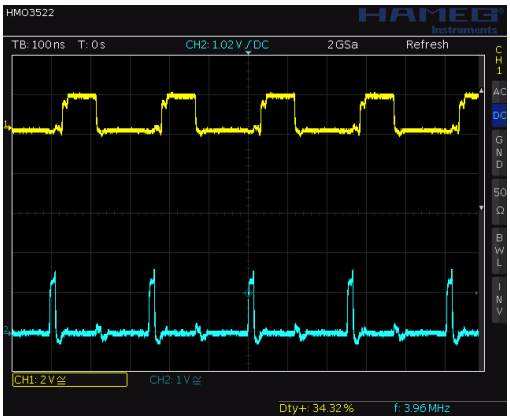
It can be seen that there is a slight discrepancy between the delays obtained through post-layout simulations and those obtained through measurements. It is evident immediately that a slight offset is introduced. This can be directly attributed to on-chip parasitic capacitances as well as additional capacitances from the PCB design, cables, and oscilloscope. Additionally, it can be seen that the discrepancy is greater when the delay element capacitor is larger. This is due to the fact that an increase in load capacitance not only modulates the offset, but also the gradient



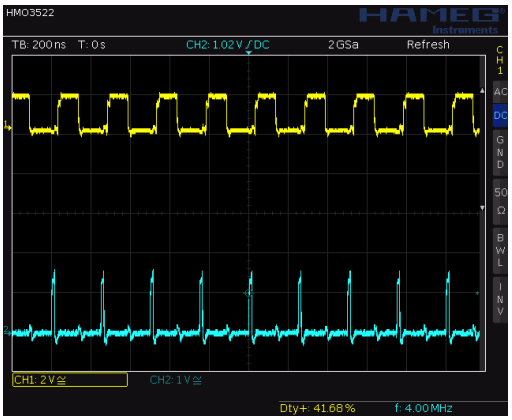
(a) 2 MHz output waveforms from the fully programmable digital divider



(b) 1.976 MHz output waveforms from the fully programmable digital divider



(c) 3.968 MHz output waveforms from the fully programmable digital divider



(d) 4 MHz output waveforms from the fully programmable digital divider

Figure 4.30: Results from the fully programmable digital divider

of the curve. Linearity also suffers in this case as any non-linearities in the delay characteristic are essentially amplified. The manufactured delay element has a range of approximately 35 ns to 191 ns, which is higher than that achieved through post-layout simulation. In this case the total range of the delay element is 30 ns to 153 ns.

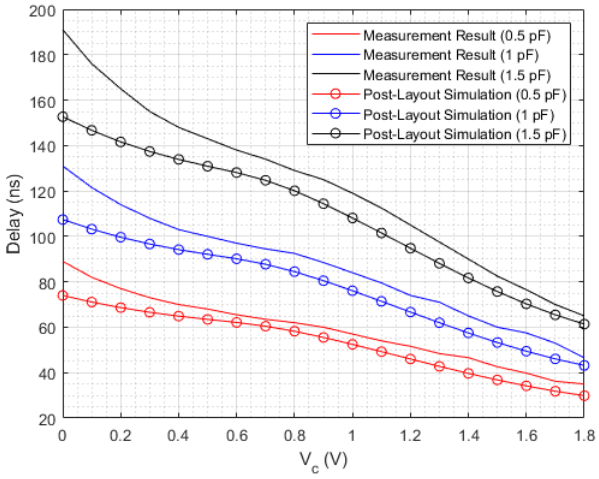


Figure 4.31: Measured delay response of the delay element and its comparison to the post-layout simulations

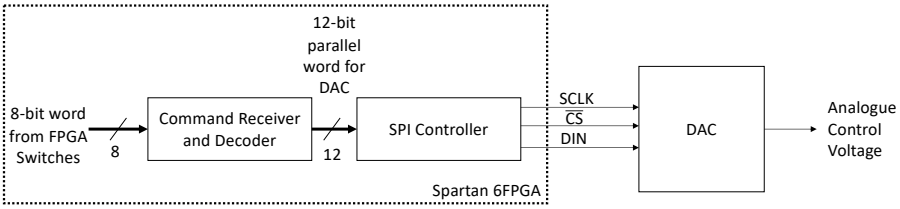


Figure 4.32: Experimental setup for testing the delay line

4.13.3 Delay Line

The control voltage of the spare delay line implemented on the ASIC is connected to the MAX5530 Digital-to-Analogue converter. Therefore to test the spare delay line, a simple VHDL program had to be written and loaded on the Spartan 6 FPGA to be able to control the DAC, which can be controlled via a Serial Peripheral Interface compatible protocol. A top-level view of the control setup is illustrated in Figure 4.32.

The program consists of two modules with the first word being the “command receiver and decoder". The eight on-board switches of the FPGA were used to vary the DAC input word such that the control voltage can be varied in steps of 0.1 V. This means that, for a reference voltage of 1.8 V, the required word should

vary in steps of 228. A total of 19 combinations were programmed to achieve the required analogue range (0 V - 1.8 V).

The second block in the FPGA control program is the “SPI Controller" block. This block takes as its input the 12-bit parallel word from the previous block and, whenever there is a change in the input word, the word is outputted serially using the SPI protocol. The SPI protocol is a 3-wire interface, consisting of the clock (SCLK), active-low chip select ($\overline{\text{CS}}$), and the serial data line (DIN). The chosen DAC [70] has an input 16-bit shift register, in which a 16-bit data word is loaded. This data word consists of a 4-bit command (“1111" for normal operation) and the 12-bit data word. During the transfer of this word the $\overline{\text{CS}}$ line must be kept low. While the DAC can allow a maximum clock frequency of 16.7 MHz, the DAC was clocked at a frequency of 1 MHz. This is due to the fact that the clock was generated on the FPGA itself, and sent via the VHDCi connector.

The FPGA was connected to the testboard by means of a VHDCi cable, and the different voltage combinations were sent to the DAC. The results from the delay line are illustrated in Figure 4.33. It can be seen that there is a good correlation between the post-layout simulations and actual measurements. Similar to the results of the delay element, there is a slight offset. This offset is less prevalent than that present in the delay element. This is due to the fact that the delay line is constructed of five delay elements. Thus, while in the delay element the additional load capacitance had a significant effect on the delay response, this is not that present here. This is due to the fact, that the additional load capacitance is only seen by the last delay element in the delay line. Since the delay range is significantly large, compared to a delay element, the effect is almost negligible on the whole line. It can be seen that the total range of the delay line varies from 935 ns to 183 ns. This is approximately equal to five times of the delay response of the single delay element, which is as expected.

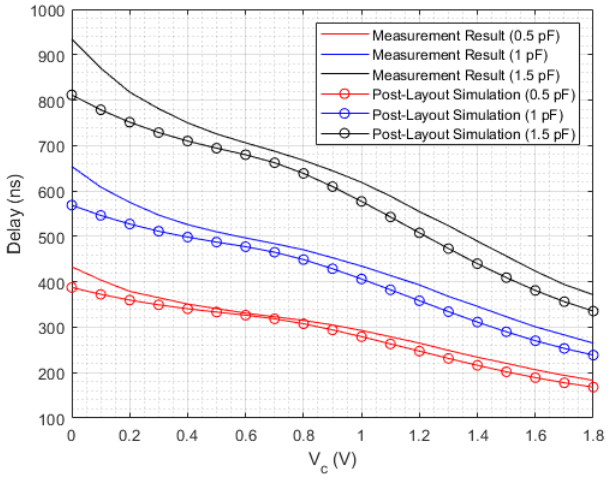


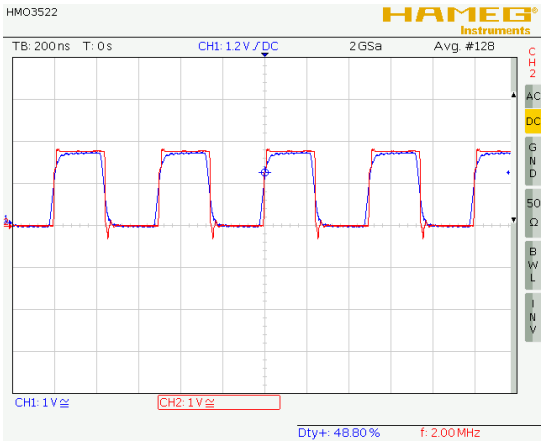
Figure 4.33: Comparison between post layout simulation and measurements

4.13.4 Delay Locked Loop

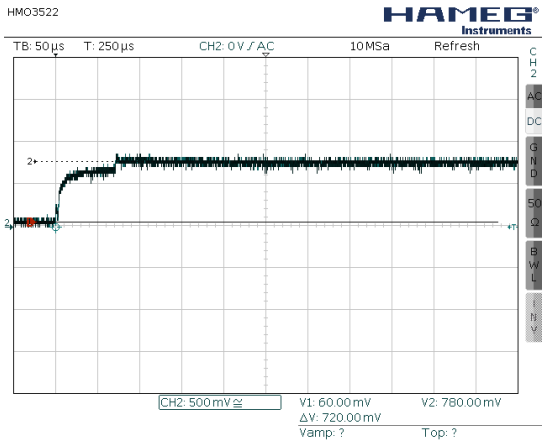
The delay locked loop was then tested. The phase difference between the input and output is calculated via the phase detector, implemented similarly to that in [71], and the control voltage is generated accordingly from the charge pump and loop filter.

Two tests were performed at different frequencies to ensure locking of the DLL. In this section the blue waveforms signify the input signal from the signal generator and the red waveforms show the output signal from the DLL.

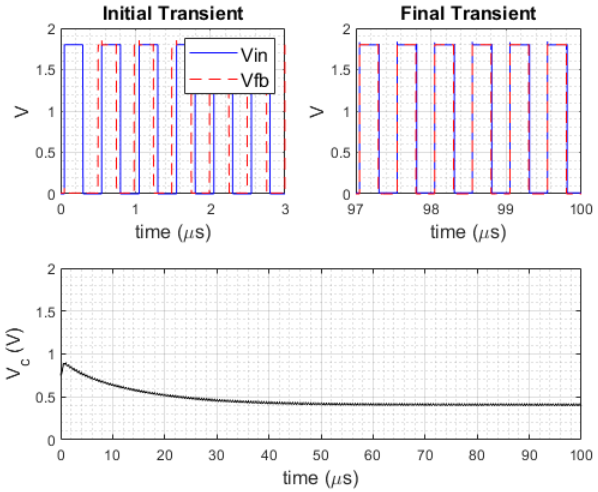
A square wave of 2 MHz was applied to the input terminal of the DLL on the PCB, and the input, output and control voltage were monitored using the oscilloscopes. The 1 pF capacitor was enabled in the voltage controlled delay line of the DLL. Figure 4.34a illustrates the relationship between the input and the output waveforms, while Figure 4.34b shows the transient of the control voltage while locking. Two observations can be noted. First, referring to figure 4.34a, the type of delay element used is symmetrical in operation, when compared to the original delay element proposed by [64], as the duty cycle of the input and output waveforms is similar. Secondly, it can be seen that the control voltage when locked is equal to



(a) Input and output waveforms while the DLL is locked for an input of 2 MHz



(b) Transient of control voltage while locking



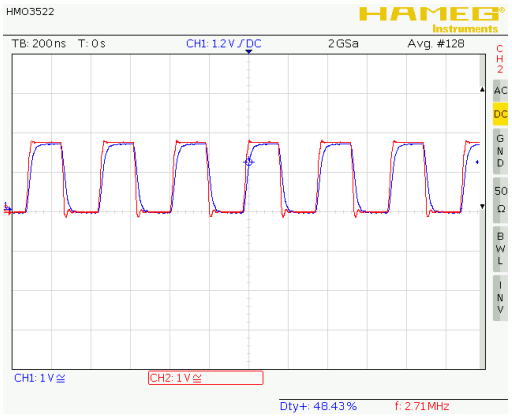
(c) Post-layout Cadence simulation results showing the initial and final transients, and the control voltage

Figure 4.34: Results from DLL for an input frequency of 2 MHz and comparison with post-layout simulation

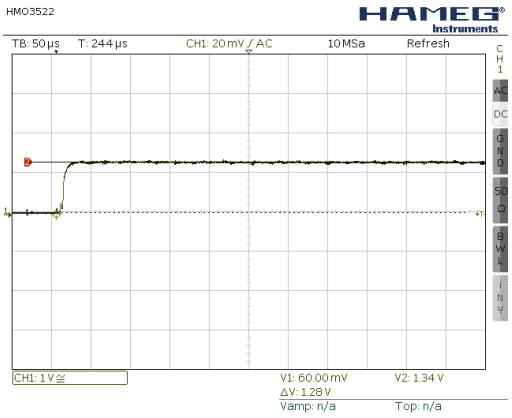
780 mV, which differs from that obtained through post-layout simulation. Simulation results (Figure 4.34c) show that the control voltage, when the loop is locked, is 400 mV. The discrepancy is due to the increased capacitative load on the delay line.

This effect is further evident when a 2.71 MHz signal is fed to the delay locked loop. Figures 4.35a and 4.35b illustrate the relationship between the input and

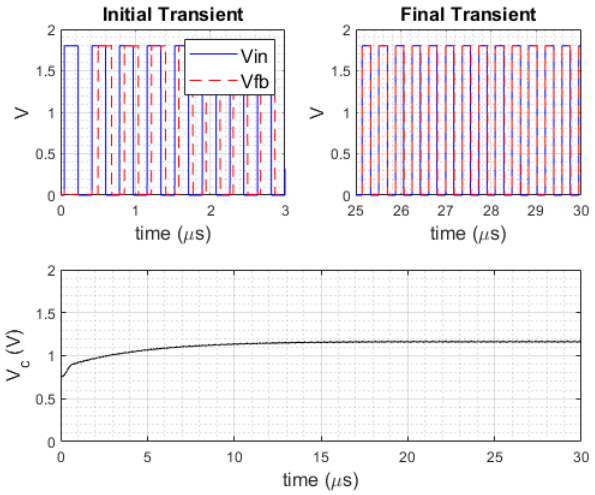
output waveforms and the control voltage response. While in this case the loop settles at 1.34 V, results from post-layout simulations show that the control voltage is 1.16 V. Again, the discrepancy is due to the increase in the capacitative load that increases the effective delay thus also increasing the control voltage. Further post-layout results are illustrated in Appendix C.



(a) Input and output waveforms while the DLL is locked for an input of 2.71 MHz



(b) Transient of control voltage while locking



(c) Post-Layout Cadence simulation results showing the initial and final transients, and the control voltage

Figure 4.35: Results from DLL for an input frequency of 2.71 MHz and comparison with post-layout simulation

4.13.5 Complete Architecture

While the full delay generator architecture could not be tested from the ASIC, a number of post-layout simulations were performed to show that the delay generator works as expected.

Figure 4.36 shows a post-layout simulation of the initial transient and the moment when the DLL is locked, for an input frequency of 4 MHz. The lock signal is set to high when the phase difference between the input and output voltages is less than 200 ps. When this signal is high, the voltage is converted into a digital code by the ADC. In this case, for a voltage of 1.674 V, the 12-bit ADC code is "111011011101". When this code is applied to the 12-bit DAC, the resulting delay is equal to 250.2 ns, resulting in an offset of 200 ps.

This procedure was repeated several times to verify the correct operation of the delay generator. Table 4.9 shows a selection of the results obtained from the delay generator. The table illustrates the output of the ADC, the control voltage from the DAC, the delay achieved when introducing the ADC word in the DAC, and the offset between the expected and obtained delay values. Since during normal operation, the delay generator will work around 400 ns, several tests were performed around this point. It can be seen that there are slight discrepancies between the delay required and those obtained. These discrepancies can be attributed to two reasons. First, the lock detector generates a lock signal when the offset between the input and output waveforms lies within 200 ps, thus the control voltage is sampled earlier by the ADC. The second source of error relates to the conversion of the ADC and DAC. In fact, it can be seen that the errors are larger when there is a higher discrepancy between the control voltage when the DLL is locked and the control voltage from the DAC. It may be noted that DLL was calibrated in steps of 1 ns, particularly around the region where the delay is 400 ns. The maximum error obtained was when the delay was set to 401 ns, where the offset is 300 ps.

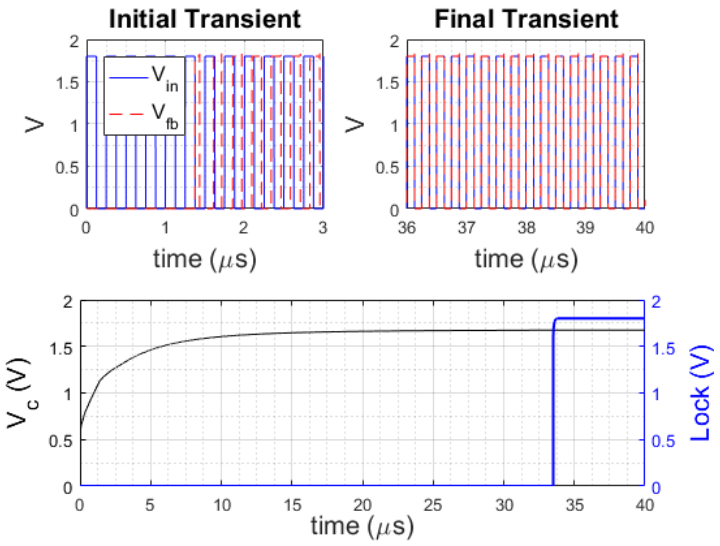


Figure 4.36: Post-layout simulation showing the transient behaviour of the DLL while locking to an input frequency of 4 MHz

This implies that although the delay generator will be tuned in steps of 2 ns, it is indeed possible to tune the delay in steps of 1 ns, provided that the error will not affect the operation and keeping in mind that perturbations on the control voltage can have a detrimental effect on the performance of the delay generator.

Required Delay	**Input Frequency**	V_c	**ADC output**	V_c **(from DAC)**	**Delay**	**Error**
500 ns	2 MHz	406.3 mV	001110011100	406.1 mV	500.1 ns	100 ps
403 ns	2.481 MHz	1.02 V	100100010001	1.02 V	403 ns	0 ns
402 ns	2.488 MHz	1.024 V	100100011010	1.0239 V	402 ns	≈ 0 ns
401 ns	2.494 MHz	1.031 V	100100101010	1.031 V	400.7 ns	300 ps
400 ns	2.5 MHz	1.033 V	100100101110	1.0327 V	400.2 ns	200 ps
399 ns	2.506 MHz	1.04 V	100100111110	1.0397 V	399.3 ns	300 ps
398 ns	2.513 MHz	1.037 V	100100110111	1.0367 V	398.1 ns	100 ps
369 ns	2.71 MHz	1.162 V	101001010100	1.1619 V	368.9 ns	100 ps
250 ns	4 MHz	1.674 V	111011011101	1.6739 V	250.2 ns	200 ps

Table 4.9: A selection of results from post-layout simulation of the complete architecture

Figure 4.37 illustrates the delay between the input (LM) and output signals achieved by applying a 1.0329 V control voltage, V_c. It can be seen that the 150 ns LM signal is delayed by 400.2 ns, as initially calibrated by the DLL.

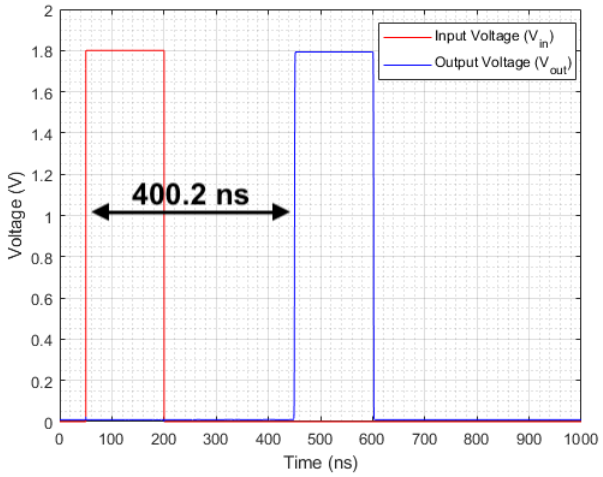


Figure 4.37: Post-layout simulation showing the delay of the LM signal by 400.2 ns

4.14 Conclusions and Discussion

This chapter has presented a novel architecture for a delay generator that is based on a DLL which, once calibrated, can be used in open-loop configuration. Specifically, this chapter has focused on the design, optimisation, implementation, simulation and testing of a symmetric operation, quasi-linear, rail-to-rail delay element. Three different techniques were used to optimise the linearity of this delay element. The first two techniques are based on approximating the time delay equation by a second-order polynomial, and minimising the second-order term. The third method is based on the use of bio-inspired algorithms to find the transistor aspect ratios that give the most-linear response for the required range and resolution. The complete architecture was fabricated using the X-FAB XH018 technology, and results for each block in the architecture were given.

A 4-layer PCB was designed such that the ASIC can be tested, with appropriate SMA connectors at various inputs and outputs of the delay generator circuit, and VHDCi connector such that the FPGA can be used to remotely configure and control the ASIC. Results from the fully programmable digital divider, the delay

lines and delay elements, and the delay locked loop have been presented. Since there was a design error that prevented the full characterisation of the ASIC, only individual blocks could be tested.

The first component that was tested was the digital divider, where 4 configurations were tested, and it was seen that the divider works as expected. The fully programmable digital divider architecture has a wide range that can generate all the output frequencies required which will then feed into the delay locked loop for calibration.

The delay element was also tested by applying an input signal, varying the control voltage, and measuring the delay. It was seen that the delay response follows that which was obtained from post-layout simulations, albeit with a slight discrepancy which can be attributed to parasitic and load capacitances. This discrepancy is greater when the 1.5 pF load capacitance of the delay line is selected, and linearity also suffers in this case. The reason for this is that the delay element was optimised to obtain the most linearity for a load of 1 pF, and therefore any non-linearities are amplified with the larger capacitance.

This effect is also evident in the delay line, although in this case the error is smaller. This is due to the fact that only the last delay element in the line is loaded, and the additional delay generated by the last delay element is negligible when compared to the complete range of the delay line. The resulting delay line has a range from 183 ns to 935 ns, which is well within the specifications required by HMPID. The required delay resolution (2 ns), can be achieved by programming accordingly the DAC. Although this can be achieved relatively easy when the 0.5 pF and the 1 pF are in the delay line, this may be problematic when the 1.5 pF capacitance is enabled. This is due to the amplified non-linearities of the delay line and therefore this requires further investigation. During normal operation, the delay range of the LM signal will normally be around 400 ns, thus this issue will not affect the

application.

The delay locked loop was tested by applying a square wave to the input of the DLL and monitoring the output and transient of the control voltage. The results were compared to post-layout simulations and it was seen that the obtained control voltage, when locked, is slightly higher than that from simulations. This was attributed to the increase in the delay of the delay line which also increases the control voltage. The delay locked loop was also tested in open-loop configuration, where both the obtained control voltage (for both cases) and an input square wave were applied directly to the delay line, and the delay measured. This resulted in the same delay as expected, thus illustrating the fact that the delay locked loop can be successfully used to implement a delay generator which can be calibrated via a closed loop DLL, and then used in open loop configuration.

While, from the ASIC, the architecture could not be fully characterised a number of post-layout simulations were run to illustrate that the architecture works as expected. In comparison to the SPI-programmable DLL based delay module used at the ICECAL detector at the LHC beauty (LHCb) experiment at CERN [37], the delay generator architecture presented here exhibits a larger delay range with respect to the 25 ns delay range achieved in [37], at the cost of a coarser delay resolution (2 ns compared to 1 ns). However, the resolution achieved is still within the required specifications. The two implementations occupy a similar area, where the radiation-hardened delay module occupies 5.7 mm^2 [37] against 5.924 mm^2. This work is also an improvement over the delay generator currently used by HMPID [38], where a finer delay resolution can be achieved (2 ns) as opposed to the 25 ns in the current module. Even though the delay range achieved in this work is smaller than the one of the delay generator currently in use, which was 2.8 µs, this does not impact the operation of the detector.

The total measured power consumption of the delay generator ASIC, including the

ADC and DAC circuits, is 6 mW from a 1.8 V supply, whereas the total power consumption of the test board (including the ASIC) is approximately 200 mW.

5. The HMPID Firmware

This chapter deals with the upgrades that had to be done to the HMPID firmware. These upgrades included the ability for HMPID to handle the new triggering requirements, and to integrate HMPID in the new O^2 readout environment. While for Runs 1 and 2 (2009 - 2018) HMPID utilised three trigger levels (L0, L1, and L2), this will no longer be the case for the forthcoming upgrade. In fact, HMPID will work with just two trigger levels (LM and L1). While the previous chapters have focused on the introduction of a delay to the LM signal, to meet the required timing of HMPID, this chapter will focus on the firmware upgrades necessary to handle the L1 trigger and the event readout between the front-end electronics and the data acquisition system. This chapter therefore starts with a brief presentation of the triggering requirements, and then moves on to the actual upgrades that were performed.

5.1 Front End Electronics Construction

HMPID is made up of 14 panels, with each one connected to the DAQ system through the Detector Data Link (DDL) link. The purpose of the read-out electronics is to measure the analogue signal present on the pads of the MWPC, such that the Cherenkov pattern is extracted and sent to the DAQ system. To achieve this, two ASICs called GASSIPLEX and DILOGIC were developed, and these form the

basic building blocks of the front-end and read-out electronics [72].

Figure 5.1 illustrates the read-out electronics used in HMPID. Three GASSIPLEX chips are connected to the cathode of the pads in the MWPC. The analogue signal from the Gassiplex card is fed into the the input of the DILOGIC. The purpose of the latter is to perform zero-suppression (a form of compression where zeros are ignored), pedestal subtraction, and data formatting. Since the data from the Gassiplex card is analogue, a 12-bit ADC is used to digitise the signal before it is fed to the DILOGIC. The ADC and the DILOGICs reside on a dedicated board, called the DILO5 card [72].

Two DILO5 cards (10 DILOGICs) form a single HMPID column, capable of reading a total of 480 pads. A single column controller card manages the timing and configuration of the two DILO5 and GASSIPLEX. This card contains an FPGA with an embedded FIFO to temporarily store the transferred data. Both the column controller cards and the DILO5 cards are mounted on a segment board, which handles the communication between the RCB and the column controllers. A single segment can handle a total of 3840 channels. Since a single HMPID panel has a total of 11520 channels to be read, three segment boards are daisy chained by short cables. Finally, the RCB manages the triggering (via the TTCRx card), control, and transfer of data between the FEE and the DDL Source Interface Unit (SIU), and vice-versa [72].

The RCB (Figure 5.2) consists of the ALTERA Stratix II (EP2S15F484C5) FPGA that may be programmed remotely through the EthernetBlaster, which is a JTAG programmer over Ethernet. The L0 trigger signal is received from the LTU on LVDS cables, while trigger messages are received from the LTU via optical fibre through the Timing, Trigger, and Control Receiver (TTCRx) mezzanine card. Whenever an L0 trigger is received, the BUSY signal goes high and is propagated to the LTU to signal that the detector is busy and cannot handle another trigger, until the

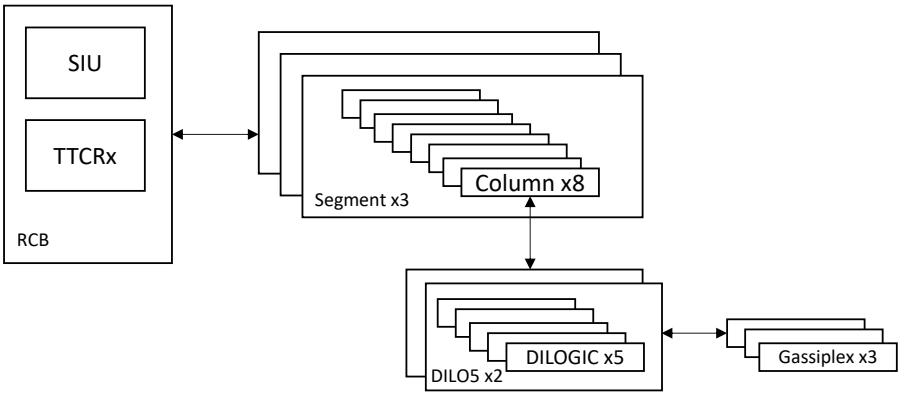


Figure 5.1: Construction of the HMPID read-out electronics

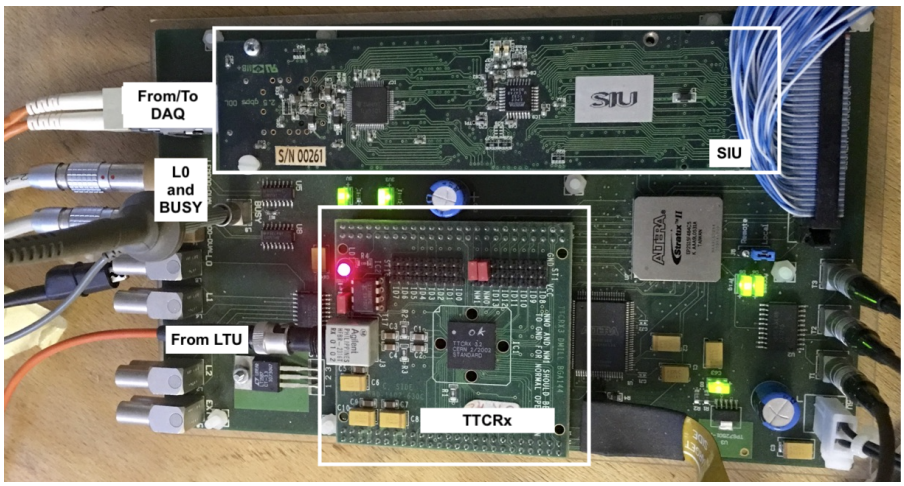


Figure 5.2: The HMPID readout control board

current event is processed. Data transfer between the RCB and the DAQ (and vice-versa) is handled by the SIU mezzanine card, which converts electrical signals to light for transmission over optical fibre [73–75].

5.2 Firmware Upgrades

This section highlights the upgrades to the firmware that have been done, such that the detector becomes Run3 compliant. In particular, the firmware upgrades focused on integrating HMPID in the new trigger and read-out mechanisms.

During Runs 1 and 2 (2009 - 2018), triggering was based on three different trigger

levels, at different latencies; Level-0 (L0), Level-1 (L1), and Level-2 (L2) [7]. To increase the readout performance in Run3, most detectors will be upgraded and will no longer keep using triggers but instead will work with continuous readout. To maintain backward compatibility, triggering will be kept for those detectors which either will not be upgraded (such as HMPID), and other detectors which inherently work with sample-and-hold circuitry (such as the Charge Particle Veto (CPV) detector). The triggering mechanism will differ slightly, as only two trigger levels will be allowed for each detector. HMPID will therefore work with the LM and L1 trigger levels. In addition to the L1 trigger, a trigger message will be received which contain information pertaining to the event. The triggers are generated by the CTP and distributed to each detector through the LTU.

In addition to the new triggering scheme, a new DAQ system, called the O^2 (Online-Offline) environment, is being introduced. While the earlier DAQ system utilised the D-RORC card, based on PCI, the new system will use a more modern version that is based on the PCI Express protocol such that higher data rates can be handled. While the C-RORC card still uses the DDL protocol [76] for communication, there have been some changes to the data transfer mechanism. In particular, the common data header that was used through runs 1 and 2 has been changed to the Raw Data Header (RDH), having a different format. In addition, a single event cannot be sent as a whole but it has to be split into 8 kB pages.

5.2.1 Firmware Design Methodology

Keeping in mind the above discussion, two modifications had to be performed. The first is to implement the new trigger handler which should correctly decode the new trigger mechanism. Once this has been tested, the second modification involved performing the necessary modifications to the interface between the FEE and the SIU. These modifications had to be done, without performing any upgrades to the electronics.

Two approaches were adopted, the first was to modify the existing firmware to remove the dependency on L2, and decode the message. This was done specifically since the new LTU was available for testing, while the new DAQ was not. Once the DAQ was available for use, modifications were performed in the existing firmware to handle the data transfer between the SIU and the C-RORC.

Once testing was successfully performed on the modifications of the existing firmware, a new firmware was designed. This had to be done as the existing firmware was written in AHDL, and therefore needed to be converted to VHDL and to increase as much as possible the readout rate without sacrificing data integrity.

5.2.2 New Firmware Layout

The firmware was implemented in VHDL using the Altera Quartus II 10.1 software, with the layout of the firmware illustrated in Figure 5.3. The largest two blocks in the firmware are the SIU interface and the FEE interface. The former is the main controller and the interface between the DAQ system and the RCB. It handles configuration commands received from the DAQ, and enables the FEE interface when the DAQ is ready to receive data.

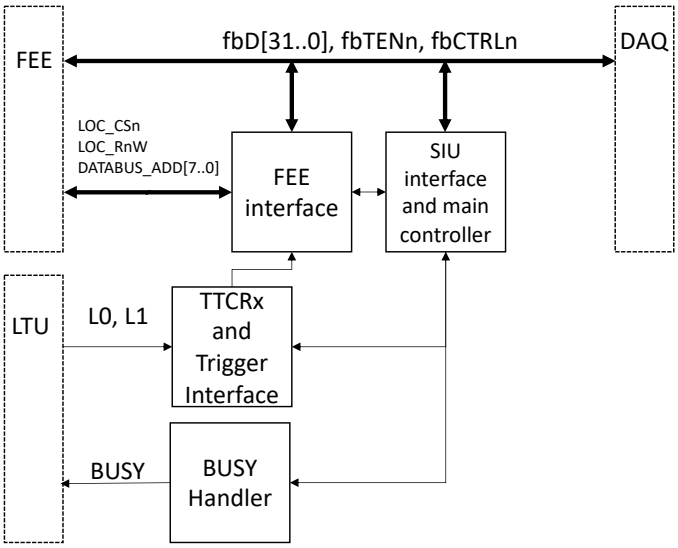


Figure 5.3: New firmware layout

The second largest block in the firmware, is the FEE interface module. Contrary to the previous firmware implementation [77], this module has been completely decoupled from the SIU interface module, as it allows for better modularity and proper clock handling.

The TTCRx and trigger interface block handles the communication between the LTU and the RCB. In particular, it is responsible for handling the received L0 signal and propagating it to the front-end electronics. In addition, this module is also responsible for receiving the L1 trigger and decoding correctly the asynchronous trigger message. The BUSY handler works in conjunction with the TTCRx and trigger interface, to assert or de-assert the BUSY signal on reception of the L0 trigger and end of data transfer.

5.2.3 SIU Interface and main controller

The largest block in the firmware is the SIU interface module. This module is the heart of the firmware, and it manages communication between the DAQ system and the FEE. Its main purpose is to receive and decode commands from the DAQ and sends the relevant control signals to the other blocks in the firmware.

Front-end commands are sent from the DAQ to the SIU interface through the DDL protocol [76], via the 32-bit fbD bus, to control the readout electronics. The assertion of the control signals fbTENn and fbCTRLn signify the presence of a command on the bus.

The “SIU interface" module consists of a state machine, clocked at 40 MHz, which decode the received commands. The commands that may be received by the SIU, and the actions taken by the state machine, are as follows.

- RDYRX - The Ready to Receive command signals the FEE interface that the DAQ is ready to receive data from an event.

- EOBTR - After data is sent to or received from the DAQ, the End of Block Transfer command closes the channel.

- STBWR - A block write transaction is started by the Start of Block Write command. This is used to upload configuration parameters to the FEE. When this command is received, the state machine starts the procedure for writing the configuration parameters (known as pedestals and thresholds) to the columns. This entails the use of a 1024-word FIFO as a buffer to temporarily store the received words, at 40 MHz, and then writes the configuration to the columns at the lower rate of 10 MHz.

- STBRD - A block read transaction is started upon reception of the Start of Block Read command. This command is used to transfer the configuration parameters from the FEE to the DAQ to ensure that the FEE have been configured correctly. The state machine sends a signal to the FEE interface to start the data transfer.

- FECTRL - When the front-end control command is received, the state machine starts to decode the detector-defined commands. These commands include the general reset of all the modules, and the clearing of the BUSY signal.

- FESTRD - The DAQ sends a Front-End Status Read-out whenever it wants to poll the status of the RCB. The RCB responds by sending a 32-bit word containing the error register. This allows for remote monitoring of the electronics between data taking.

5.2.4 TTCRx and Trigger interface Module

The TTCRx chip is a custom ASIC that was developed at CERN to act as an interface between the front-end electronics and the Timing, Trigger, and Control system. The purpose of this chip is two-fold; to synchronise the detector with the

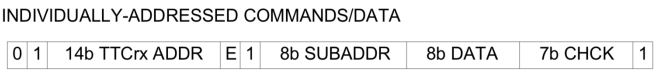


Figure 5.4: TTCRx frame format for an individually-addressed command

40 MHz LHC clock, and to receive trigger message information. This is all received through optical fibre, split into two channels. The L0 and L1 triggers are received on Channel A, while the trigger messages are received on Channel B [78].

Data on Channel B can be either broadcast commands, or individually-address commands/data. The former is used to send messages to all TTC destinations, while the latter can only send message to an individual TTCRx chip. The frame-format for the individually-addressed commands/data frame is illustrated in Figure 5.4. The frame is identified as an individually-address command through the header bits “01". The final bit of the frame is the stop bit, where the bus is returned to logic high. The rest of the fields in the frame are the 14-bit TTCRx address, the address-space selection bit (E), the 8-bit sub-address and data, and the checksum. Usually the data and the lower four bits of the sub-address are grouped together to form a single 16-bit word. The upper four bits of the sub-address show whether the word is the message header, or normal data. A strobe signal, called D_Str signals the presence of a word on the bus.

The format of the asynchronous L1 message is illustrated in Table 5.1, and it contains information relating to the Trigger Type (TTYPE), Orbit, and Bunch Crossing Identification (BCID). With this information, every event can be identified for offline reconstruction and analysis.

The TTCRx module consists of two processes. The first process ensures that the triggers arrive in the correct order, while the second process decodes the asynchronous L1 message. The former consists of a simple state machine that waits for the arrival of L0, on LVDS cable, and L1, on Channel A of TTC. Error detection mechanisms have been added in the software to ensure that L1 is received within

Table 5.1: TTC data format Trigger Message

Data	TTC word	Payload	Content
<15:12>	0	<3:0>	L1 header
<11:8>	0	<23:4>	spare
<7:0>	0	<31:24>	TType
<15:12>	1	<3:0>	L1 data
<11:0>	1	<23:12>	TType
<15:12>	2	<3:0>	L1 data
<11:0>	2	<11:0>	TType
<15:12>	3	<3:0>	L1 data
<11:0>	3	<11:0>	BCID
<15:12>	4	<3:0>	LL1 data
<11:8>	4	<23:4>	spare
<7:0>	4	<31:24>	ORBIT
<15:12>	5	<3:0>	L1 data
<11:0>	5	<23:12>	ORBIT
<15:12>	6	<3:0>	L1 data
<11:0>	6	<11:0>	ORBIT

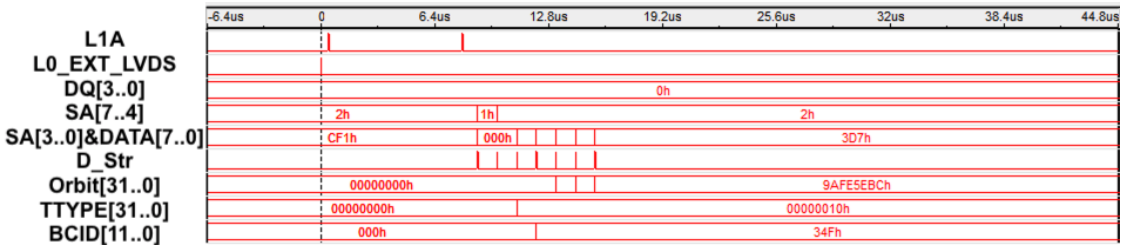


Figure 5.5: Decoding of the asynchronous L1 trigger Message

10 µs, and also to ensure that the correct sequence of triggers is received. In the case of an error, the trigger sequence is ignored and the firmware waits for a new trigger.

Figure 5.5 illustrates the received trigger sequence, from the embedded software oscilloscope SignalTap. An L0 trigger is received on L0_EXT_LVDS. This trigger is later received on Channel A of TTCRx, as a 25 ns wide pulse, and is followed by a 50 ns pulse which signifies the arrival of the L1 accept trigger (L1A). The trigger is followed by the data on the SA and Data busses. The upper 4 bits of the SA denote whether the data received is the L1 message header, or normal data. The

presence of data is indicated through the strobe signal, D_Str. When the Orbit, TTYPE, and BCID are correctly decoded, they are sent to the “FEE Interface" for further processing.

5.2.5 FEE Interface

This block handles the communication between the readout electronics and the RCB. This is done through a simple protocol where two control lines and an 8-bit address bus are used to communicate with the readout electronics.

- **LOC_CSn (active low)**: Chip Select.
- **LOC_Rn/W**: Signals whether a read or write operation is taking place.
- **DATABUS_ADD[7..0]**: Describes the operation that has to be performed, and indicates the specific chip to select.

The upper four bits of the DATABUS_ADD[7..0] is called the NUM field, and its value shows whether a segment or a column is selected. The lower four bits indicate the function that is to be performed. If a segment is selected, this field indicates the segment number, while if a column is selected, it indicates the function to be performed.

The 32-bit bus (fbD[31..0]) is used for data transfer to and from the readout electronics. This bus is shared between the SIU and the FEE. During readout, the words transferred on this bus have a different format depending on the type of word. The data structure that will be used by HMPID during Run3 is shown in Table 5.2.

The first section of the structure is the Raw Data Header, and it consists of 16 words that contain FEE and event identification (Orbit, TType, and BCID). This is followed by the column header. Depending on the type of run (calibration or standalone), the lower 16 bits are set to x“32A8" or x“36A8". The column header

Raw Data Header
Header Column 1
Data word 1 . . Data word n
EoE marker DILOGIC 1
Data word 1 . . Data word n
EoE marker DILOGIC 2 . . . .
EoE marker DILOGIC 10
Header Column 2
. . . .
Header Column 8
Marker Segment 1
. . . .
Marker Segment 2
. . . .
Marker Segment 3
. . . .

Table 5.2: HMPID Data event structure

also contains information on the number of words in the column. This will aid the reconstruction software in correctly assembling an event.

The column header is then followed by the data word, which contains information about the charge-value on the activated pad (12 bits), the channel address (6 bits), the DILOGIC address (4 bits), and the column address (5 bits). Once all the words in a DILOGIC have been read, the DILOGIC generats the End-of-Event (EoE) marker which contains the column address, the DILOGIC address, and the number of words in that particular DILOGIC. When a complete segment is read, the segment marker is sent containing the number of words in the segment, together

with the segment address [77].

The readout electronics interface was implemented in VHDL and it consists of a state machine to interact with the segments and columns, with a clock frequency of 20 MHz. This has been improved from the firmware used during runs 1 and 2, where communication was done at a lower clock frequency (10 MHz). While in previous versions using a lower clock frequency was done to improve synchronisation between the RCB and the segments/columns, the approach used in this firmware is to add a synchronisation layer that waits for the expected word, before continuing.

In its idle state, the state machine listens to commands from the “SIU Interface" block. The state machine may move into four possible states. The first state is the main read-out state, which is responsible for transferring data between the FEE and the DAQ.

A simplified version of the reading process is illustrated in Figure 5.6. Readout begins whenever the DAQ is ready to receive (RDYRX) and the correct trigger sequence has been decoded by the TTCRx. The first step is to send the RDH containing the event information. The first segment is then selected, and this is followed by the selection of the first column. Whenever a column is first selected, it sends the column header. To ensure that the data is truly ready in the column, the code repeatedly monitors the bus, until the correct header is received. A watchdog counter in this state ensures, that if the correct header is not received on time, the column is skipped. Once the header has been received, the data from the column is read. This process repeats until all the 8 columns in a segment are read. The segment marker is then sent, and the second segment is then selected. This process is then repeated, until all the data has been transferred between the FEE and the DAQ. At the end of the data transfer, the EOBTR command is sent to signal that the data transfer has completed.

The FEE interface also takes care of uploading the pedestals and thresholds to the

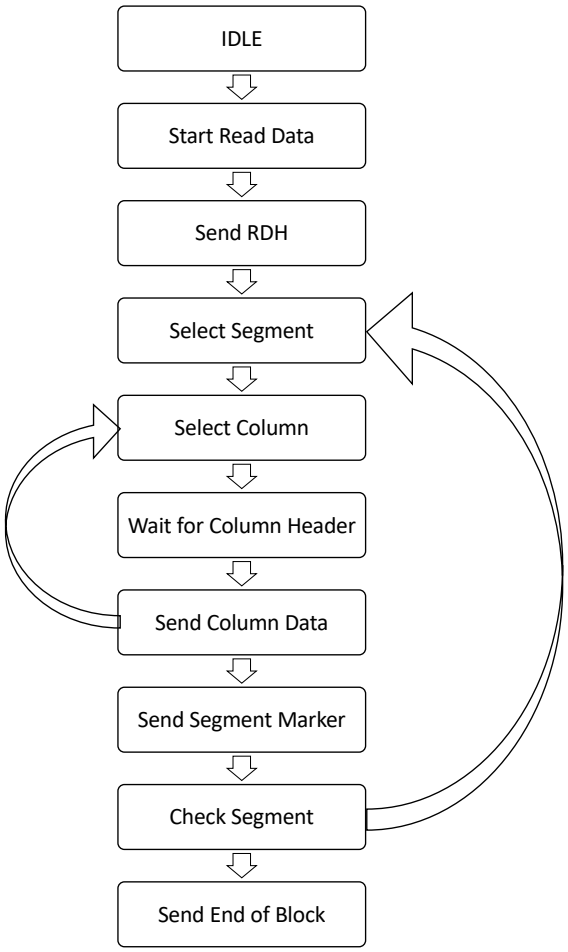


Figure 5.6: Typical Readout Process

columns. Pedestals and thresholds are uploaded to the DILOGIC cards to perform rudimentary event selection. This is done by subtracting the channel value received from the ADC by the pedestal, and any remaining events under the threshold are ignored. This is known as Zero Suppression, and is a simple, yet effective, technique to reduce the amount of words transferred thereby increasing the readout rate.

Zero suppression may be enabled or disabled when the proper command is received from the SIU interface. Whenever an FECTRL command is received, the “FEE Interface" decodes the address, and enables or disables zero suppression for that particular column. This is typically done for all columns, through a script. Due to a bug in the column controller, which cannot be upgraded, zero suppression is

disabled after every event. As such, a loop is performed at the end of every event to re-enable zero suppression. This is one of the bottle-necks during data transfer when zero suppression is enabled.

5.3 Testing Methodology and Results

Testing of the firmware was done at the HMPID Electronics Lab at CERN, where an exact replica of a single HMPID module is present. The RCB of this module is connected to the new LTU and DAQ system. While the new LTU, including a rudimentary version of the software were available, the same cannot be said for the DAQ. In fact, for the DAQ, only the main readout program was written. Thus small C++ programs had to be written such that an interface to the C-RORC card was created, to open a channel of communication with the FEE. This enables correct configuration of the RCB and FEE.

The following sections outline the main results obtained when running the firmware, operating fully in the Run3 environment.

5.3.1 Readout with Zero Suppression Off

On startup, the default mode of readout is with zero suppression turned off. This is done specifically such that a number of events may be gathered to determine the background noise. Through this background noise, the pedestals and thresholds may then be computed and re-uploaded for configuration.

Figure 5.7 illustrates the procedure to perform column selection. The address of the column (in this case Column 2) is placed on the upper 4-bits of DATABUS_ADD, while the lower 4-bits indicate the relevant command sent to the column. Initially, command “1"h is sent to select the column and access its status register. If the data transfer from the DILOGICs to the column controller FIFO has been completed, the column responds with the correct header on the fbD[31..0] bus. In this case, the

column replies back “01EA32A8"h. The lower 16 bits indicate that zero suppression is indeed disabled, while the upper 16 bits indicate the amount of words in the column (“1EA"h). This implies that there are 490 words present in the column FIFO, consisting of 480 channel words and the 10 End-of-Event DILOGIC markers. Once the header has been decoded, the command is set to “4"h indicating the start of the data read cycle, and then changed to “2"h to signal that the data can be transferred. Whenever a correct word is present on the fbD[31..0] bus, the fbTENn signal is asserted (active low) indicating to the SIU that a word has been transferred.

It can be seen, that a single word from the column controller is read every 100 ns. This is much faster than the previous version of the firmware, where a single word was read approximately every 200 ns. In fact, a single column can be selected and read in 49.5 µs (Figure 5.8) as opposed to 122.5 µs in the old firmware. This implies that the a single HMPID module, including segment selection and transmission of RDH, can be read in approximately 1.3 ms - an improvement of 1.3 ms over the version of the firmware used during Run2 [77], and on the preliminary prototype version of the firmware for Run3 [75].

5.3.2 Readout with Zero Suppression On

During normal data taking, the FEE works in zero suppression mode. This process involves the uploading of pedestals and thresholds to the readout, enabling zero suppression, and then reading the electronics using the same method as when zero suppression is turned off.

Since no software was available to test the writing of pedestals and thresholds to the FEE, a simple command line utility was written in C++. Listing 5.1 illustrates the process of sending the pedestals and thresholds to a single column. The program starts by performing a Reset of the C-RORC card, and the SIU, and the optical

multiplexer is configured so that data is sent via the connected channel. The first segment is selected by sending the command “0x5A4A80C4". This corresponds to the word format depicted in Table 5.3 [77], where the DES and NUM fields indicate the destination (in this case, segment 1), and the control field, CTL, gives a command to the RCB to clear the BUSY signal. The DATA fields (from bit 12 to 20) contain a bit to enable or disable Zero Suppression (ZS), and the DILOGIC commands (DILO_CMD, CS, FS). The latter set the number of channels and the multiplexing frequency for the DILOGIC to use [77]. The final 12-bits are SIU defined and are set to “0x0C4" to indicate the start of the FECTRL command, “0x044" to indicate the FESTRD command, and “0x0D4" to indicate the start of a block write operation.

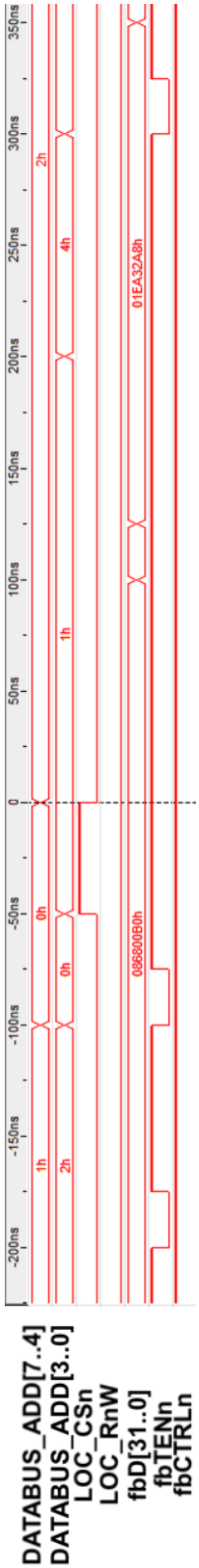


Figure 5.7: Selecting the column

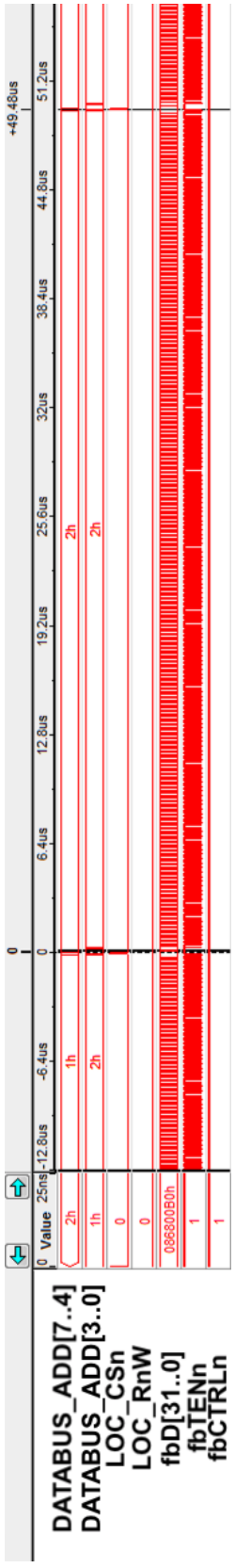


Figure 5.8: Reading a single column

fbD[31..0]										
E	**DES**	**NUM**	**CTL**	**DATA**				**ID**	**CMD**	**0x4**
				ZS	**DILO CMD**	**CS**	**FS**			
31	**30..28**	**27..24**	**23..21**	**20**	**19..16**	**15..14**	**13..12**	**11..8**	**7..4**	**3..0**
Detector Defined								**SIU Defined**		

Table 5.3: Control word format of data from SIU to the FEE

Listing 5.1: Uploading pedestals and thresholds to the first column

```
getLogger() << "- Selecting segment 1" << InfoLogger::endm;
//  ddlSendCommand: 0x5A4A80C4
channel->writeRegister(0x06, 0x5A4A80C4);
std::this_thread::sleep_for(1ms);

// SEND STBWR CMD
channel->writeRegister(0x06, 0x614080D4);
getLogger() << "- Loading Pedestals and Thresholds from file /tmp/
   thr_1.dat" << InfoLogger::endm;
std::this_thread::sleep_for(1ms);

ifstream PedThr_file_1("/tmp/thr_1.dat");
if (PedThr_file_1.is_open())
{
    while (getline(PedThr_file_1, line))
    {
        PedThr = stoi(line, 0, 16);
        channel->writeRegister(0x5A, PedThr);
        std::this_thread::sleep_for(1ms);
    }
    // SEND EOBTR CMD
    channel->writeRegister(0x06, 0x614080B4);
    std::this_thread::sleep_for(1ms);

    // ddlSendCommand: 0x61608044
    getLogger() << "- ddlSendCommand: 0x61608044 - sending
       pedestals and thresholds to Column 1" << InfoLogger::endm;
```

```
    channel->writeRegister(0x06, 0x61608044);
    std::this_thread::sleep_for(10ms);

    // ddlSendCommand: 0x314E80C4
    getLogger() << "- ddlSendCommand: 0x314E80C4" << InfoLogger::
       endm;
    channel->writeRegister(0x06, 0x314E80C4);
    std::this_thread::sleep_for(1ms);

    PedThr_file_1.close();
}
else getLogger() << "- Unable to open file /tmp/thr_1.dat" <<
   InfoLogger::endm;
```

After segment selection is complete, the column is selected by sending the command "0x614080D4", which indicates the start of the block write. The pedestals and thresholds for the column are loaded from a text file and each line is sent to the FIFO in the RCB. The FESTRD command is then sent to signal to the RCB that all pedestals and thresholds have been transferred. Another command is then received ("0x314E80C4") to save the configuration of the pedestals and thresholds in the columns. This process is repeated for all 24 columns in the 3 segments.

Another script was written to enable zero suppression. This script selects each segment and column, and sends the command "0x3X15a80C4", where the "X" denotes the column number. After zero suppression is turned on, data taking can take place.

5.3.3 Performance

The BUSY time is used to measure the performance of the system - the readout rate. This is the period in which the BUSY signal goes high, upon reception of the L0 trigger, until the BUSY signal goes down at the end of an event. This

transmission time depends on the time between L0, L1, and the transmission time of the event, which depends on the number of words. Thus, the data that is sent to the DAQ consists of a fixed part which consists of the various headers and markers, and a variable part which depends on the amount of data to be transmitted. The fixed part consists of the following:

- Raw Data Header (16 words, 64 bytes)
- Column Headers (24 words, 96 bytes)
- DILOGIC Markers (240 words, 960 bytes), and
- Segment Markers (3 words, 12 bytes).

Since a fixed paging mechanism is being used, where each segment is divided into two pages, an additional five RDH are being sent as well to denote the start of a new page. A complete event readout is shown in Figure 5.9. After the L0 trigger is received, the BUSY signal is asserted. When the second L1A trigger is received, the RDH is sent, and data transfer starts. It can be seen that a whole event, with an occupancy of 0%, can be read in 65.62 µs. This is an improvement over the old firmware [77], where the minimum time to readout a single event was 214 µs, corresponding to a readout rate of 4.67 kHz. This result is also an improvement over the first prototype of the upgraded firmware [75], where the minimum readout time was 99.8 µs (10.21 kHz).

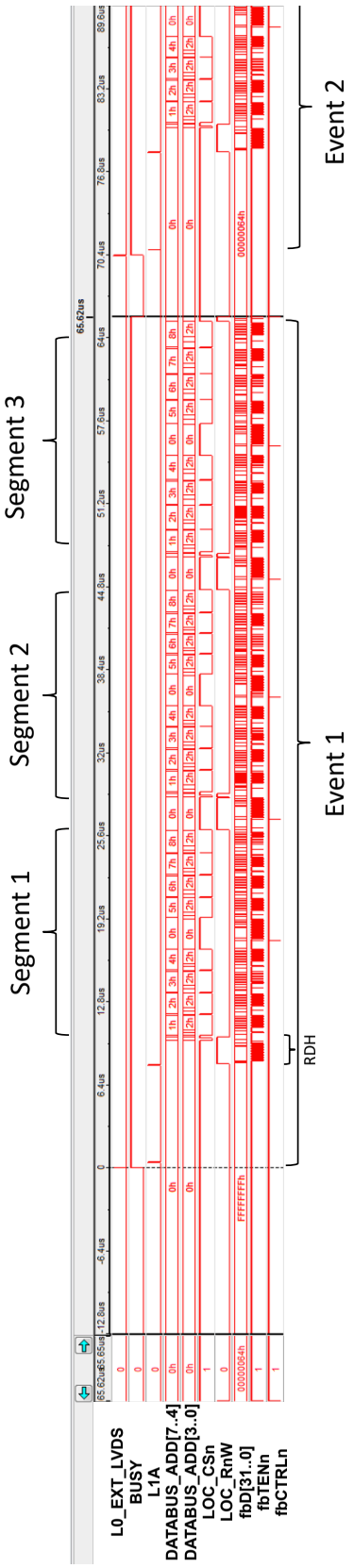
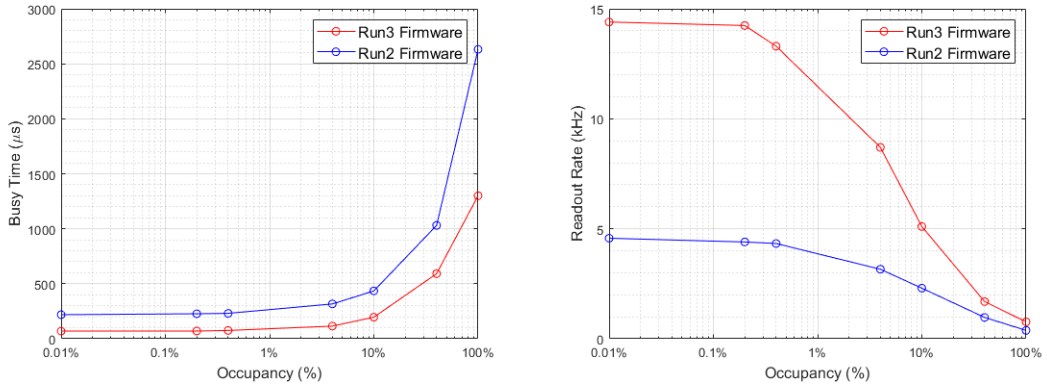


Figure 5.9: Complete Event Readout for an occupancy of 0%



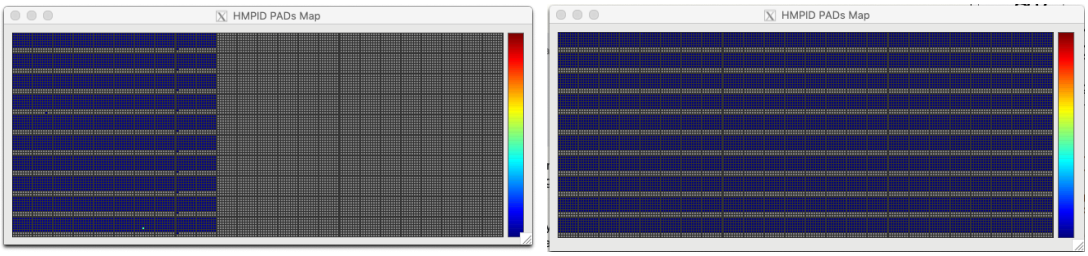
(a) Busy time (lower is better) **(b)** Readout rate (higher is better)

Figure 5.10: Comparison between the busy time and event readout rate of the firmware used in Run2 and this work

Further testing was performed at different occupancy levels which was done by enabling/disabling channels, through the uploading of channel-specific pedestals and thresholds. The variation of the busy time, which conditions the event rate, are illustrated in figure 5.10a and are compared with those from the firmware version of Run2 [77]. Figure 5.10b illustrates the increase in the readout rate, when compared to the previous version of the firmware. In general, a 200% improvement over the readout rate achieved in the previous version of the firmware [77] has been achieved. To verify that there has indeed been no loss of data integrity, the **hmpDisplayMap** tool was used. Figures 5.11a and 5.11b illustrate the occupancy at 40% and 100% occupancy and show that there indeed has been a preservation of data integrity, when operating at a higher readout rate.

5.4 Conclusion

This chapter has presented the necessary upgrades which were performed on the firmware. This was done for two reasons; to handle the new triggering mechanism, and to handle the new Online-Offline Data Acquisition software which will be used in Run3.



(a) HMPID Event with 40% Occupancy **(b)** HMPID Event with 100% occupancy

Figure 5.11: HMPID Raw Data Display Mapper, showing HMPID events at different levels of occupancy

Two approaches were adopted to achieve this. The first was to modify the existing firmware to remove the dependency on the L2 trigger, decode correctly the trigger message, and to send data to the new C-RORC card in the DAQ system. Once this was tested, and verified that it was working correctly, a new firmware, written completely in VHDL, was developed to improve the data transfer rate between the SIU and the DAQ system, without loosing any data integrity. In fact, through this new firmware HMPID is now capable of operating with a maximum readout rate of 14.4 kHz, which is an improvement over the existing firmware.

6. Conclusions and Perspective

This work has presented the design, simulation, implementation, and testing of a novel remotely configurable delay generator, which will be used in the HMPID, as part of the LHC-wide upgrade in preparation for Run3 (2021 - 2023). While during runs 1 and 2, the detectors in ALICE made use of three trigger signals (L0, L1, and L2), during the forthcoming run the triggering mechanism will change. In the previous runs, HMPID was triggered by the L0 trigger, which arrives at 1.2 µs after an event. This directly corresponds to the peak of the charge on the pads of HMPID. In Run3, HMPID will no longer be utilising the L0 trigger, but instead will be using the LM trigger which arrives at around 800 ns after an event. This therefore required the design of a new remotely configurable delay generator. In addition, due to changes in the triggering scheme, the HMPID read-out firmware, had to be upgraded to handle the arrival of the new high-level triggers. These tasks were carried out as part of this doctoral research presented in this dissertation. Furthermore, this work has been published in five peer reviewed conference proceedings and a journal, thus showing the validity of this research.

In summary, the main contributions of this work have been:

- Design and implementation of a remotely configurable digital delay generator, that has been implemented on an FPGA and verified to have a range of 525 ns, and a resolution of 1 ns.

- Design, implementation, and testing of a novel remotely configurable delay generator that is based on a DLL architecture which, once calibrated, can be used in open-loop configuration to generate the required delays. While the architecture itself is novel, various techniques have been used to optimise the design of a symmetric operation, quasi-linear, rail-to-rail delay element which has a range from 183 ns to 935 ns, and can be varied in steps of 2 ns. Three types of techniques were used to linearise the delay response of the delay element, with a range that can be extended through the use of a switched capacitor bank. The first two optimisation techniques are based on approximating the time delay equation through the use of a second-order polynomial, and then minimising the second-order term. The third method is based on the use of the particle swarm optimisation techniques to find the transistor aspect ratios giving the most-linear response for the required range and resolution. The complete architecture has been fabricated using the X-FAB XH018 technology. While the delay response achieved is quite linear when using the 0.5 pF and 1 pF capacitances, this was not the case when enabling the 1.5 pF capacitor due to the amplification of non-linearities. However, it was confirmed that it does not affect the application. The complete architecture was tested and it was shown that the DLL can indeed be used to generate the required delay, and then used in open-loop configuration. The total measured power consumption of the delay generator ASIC, including the ADC and DAC circuits, is 6 mW from a 1.8 V supply, whereas the total power consumption of the test board (including the ASIC) is approximately 200 mW.

- The design of a new firmware architecture which will be used in HMPID to handle the new triggering mechanism, and to successfully integrate HMPID in the new online-offline data acquisition software. The new firmware, written completely in VHDL, was developed to improve the data transfer rate between

the SIU and the DAQ system. In fact, through this new firmware HMPID is now capable of operating with a maximum readout rate of 14.4 kHz, which is an improvement over the firmware used during runs 1 and 2 where the maximum readout rate was 4.6 kHz. This was achieved without any loss of data.

6.1 Future Work

The work presented in this dissertation can be extended further, with the following a list of some of the possible research avenues:

- Design and investigation of a replica feedback loop in the DLL architecture, to perform online compensation in the delay generator architecture.
- Integration of 14 delay generators on a single ASIC, such that each channel can be programmed individually and sent to each HMPID module.
- Design and implementation of a single FPGA-based board that can receive commands from the IPBus protocol, to provide remote control and monitoring of the delay.
- Modelling and investigation of the affect of noise, using various RF models, on the DLL which may affect the calibration stage of the delay generator.
- Improve further the firmware of HMPID to implement dynamic paging and therefore further increase the readout rate.
- Implementation of the Heartbeat trigger accept/reject handler in the firmware for full compliance with Run3 specifications which, at the time of writing, are still under discussion.

References

[1] B. Abelev, ALICE collaboration, *et al.*, "Upgrade of the ALICE experiment: letter of intent," *Journal of Physics G: Nuclear and Particle Physics*, vol. 41, no. 8, p. 87001, 2014.

[2] K. Aamodt, A. A. Quintana, R. Achenbach, S. Acounis, D. Adamová, C. Adler, M. Aggarwal, F. Agnese, G. A. Rinella, Z. Ahammed, *et al.*, "The ALICE experiment at the CERN LHC," *Journal of Instrumentation*, vol. 3, no. 8, p. S08002, 2008.

[3] E. Mobs, "The CERN accelerator complex. Complexe des accélérateurs du CERN," Jul 2016. General Photo.

[4] F. Piuz, W. Klempt, L. Leistam, J. De Groot, and J. Schükraft, *ALICE high-momentum particle identification: Technical Design Report*. Technical Design Report ALICE, Geneva: CERN, 1998.

[5] J. L. Gauci, E. Gatt, O. Casha, ALICE Collaboration, *et al.*, "Preparing the ALICE-HMPID RICH for the high-luminosity LHC period 2021–2023," *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, 2019.

[6] J.-C. Santiard and K. Marent, "The Gassiplex0.7-2 integrated front-end analog processor for the HMPID and the dimuon spectrometer of ALICE,"

tech. rep., CERN, 2001.

[7] P. Antonioli, A. Kluge, and W. Riegler, "Upgrade of the ALICE Readout & Trigger System," Tech. Rep. CERN-LHCC-2013-019. ALICE-TDR-015, CERN, Sept. 2013.

[8] P. Napolitano, A. Moschitta, and P. Carbone, "A survey on time interval measurement techniques and testing methods," in *2010 IEEE Instrumentation & Measurement Technology Conference Proceedings*, pp. 181–186, IEEE, 2010.

[9] P. Andreani, F. Bigongiari, R. Roncella, R. Saletti, and P. Terreni, "A digitally controlled shunt capacitor cmos delay line," *Analog Integrated Circuits and Signal Processing*, vol. 18, no. 1, pp. 89–96, 1999.

[10] K. Sakamoto, J. McDonald, M. Swapp, and B. Weir, "A digitally programmable delay chip with picosecond resolution," in *Proceedings of the bipolar circuits and technology meeting*, pp. 295–297, IEEE, 1989.

[11] N. H. Weste and D. Harris, *CMOS VLSI design: a circuits and systems perspective*. Pearson Education India, 2015.

[12] M. A. Abas, G. Russell, and D. Kinniment, "Built-in time measurement circuits–a comparative design study," *IET Computers & Digital Techniques*, vol. 1, no. 2, pp. 87–97, 2007.

[13] B. Abdulrazzaq, I. A. Halin, S. Kawahito, R. M. Sidek, S. Shafie, and N. A. Yunus, "A Review on High-Resolution CMOS Delay Lines: Towards sub-picosecond Jitter Performance," *SpringerPlus*, vol. 5, no. 1, pp. 1–32, 2016.

[14] M. Maymandi-Nejad and M. Sachdev, "A Monotonic Digitally Controlled Delay Element," *IEEE Journal of Solid-State Circuits*, vol. 40, no. 11,

pp. 2212–2219, 2005.

[15] J. L. Gauci, E. Gatt, G. De Cataldo, O. Casha, and I. Grech, "An Analytical Model of the Delay Generator for the Triggering of Particle Detectors at CERN LHC," in *CAS (NGCAS), 2017 New Generation of*, pp. 69–72, IEEE, 2017.

[16] A. Melloni, A. Canciamilla, C. Ferrari, F. Morichetti, L. O'Faolain, T. Krauss, R. De La Rue, A. Samarelli, and M. Sorel, "Tunable delay lines in silicon photonics: coupled resonators and photonic crystals, a comparison," *IEEE Photonics Journal*, vol. 2, no. 2, pp. 181–194, 2010.

[17] P. A. J. Nuyts, P. Reynaert, and W. Dehaene, *Continuous-Time Digital Design Techniques*, pp. 125–185. Cham: Springer International Publishing, 2014.

[18] M. R. Jan, C. Anantha, and N. Borivoje, "Digital integrated circuits: a design perspective," 2003.

[19] N. R. Mahapatra, A. Tareen, and S. V. Garimella, "Comparison and analysis of delay elements," in *Circuits and Systems, 2002. MWSCAS-2002. The 2002 45th Midwest Symposium on*, vol. 2, pp. II–473, IEEE, 2002.

[20] S. Schidl, K. Schweiger, W. Gaberl, and H. Zimmermann, "Analogously tunable delay line for on-chip measurements with sub-picosecond resolution in 90 nm cmos," *Electronics Letters*, vol. 48, pp. 910–911, July 2012.

[21] C.-K. K. Yang, "Delay-locked loops–an overview," *Phase-Locking in High-Peformance Systems, Wiley-IEEE Press, New York, NY*, pp. 13–22, 2003.

[22] B. Markovic, S. Tisa, F. A. Villa, A. Tosi, and F. Zappa, "A high-linearity, 17 ps precision time-to-digital converter based on a single-stage vernier delay

loop fine interpolation," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 60, no. 3, pp. 557–569, 2013.

[23] G. Jovanović and M. Stojčev, "Current starved delay element with symmetric load," *International journal of electronics*, vol. 93, no. 3, pp. 167–175, 2006.

[24] R. J. Baker, *CMOS: circuit design, layout, and simulation*, vol. 1. John Wiley & Sons, 2008.

[25] N. R. Mahapatra, S. V. Garimella, and A. Tareen, "An empirical and analytical comparison of delay elements and a new delay element design," in *VLSI, 2000. Proceedings. IEEE Computer Society Workshop on*, pp. 81–86, IEEE, 2000.

[26] A. El Mourabit, G.-N. Lu, P. Pittet, Y. Birjali, and F. Lahjomri, "Low power, high resolution cmos variable-delay element," *AEU-International Journal of Electronics and Communications*, vol. 66, no. 6, pp. 455–458, 2012.

[27] R. Zhang and M. Kaneko, "Robust and Low-Power Digitally Programmable Delay Element Designs Employing Neuron-MOS Mechanism," *ACM Transactions on Design Automation of Electronic Systems*, vol. 20, no. 4, pp. 1–19, 2015.

[28] J. M. Rabaey, A. P. Chandrakasan, and B. Nikolic, *Digital integrated circuits*, vol. 2. Prentice hall Englewood Cliffs, 2002.

[29] W. C. Elmore, "The transient response of damped linear networks with particular regard to wideband amplifiers," *Journal of applied physics*, vol. 19, no. 1, pp. 55–63, 1948.

[30] S. Eto, H. Akita, K. Isobe, K. Tsuchida, H. Toda, and T. Seki, "A 333 mhz, 20 mw, 18 ps resolution digital dll using current-controlled delay with parallel

variable resistor dac (pvr-dac)," in *ASICs, 2000. AP-ASIC 2000. Proceedings of the Second IEEE Asia Pacific Conference on*, pp. 349–350, IEEE, 2000.

[31] G. Kim, M.-K. Kim, B.-S. Chang, and W. Kim, "A low-voltage, low-power cmos delay element," *IEEE Journal of Solid-State Circuits*, vol. 31, no. 7, pp. 966–971, 1996.

[32] M. Maymandi-Nejad and M. Sachdev, "A Digitally Programmable Delay Element: Design and Analysis," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 11, no. 5, pp. 871–878, 2003.

[33] S. B. Kobenge and H. Yang, "Digitally controllable delay element using switched-current mirror," *WSEAS Transaction on Circuit and Systems*, no. 7, pp. 599–608, 2009.

[34] N. Angeli and K. Hofmann, "A low-power and area-efficient digitally controlled shunt-capacitor delay element for high-resolution delay lines," in *2018 25th IEEE International Conference on Electronics, Circuits and Systems (ICECS)*, pp. 717–720, IEEE, 2018.

[35] O. Z. Batur, N. Pekçokgüler, G. Dündar, and M. Koca, "A high resolution and low jitter linear delay line for ir-uwb template pulse synchronization," in *Circuit Theory and Design (ECCTD), 2015 European Conference on*, pp. 1–4, IEEE, 2015.

[36] "IEEE Standard for A Versatile Backplane Bus: VMEbus," standard, ANSI/IEEE.

[37] J. Mauricio, D. Gascón, X. Vilasís, E. Picatoste, F. Machefert, J. Lefrancois, O. Duarte, and C. Beigbeder, "Radiation hard programmable delay line for lhcb calorimeter upgrade," *Journal of Instrumentation*, vol. 9, no. 1, p. C01016, 2014.

[38] M. Krivda, "Fanin\Fanout Unit," Aug. 2007.

[39] A. Aloisio, R. Lomoro, S. Loffredo, V. Izzo, P. Branchini, R. Cicalese, and R. Giordano, "High-resolution time-to-digital converter in field programmable gate array," 2008.

[40] A. Aloisio, P. Branchini, R. Giordano, V. Izzo, and S. Loffredo, "High-precision time-to-digital converter in a fpga device," in *Nuclear Science Symposium Conference Record (NSS/MIC), 2009 IEEE*, pp. 290–294, IEEE, 2009.

[41] J. Song, Q. An, and S. Liu, "A high-resolution time-to-digital converter implemented in field-programmable-gate-arrays," *IEEE Transactions on Nuclear Science*, vol. 53, no. 1, pp. 236–241, 2006.

[42] J. Kalisz, "Review of methods for time interval measurements with picosecond resolution," *Metrologia*, vol. 41, no. 1, p. 17, 2003.

[43] T.-I. Otsuji, "A picosecond-accuracy, 700-mhz range, si bipolar time interval counter lsi," *IEEE journal of solid-state circuits*, vol. 28, no. 9, pp. 941–947, 1993.

[44] J. Aveynier and R. Van Zurk, "Vernier chronotron reflex," *Nuclear Instruments and Methods*, vol. 78, no. 1, pp. 161–170, 1970.

[45] Z. Cheng, X. Zheng, M. J. Deen, and H. Peng, "Recent developments and design challenges of high-performance ring oscillator CMOS time-to-digital converters," *IEEE Transactions on Electron Devices*, vol. 63, no. 1, pp. 235–251, 2015.

[46] J. Kalisz, R. Szplet, J. Pasierbinski, and A. Poniecki, "Field-programmable-gate-array-based time-to-digital converter with 200-ps

resolution," *IEEE Transactions on Instrumentation and Measurement*, vol. 46, no. 1, pp. 51–55, 1997.

[47] J. Wang, S. Liu, Q. Shen, H. Li, and Q. An, "A fully fledged tdc implemented in field-programmable gate arrays," *IEEE Transactions on Nuclear Science*, vol. 57, no. 2, pp. 446–450, 2010.

[48] C. Hervé, J. Cerrai, and T. Le Caër, "High resolution time-to-digital converter (TDC) implemented in field programmable gate array (FPGA) with compensated process voltage and temperature (PVT) variations," *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, vol. 682, pp. 16–25, 2012.

[49] J. Torres, A. Aguilar, R. Garcia-Olcina, P. Martı, J. Martos, J. Soret, J. Benlloch, P. Conde, A. Gonzalez, F. Sanchez, *et al.*, "Time-to-digital converter based on FPGA with multiple channel capability," *IEEE Transactions on Nuclear Science*, vol. 61, no. 1, pp. 107–114, 2013.

[50] J. L. Gauci, E. Gatt, O. Casha, G. De Cataldo, and I. Grech, "On the Design of a Linear Delay Element for the Triggering Module at CERN LHC," in *14th IEEE Conference on PhD Research in Microelectronics and Electronics (PRIME)*, 2018.

[51] J. L. Gauci, E. Gatt, O. Casha, G. De Cataldo, I. Grech, and J. Micallef, "Design of a Quasi-Linear Rail-to-Rail Delay Element with an Extended Programmable Range," in *2018 25th IEEE International Conference on Electronics, Circuits and Systems (ICECS)*, pp. 265–268, IEEE, 2018.

[52] J. L. Gauci, E. Gatt, O. Casha, G. De Cataldo, I. Grech, and J. Micallef, "Particle Swarm Optimization of a Rail-to-Rail Delay Element for Maximum

Linearity," in *2019 6th International Conference on Control, Decision and Information Technologies (CoDIT)*, pp. 420–424, IEEE, 2019.

[53] J. L. Gauci, E. Gatt, O. Casha, G. De Cataldo, I. Grech, and J. Micallef, "A Novel Delay Generator for the Triggering of Particle Detectors at the CERN LHC," in *2019 15th Conference on Ph. D Research in Microelectronics and Electronics (PRIME)*, pp. 33–36, IEEE, 2019.

[54] T. Xanthopoulos, "Digital delay lock techniques," in *Clocking in Modern VLSI Systems*, pp. 183–244, Springer, 2009.

[55] C. Jia, *A delay-locked loop for multiple clock phases/delays generation*. PhD thesis, Georgia Institute of Technology, 2005.

[56] C. Jia and L. Milor, "A dll design for testing i/o setup and hold times," *IEEE transactions on very large scale integration (VLSI) systems*, vol. 17, no. 11, pp. 1579–1592, 2009.

[57] H.-H. Chang, J.-W. Lin, C.-Y. Yang, and S.-I. Liu, "A wide-range delay-locked loop with a fixed latency of one clock cycle," *IEEE journal of solid-state circuits*, vol. 37, no. 8, pp. 1021–1027, 2002.

[58] M. Moazedi, A. Abrishamifar, and A. Sodagar, "A highly-linear modified pseudo-differential current starved delay element with wide tuning range," in *Electrical Engineering (ICEE), 2011 19th Iranian Conference on*, pp. 1–4, IEEE, 2011.

[59] Y.-J. Jung, S.-W. Lee, D. Shim, W. Kim, C. Kim, and S.-I. Cho, "A dual-loop delay-locked loop using multiple voltage-controlled delay lines," *IEEE Journal of solid-state circuits*, vol. 36, no. 5, pp. 784–791, 2001.

[60] J. Choi, S. T. Kim, W. Kim, K.-W. Kim, K. Lim, and J. Laskar, "A low power and wide range programmable clock generator with a high

multiplication factor," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 19, no. 4, pp. 701–705, 2011.

[61] "Europractice-IC - ASICS [Online]."

[62] B. Razavi, *RF microelectronics*. Pearson Education International, 2013.

[63] E. Zafarkhah, M. Maymandi-Nejad, and M. Zare, "Improved accuracy equation for propagation delay of a CMOS inverter in a single ended ring oscillator," *AEU-International Journal of Electronics and Communications*, vol. 71, pp. 110–117, 2017.

[64] H. Rivandi, S. Ebrahimi, and M. Saberi, "A low-power rail-to-rail input-range linear delay element circuit," *AEU-International Journal of Electronics and Communications*, vol. 79, pp. 26–32, 2017.

[65] J. Lambers, "Mat 772 - fall semester 2010-11 lecture 5 notes - lagrange interpolation." Lecture Notes, 2010.

[66] H. M. College, "The newton polynomial interpolation." `http://fourier.eng.hmc.edu/e176/lectures/ch7/node4.html`. Accessed: 30 August 2019.

[67] S. V. Kumar, P. Rao, H. Sharath, B. Sachin, U. Ravi, and B. Monica, "Review on VLSI design using optimization and self-adaptive particle swarm optimization," *Journal of King Saud University-Computer and Information Sciences*, 2018.

[68] R. Hassan, B. Cohanim, O. De Weck, and G. Venter, "A comparison of particle swarm optimization and the genetic algorithm," in *46th AIAA/ASME/ASCE/AHS/ASC structures, structural dynamics and materials conference*, p. 1897, 2005.

[69] P. Kumar and K. Duraiswamy, "An optimized device sizing of analog circuits using particle swarm optimization," in *Proceedings of IEEE International Conference on Neural Networks, Nov. 27-Dec. 1, IEEE Xplore*, Citeseer, 2012.

[70] Maxim Integrated Products, "MAX5530/MAX5531 Ultra-Low-Power, 12-Bit, Voltage-Output DACs (Datasheet)."

[71] G. Jovanovic, M. Stojcev, and D. Krstic, "Delay locked loop with linear delay element," in *TELSIKS 2005-2005 uth International Conference on Telecommunication in ModernSatellite, Cable and Broadcasting Services*, vol. 2, pp. 397–400, IEEE, 2005.

[72] F. Piuz, W. Klempt, L. Leistam, J. De Groot, and J. Schükraft, *Detector for High Momentum PID - ALICE Technical Design Report.* Geneva: CERN, 1998.

[73] J. Magri, "High-Level Test Bench Design for High-Speed Multi-FPGA Pipelines," master's thesis, University of Malta, 2018.

[74] J. Magri, O. Casha, K. Bugeja, I. Grech, and E. Gatt, "HLTB design for high-speed multi-FPGA pipelines," in *2017 24th IEEE International Conference on Electronics, Circuits and Systems (ICECS)*, pp. 182–185, IEEE, 2017.

[75] S. Calleja, "Implementation of the readout control firmware upgrade for the CERN Large Ion Collider Experiment," master's thesis, University of Malta, 2016.

[76] G. Rubin, J. Sulyán, N. Tóth, C. Soós, P. Van de Vyvre, I. Novák, P. Csató, D. Tarján, A. Telegdy, G. Vesztergombi, *et al.*, "The ALICE detector data link," tech. rep., CERN, 1999.

[77] L. M. Minervini and HMPID Group, "The architecture, performance and perspective of the ALICE-HMPID RICH read-out," tech. rep., HMPID - ALICE - CERN, 2015.

[78] P. Moreira *et al.*, "TTCrx Reference Manual," *CERN, Internal document*, 2003.

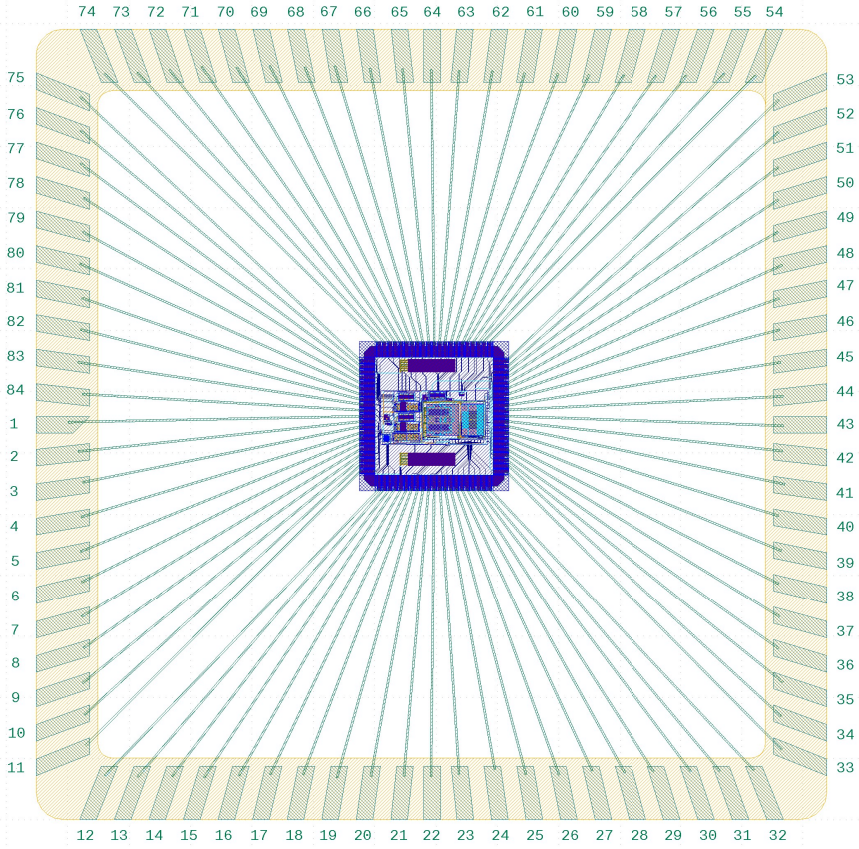
A. Chip Bonding Diagram, and Manufactured Samples

A.1 Bonding Diagram

Request: JLG

LCC 84

Ceramic Leadless Chip Carrier



Comment:

Max. wire length :

Pad-pitch :

Pad-opening:

Die-thickness:

Extra Info:

MPW: 6432	Date: 29/10/2018	Scale
Die: Chip_Pads	Size incl scribe: 2.4 mm x 2.4 mm	10
Qty packaged: 20	Lid: Taped x ☐ Sealed x ☐ Glued ☐ Glass ☐	
Qty naked: What remains		
Die Attach: Default		
Wire: Default		

IC Service

Coordinated by IMEC

A.2 Manufactured Samples

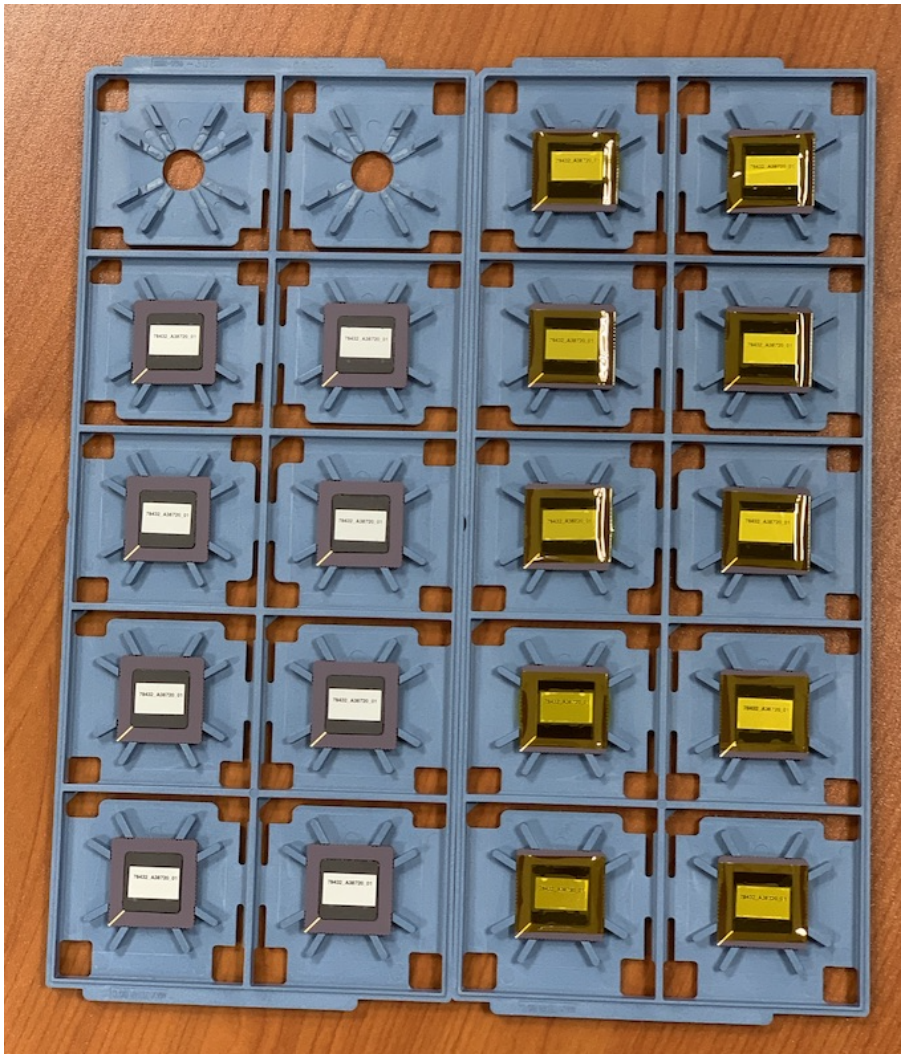


Figure A.1: 20 manufactured samples (2 not shown)

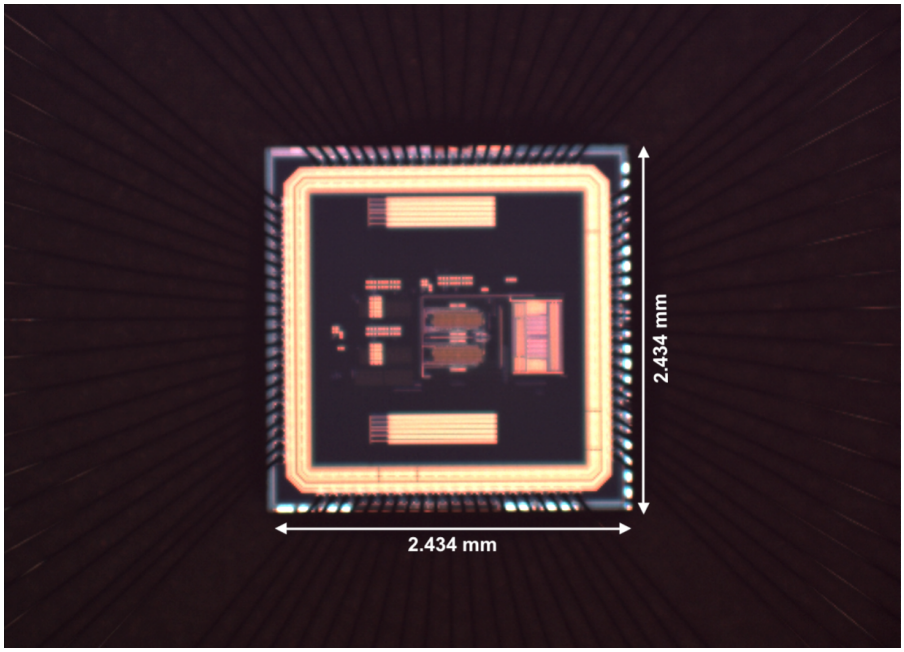


Figure A.2: Microscope image of manufactured ASIC with 2x magnification

B. PCB Schematic and Layout

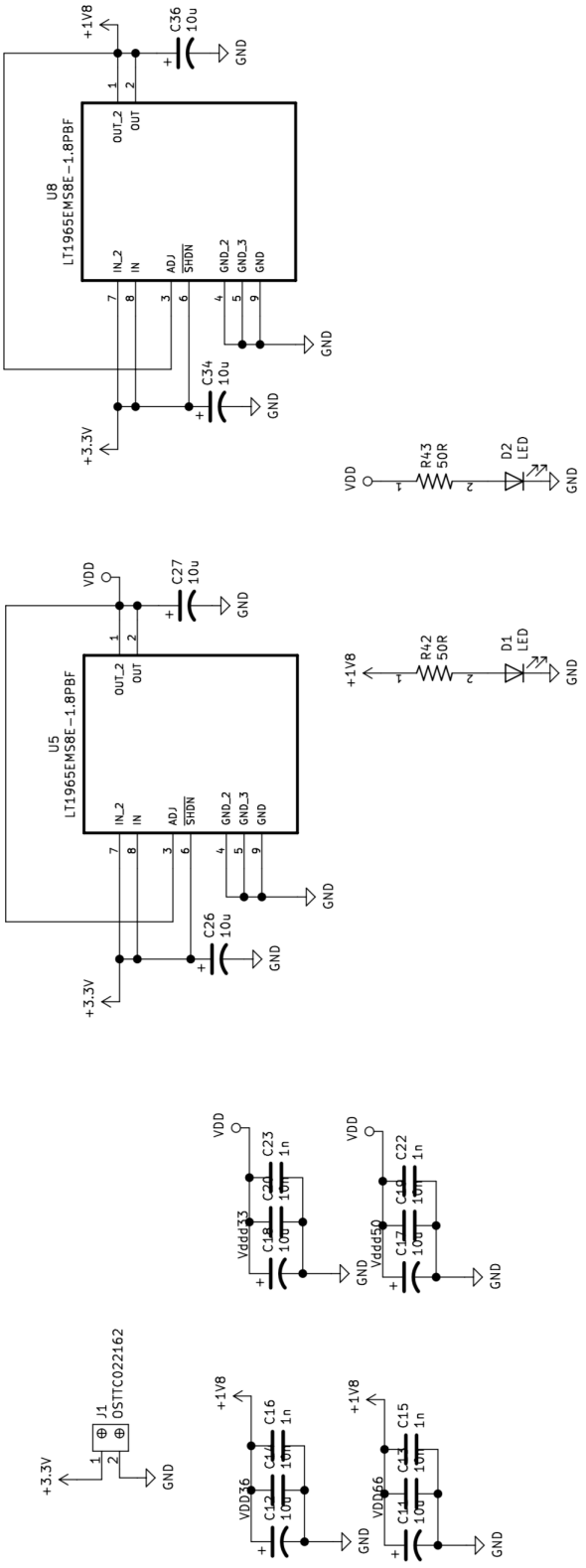


Figure B.1: Excerpt from PCB Schematic - Power

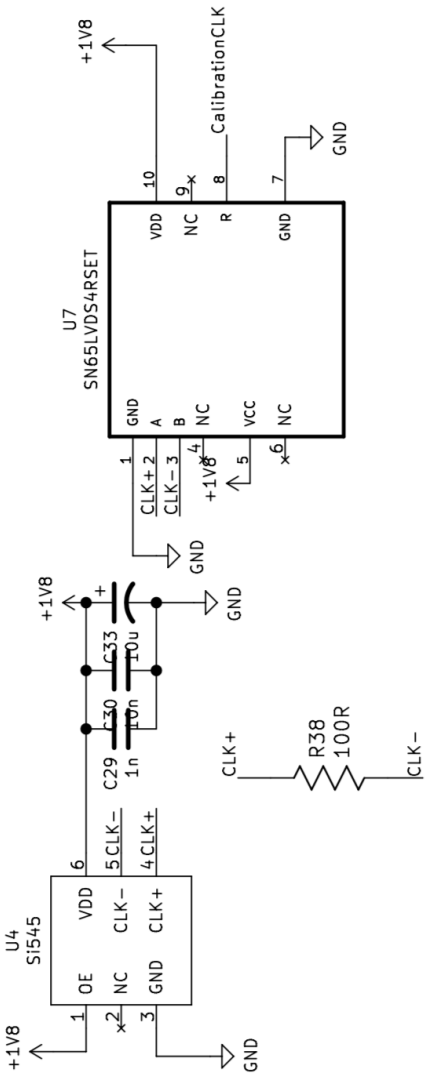


Figure B.2: Excerpt from PCB Schematic - 500 MHz clock generator

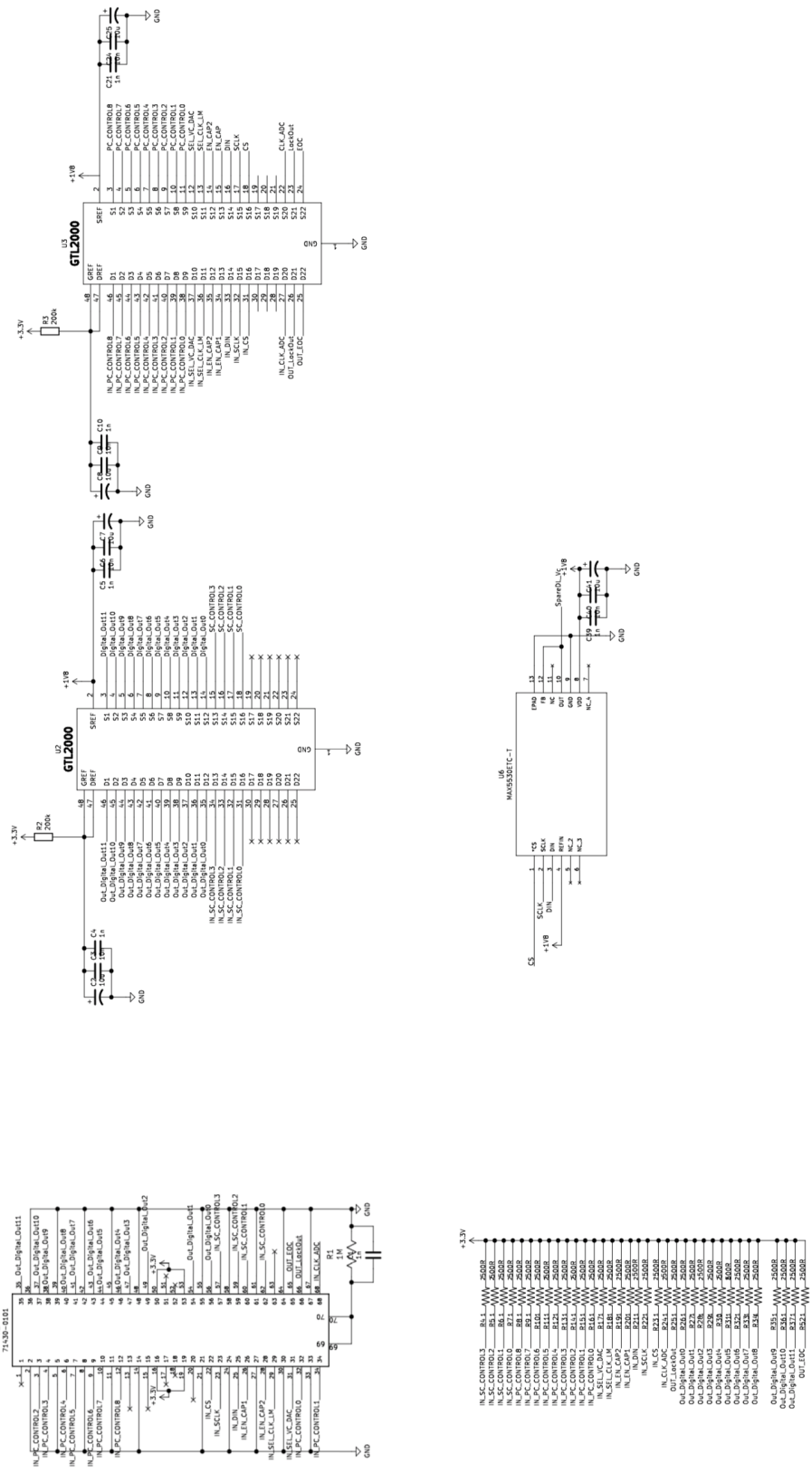


Figure B.3: Excerpt from PCB Schematic - FPGA Interface

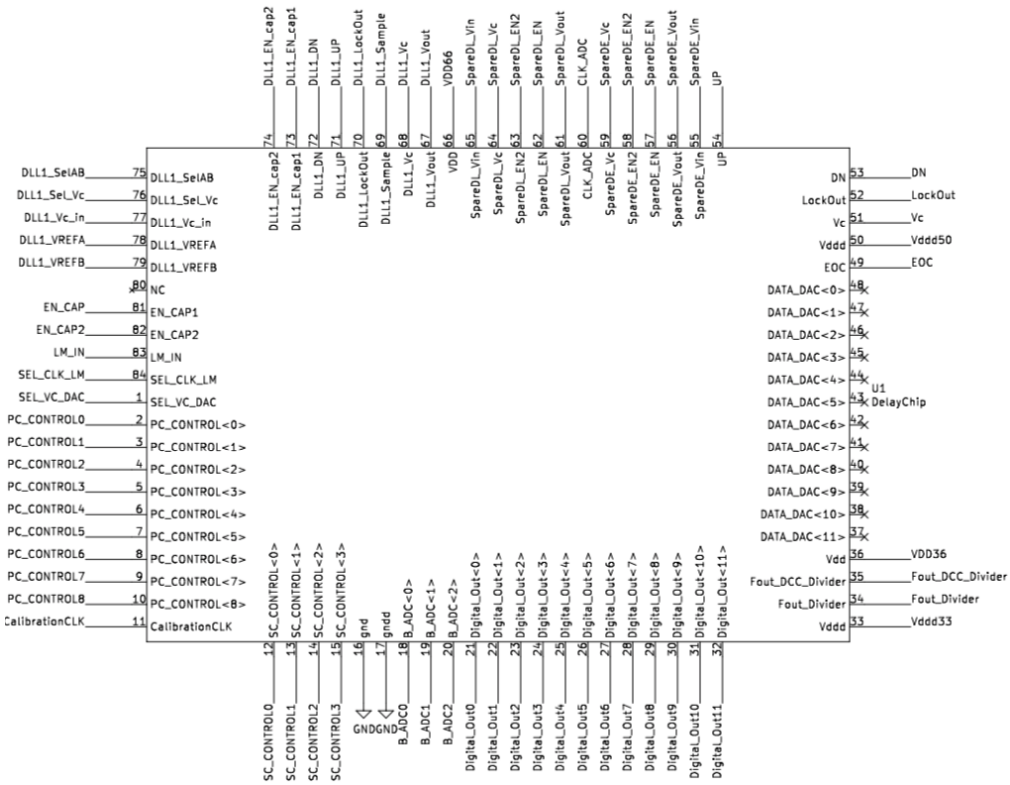
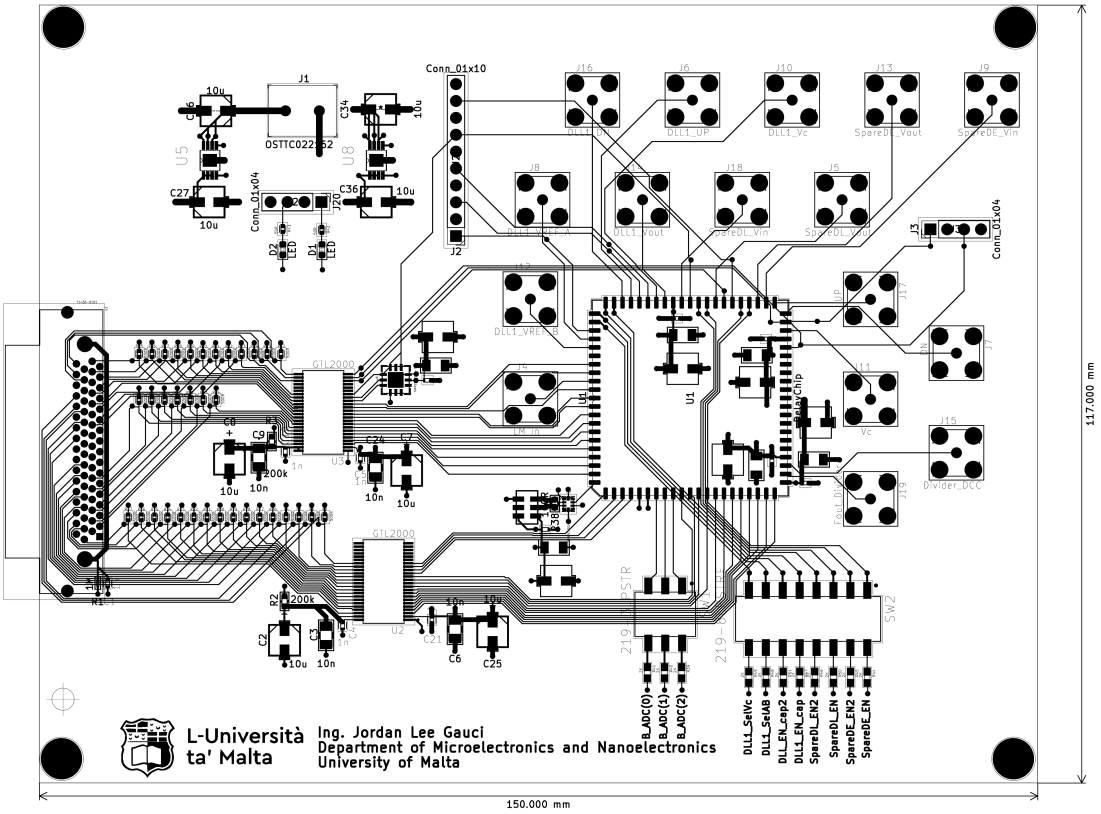


Figure B.4: Excerpt from PCB Schematic - ASIC Connections

B.1 PCB Layout



B.2 Bill of Materials

Quantity	Manufacturer	Part Number	Description	Package
2	Linear Technology	LT1965EMS8E-1.8#PBF	Voltage Regulators 3.3 to 1.8 V	8-MSOP-EP
16	Amphenol	132134-15	SMA Connectors	SMA Through Hole
14	Nichicon	UWX1C100MCL2GB	10 uF Capacitors	Radial, Can - SMD
2	Wurth Electronics Inc.	61300411121	4-Pin Header (2.54 mm pitch)	Through Hole (2.54 mm pitch)
35	Vishay Dale	PNM0402E5000BST1	500R Resistors	0402 (1005 Metric)
1	Molex	71430-0101	VHDCi Connector	Through Hole VHDCi Connector
1	Texas Instruments	SN65LVDS4RSET	LVDS Receiver	10-UFQFN
1	Maxim Integrated	MAX5530ETC+	12-Bit DAC	12-TQFN (4x4)
1	Silicon Labs	545BAA500M000BAG	SI545 - 500 MHz Clock Generator	6-SMD, No Lead 0.197" x 0.126" W
2	NXP USA Inc.	GTL2000DGG,118	GTL2000 - Voltage Translators	TSSOP48
1	CTS Electrocomponents	219-8LPSTRF	8 SWITCH DIP	Surface Mount
1	CTS Electrocomponents	219-3LPSTR	3 SWITCH DIP	Surface Mount
11	Panasonic Electronic Components	ERA-3AEB202V	2K Resistors	0603 (1608 Metric)
2	Vishay Thin Film	FC0402E50R0BST1	50R Resistors	0402 (1005 Metric)
1	Rohm Semiconductor	ESR03EZPJ101	100R Resistor	0603 (1608 Metric)
2	Panasonic Electronic Components	ERA-3AEB204V	200K Resistor	0603 (1608 Metric)
1	Yageo	RC0201FR-071ML	1M Resistor	0201 (0603 Metric)
1	Wurth Electronics Inc.	61301011121	10-Pin Header	Through Hole (2.54 mm pitch)
1	On Shore Technology Inc.	OSTTC022162	Terminal Block	Through Hole Terminal Block
2	OSRAM Opto Semiconductors Inc.	LG L29K-F2J1-24-Z	LEDs	0603 (1608 Metric)
10	Panasonic Electronic Components	ECH-U1H103JX5	10 nF Capacitors	1206 (3216 Metric)
11	Samsung Electro-Mechanics	CL05B102KB5NNNC	1 nF Capacitors	0402 (1005 Metric)

Table B.1: List of Components Used in PCB

B.3 Assembled PCB

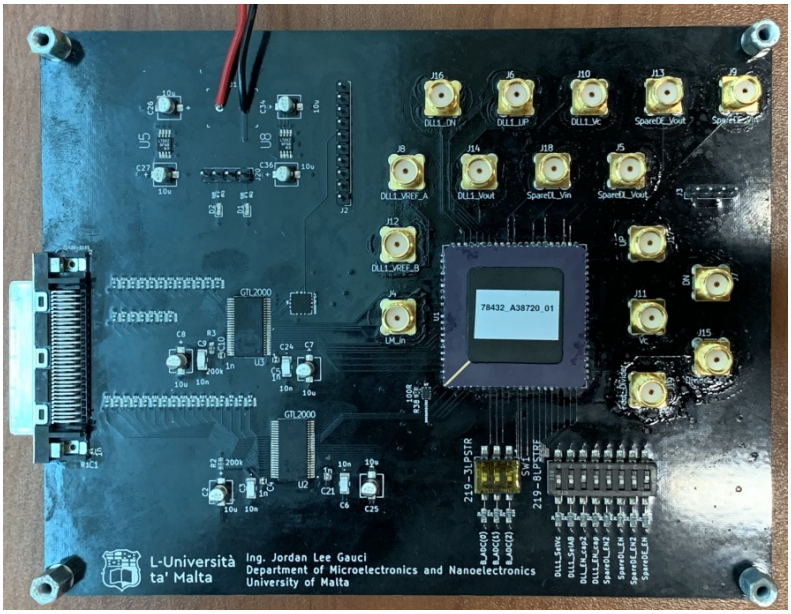


Figure B.5: Top view of PCB

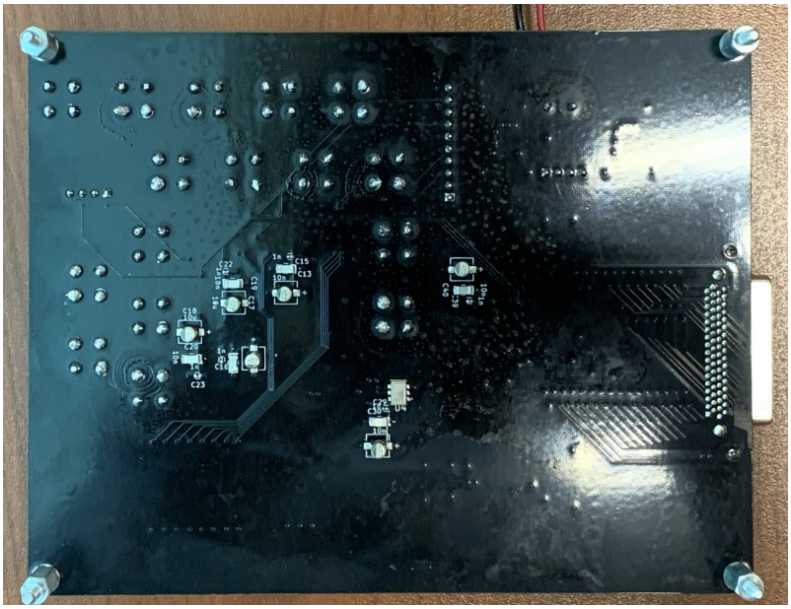
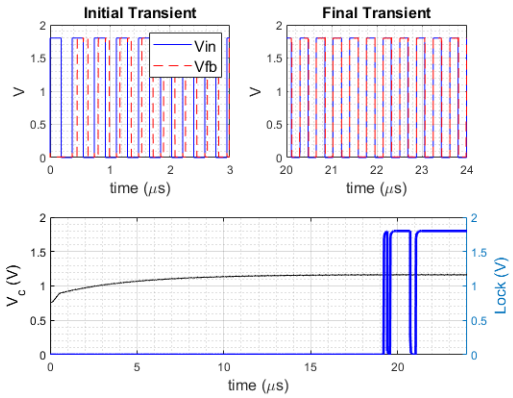
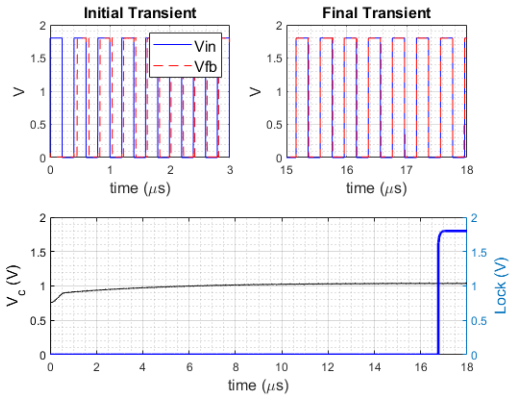


Figure B.6: Bottom view of PCB

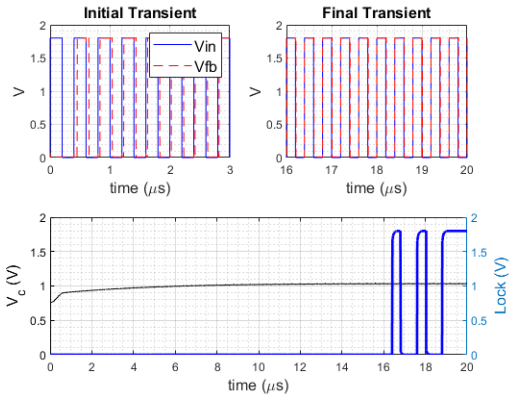
C. Further Post-Layout Simulations



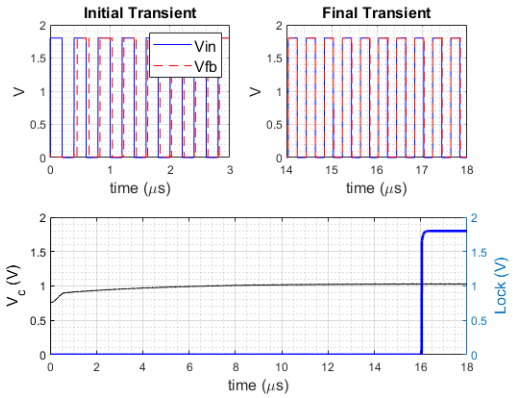
(a) 369 ns input delay



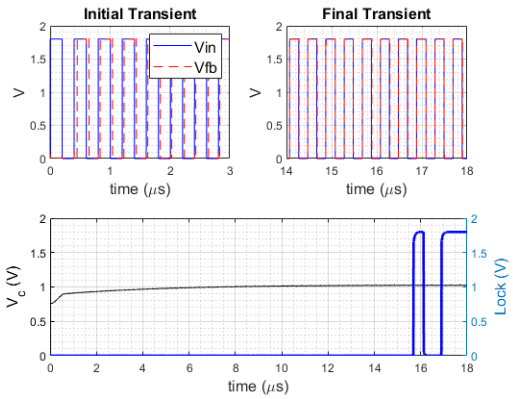
(b) 399 ns input delay



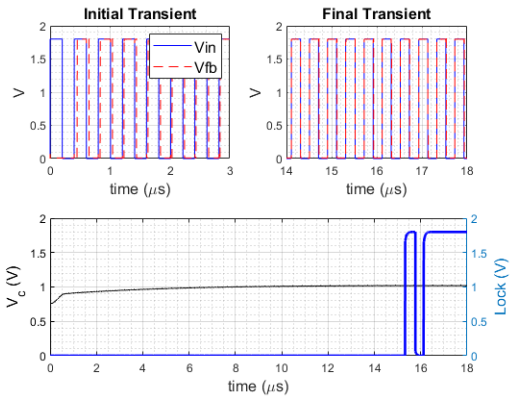
(c) 400 ns input delay



(d) 401 ns input delay



(e) 402 ns input delay



(f) 403 ns input delay

Figure C.1: A selection of post-layout results showing the initial and final transients, and the control voltage for the DLL at different input frequencies