DESIGN AND EVALUATION OF A TIME TO DIGITAL CONVERTER

A THESIS

Submitted by

GOPAL S. PATIL

for the award of the degree

of

MASTER OF TECHNOLOGY

DEPARTMENT OF ELECTRICAL ENGINEERING
INDIAN INSTITUTE OF TECHNOLOGY MADRAS
CHENNAI-600036
MAY 2019.

THESIS CERTIFICATE

This is to certify that the thesis entitled **"DESIGN AND EVALUATION OF A TIME TO DIGITAL CONVERTER"** submitted by **GOPAL S. PATIL** to the Indian Institute of Technology, Madras for the award of the degree of **MASTER OF TECHNOLOGY** is a bona fide record of research work carried out by him under my supervision. The contents of this thesis, in full or in parts, have not been submitted to any other Institute or University for the award of any degree or diploma.

Dr. Boby George
Department of Electrical Engineering
Indian Institute of Technology Madras
Chennai – 600 036.

Place: Chennai

Date: 5th May 2019

ACKNOWLEDGEMENTS

I would like to thank all the people who have given me constant support and encouragement all the time. This project would not have been possible without the kind support and help of many individuals and I would like to extend my sincere thanks to all of them.

I am highly indebted to Dr. Boby George for his guidance and constant supervision as well as for providing necessary information regarding the project and also for their support in completing the project. I would like to express my gratitude towards my parents for their kind co-operation and encouragement which help me in the completion of this project.

I would like to thank all the teaching and non-teaching staff who directly or indirectly helped me to complete this project, especially Mrs.Rekha mam and Mr.Anand sir for their quick help whenever there was a requirement of any lab instrument.

I am sincerely thankful to my Electrical Department IITM for providing me all the facilities to complete this project.

ABSTRACT

Key words: Time-to-Digital Converter (TDC); Synthesis, Simulation, Implementation, Field Programmable Gate Array (FPGA), Resolution.

Time-to-Digital Converters are used in many applications; whenever there is a need of very high time resolution, for example, to measure the time difference between two pulses. The proposed TDC uses a vernier delay line for the fine measurement and a counter for the coarse measurement. The target of this project was to get the resolution of 500ps and we have got 413 ps of resolution with highest deviation of 196ps. In this project delay chain is made up of latches and LUT's.

There are different types of delay chain that can be implemented in FPGA hardware, and in this project vernier delay line is used because of better resolution. In this thesis different types of delay chain are discussed with their working principle, advantages and disadvantages. The implementation is done in the edge artix-7 FPGA board with the help of Xilinx Vivado 2018.3 tool. The HDL Verilog code requires to implement this design is provided in the appendix. The constraint file is also provided in the appendix.

All the results are taken from the post-implemented timing simulation. The synthesized and implemented design of Counter, Vernier delay line and combined both counter and delay line is discussed in the thesis. Results of respected blocks are also discussed and shown in the simulation in vivado 2018.3 design tool.

TABLE OF CONTENTS

ACKNOWLEDGEMENTS i

ABSTRACT ii

LIST OF FIGURES v

LIST OF TABLES vi

ABBREVIATIONS vii

CHAPTER 11

INTRODUCTION 1

1.1. INTRODUCTION 1

1.2. OBJECTIVE OF THE WORK 1

1.3. SCOPE OF THE THESIS 1

CHAPTER 2 **3**

TIME TO DIGITAL CONVERTER (TDC) 3

2.1 INTRODUCTION 3

2.2 WORKING PRINCIPLE OF TDC 4

2.3 TYPES OF THE TDC 6

2.3.1 BUFFER DELAY LINE BASED TDC 7

2.3.2 VERNIER DELAY LINE BASED TDC 8

2.3.3 INVERTER BASED DELAY LINE BASED TDC 9

CHAPTER 3 **10**

DESIGN AND IMPLEMENTATION OF TDC 10

3.1 INTRODUCTION OF FPGA EDGE ARTIX 7 AND VIVADO 10

3.2 DESIGN OF IMPLEMENTED CIRCUIT 11

3.3 IMPLEMENTATION OF DESIGN 13

3.3.1 IMPLEMENTATION OF VERNIER DELAY LINE 13

3.3.2 IMPLEMENTATION OF COUNTER WITH CONTROL SIGNAL 14

3.3.3 IMPLEMENTATION OF TDC 15

CHAPTER 4 **20**

RESULTS AND CHALLENGES 20

CHAPTER 5 **24**

CONCLUSIONS 24

APPENDIX 25

REFERENCES 41

LIST OF FIGURES

2.1. Principle of a counter based TDC.

2.2. Operating principle of time-to-digital converter is delay versions of the start signal are on the rising edge of the stop signal.

2.3. Input-Output characteristics of a TDC with offset error.

2.4. Input–output characteristics of TDC illustrating DNL error.

2.5. Buffer Chain delay line.

2.6. Propagation of start signal in the buffer chain.

2.7. Vernier delay line.

2.8. Propagation of start and stop signal through vernier delay line.

2.9. Inverter based Delay line.

3.1. Picture of Edge Artix-7 FPGA (Xilinx XC7A35T IC FPGA).

3.2. Complete Block diagram of Time-to-Digital converter.

3.3. Synthesized design of Vernier delay line in vivado.

3.4 Implemented design of Vernier delay line with inter connets and elements.

3.5. Placement of LUT's and FF's in the FPGA.

3.6 Synthesized design of counter by vivado tool.

3.7. Synthesized design of TDC in vivado.

3.8 Implemented design of TDC in the Edge Artix-7 FPGA.

3.9. Buffer chain based test setup.

4.1. Output of counter with time interval 40ns.

4.2. Delay line Simulation result after post implementaton timing simulation.

4.3. Input Output curve of measured and actual values.

LIST OF TABLES

4.1.Output of the Delay line for a applied time Interval.

ABBREVIATIONS

TDC	Time-to-Digital Converter
Dff	D-Flip-Flop
FPGA	Field Programmable Gate Array
LUT	Look Up Table
RADAR	Radio Detection And Ranging
LiDAR	Light Detection And Ranging
ADC	Analog to Digital converter
DNL	Differential Non-Linearity
INL	Integral Non-Linearity
LSB	Least Significant Bit
MHz	Mega Hertz
ISE	Integrated System Environment
ns	nano-second
ps	pico-second
clk	clock
CML	Current Mode Logic

CHAPTER 1

INTRODUCTION

1.1 TIME TO DIGITAL CONVERTER

Time-to-Digital Converters (TDC) are used to convert time interval between two pulses into a digital form. Time-to-Digital Converters are widely used in many applications whenever there is a requirement of measurement of time interval between two pulses, like RADAR, LiDAR, and time of flight measurement etc. TDCs have been used for more than 20 years in the field of particle and high-energy physics, where very high resolution time measurement is required. In the analog TDCs the traditional approach to time-to-digital conversion is first to convert the time interval into a voltage. In a second step this voltage is digitized by a conventional analog-to-digital converter (ADC). But, the Analog issues that degrade the TDC performance in time measurement. All building blocks in Analog TDC, i.e. the pulse generator, the integrator, and the ADC have to meet the full linearity demands of the overall TDC.

The analog TDCs mainly consist of an ADC so have all the impairments of analog circuits in deep sub-micron technologies. The advantages of the time domain can be exploited only if there is no analog conversion step in the time-to-digital conversion. To overcome the problems of analog TDC now a day's fully digital TDCs are designed. Fully digital TDCs are available in the market, in the form of ASIC but another easy way to design the TDC is to implement the TDC in a FPGA. This project is dedicated to implementation of the TDC in the FPGA edge Artix7.

1.2 OBJECTIVE OF THE WORK

The objective of this project is to design and implement a TDC in the FPGA with the resolution of 500 ps with high precision and minimum deviation.

1.3. SCOPE OF THE THESIS

Thesis is mainly focus on the design and implementation of TDC in FPGA, so that any time interval measurement application industrial or non-industrial can be implemented in the FPGA. Thesis will explain the concepts regarding design and implementation of TDC in FPGA. The thesis is organized in five chapters, in which first chapter gives the idea of analog and digital TDC and what are different issues in analog TDC? How that are solved in digital TDC. Different TDC applications and what is the concept used in the implementation is also explained.

Second chapter explains the detail concept of TDC, working principle and the different designs of implementation of TDCs in FPGA. It also gives the idea of resolution affected by different design and how different designs have their own pros and cons.

Third chapter mainly focused on implementation of TDC in FPGA. It will explain how to implement all the sub blocks of project in FPGA step by step to get a final result. In this we can see the delay line implementation, buffer line implementation, combination of both and the full project implementation.

Fourth chapter dedicated to results of the project and the experimental set up which is FPGA.

In the fifth chapter conclusions made from this work is explained. After the fifth chapter Verilog code of the design is provided with the constraints file in appendix.

CHAPTER 2

TIME TO DIGITAL CONVERTER

2.1. INTRODUCTION

A time to digital converters measures the time difference between two pulses and returns the time difference as a digital output code. The minimum value of time difference which can be measured is known as the resolution of the TDC. There are several ways and platforms to implement the TDC, which affects the resolution and performance of the measurement. The elements available in the FPGA are already placed in such a way that it creates favorable environment to implement the TDC delay line in the FPGA. An Array of flip-flops and LUT's allow a designer to implement the TDC in FPGA with high resolution in the range of few pico-second.

TDC's are widely used in various devices like ADPLL, time of flight cameras, integrated circuits, high speed data transfer etc. A time to digital conversion can be done by counter but to get a higher resolution a higher clock frequency is require so higher the power consumption for the generation and processing of the clock signal is require. Above a certain clock frequency CMOS based oscillators are not available anymore, so even more expensive Current Mode Logic (CML) based or LC oscillators are required for the clock generation. Timing restrictions in the counter as well as in the controller circuit pose another limit on the maximum frequency. Delay line specially used to divide the clock cycle to increase the resolution so we can say that delay line implemented in FPGA is the heart of the TDC which ultimately responsible for the higher resolution.

The architecture of TDC can be implemented in analog or digital approaches. In analog domain time stretching and time-to-amplitude conversion methods are two classical analog TDC methods. Analog TDC's often suffer from large temperature drift and poor stability also the area-consuming devices in analog TDC hinder its implementation in integrated circuits. This motivates the designer to design a fully digital TDC.

To implement a digital TDC in FPGA we need to use LUTs and registers already available in the FPGA. In the present day's many designer are designing TDCs in FPGA with a resolution of few hundred and tens of picoseconds.

Time-to-Digital Converter draws many parallels with an ADC (Analog-to-Digital Converter) in terms of its characteristics. The basic difference is that the nature of the analog input is voltage domain for ADC's while that of TDC's is time domain. Besides that many of the terms used to describe the imperfections of an ADC such as INL (integral non-linearity) and DNL (differential non-linearity) are applicable to a TDC also.

2.2. WORKING PRINCIPLE OF TDC

TDC simply quantize the time interval with the help of counter by counting the cycles of a reference clock fitting into the respective measurement interval between start and stop pulse [1]. The measurement accuracy can be increased by a higher clock frequency. However, higher the clock frequency, higher the power consumption for generation and the processing of the clock signal. Timing restrictions in the counters as well as in the controller circuit pose another limit on the maximum frequency. To avoid a higher power consumption and at the same time to increase the time resolution a delay line can be design which subdivides the reference clock cycle.

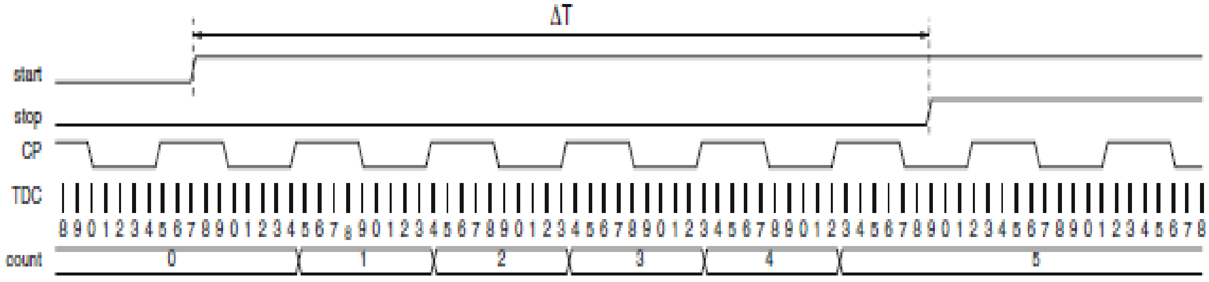


Fig.2.1. Principle of a counter based TDC.

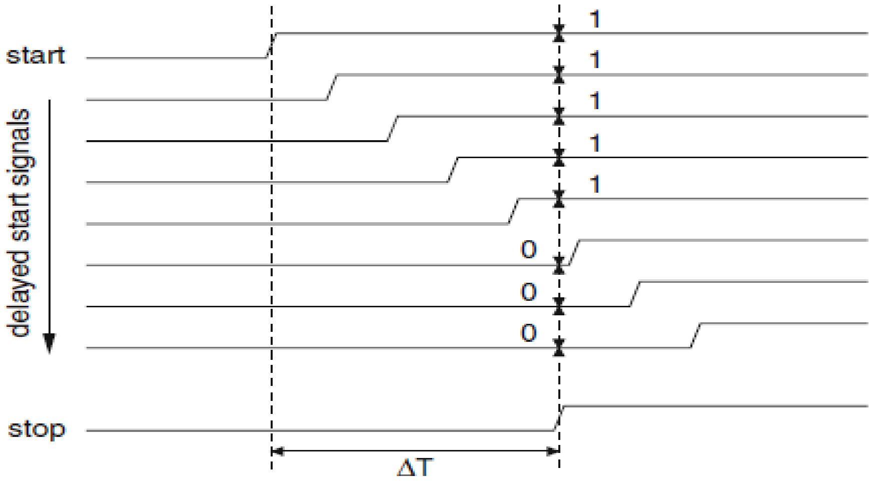


Fig.2.2.Operating principle of time-to-digital converter is Delayed versions of the start signal are sampled on the rising edge of the stop signal.

The interval that is less than reference clock cycle is measure with the help of delay line. Through Delay line start pulse propagate at each positive edge of reference clock cycle and it is delayed by a delay elements. The value of delay element is encodes when the stop pulse comes. The overall view of the principle of working can be seen in the figure 2.2.

The term time-to-digital converter refers to a data converter interface whose analog input is a timing event and output is a digital word corresponding to the magnitude (and sometimes polarity) of that timing event with some quantization error [2].

$$\Delta\ T = [Bout] \times T_{LSB} + \varepsilon + \text{(Some extra element propagation delay)} \qquad (2.1)$$

Where ε represents the quantization error associated with finite resolution of the conversion process (this will be further explained), ΔT describes the analog time event and Bout is the binary digital word output of the conversion process. Here some extra propagation delay is given for the AND gate which is used for gating, it is adding extra delay in experiment that needs to added in the final reading.

2.2.1 LINEAR AND NON-LINEAR NON-IDEALITIES OF TDC CHARACTERISTICS

The imperfections or non-idealities of the TDC characteristic can be classified as linear and non-linear. Offset is the linear imperfection while Integral Non-linearity (INL) and Differential Non-linearity (DNL) are non-linear imperfections [2]. Linearity problem can be easily correct but the Non-linear problems are very difficult to correct or mostly it can't be removed completely.

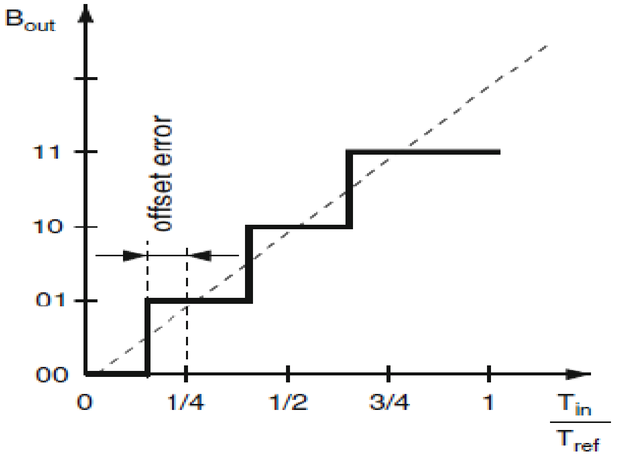


Fig: 2.3. Input-Output characteristics of a TDC with offset error.

The first transition for an ideal TDC occurs when the input is T_{LSB} i.e. $T_{00...01} = T_{LSB}$. The offset error is the deviation of the $T_{00...01}$ value from this ideal value, expressed in terms of T_{LSB}. This is best expressed in the following equation and illustrated in figure 2.3.

$$E_{offset} = (T_{00..001} - T_{LSB}) / T_{LSB} \qquad (2.2)$$

The non-linear imperfections cover all the deviations in the TDC characteristic that potentially lead to non-linear distortion in its output for a dynamic input signal. Differential Non-Linearity (DNL) is used to describe the deviation of each step from its ideal value of T_{LSB} normalized to T_{LSB}. INL (Integral Non-Linearity) describes the cumulative deviation of each step from the ideal value. An example of a TDC characteristic with DNL is shown in figure 2.4 [2].

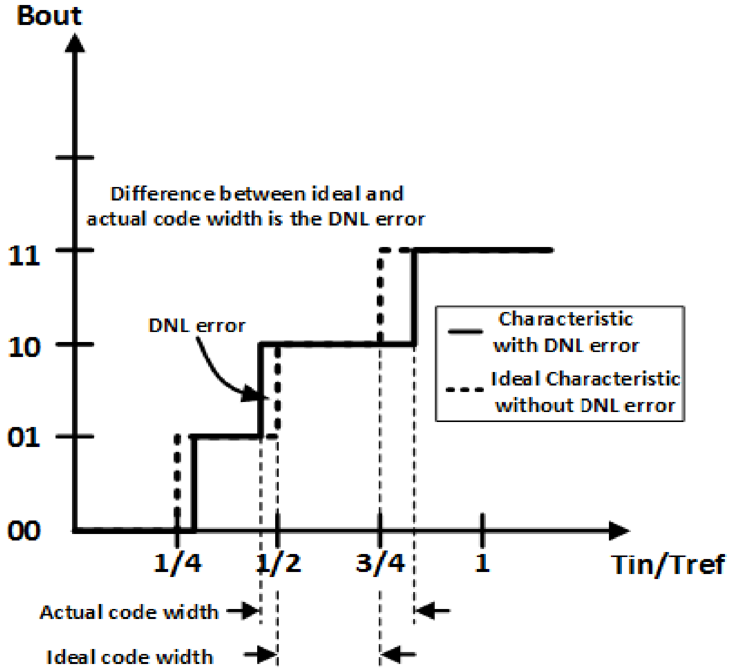


Fig .2.4. Input–output characteristics of TDC illustrating DNL error [1].

2.3. TYPES OF TDC

TDC can be design in various ways to measure the fine reading, but coarse measurement is performed with the help of counter in each design. At the end, coarse and fine reading can be add and subtract to get the final measurement value which is the time difference of start and stop pulse. TDC's are differentiated on the basis of the delay line used to measure the fine reading. Different delay lines used in the TDC are Buffer delay line, Vernier delay line and Inverter based delay line [1]. Time resolution is the important parameter in designing of TDC. Different delay lines are used to increase the resolution and each delay line has its own advantages and disadvantages which are explained below.

2.3.1. BUFFER DELAY LINE BASED TDC

The working principle of Buffer delay line is shown in figure below. As the start signal (F_{CKV}) travels through the delay line it delayed by the buffer elements after propagating through it and the output of each buffer is stored in the register when stop signal (F_{REF}) arrives and then it is encoded by encoder. The output we get is thermometer code which we have to encode into a binary to add or subtract with the coarse measurement of counter to get the final reading. The resolution of this TDC is the propagation delay (T_{DEL}) of the buffer.

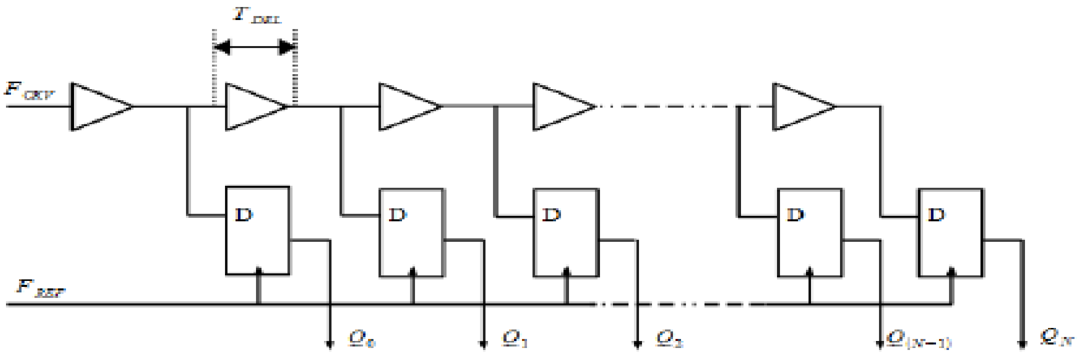


Fig. 2.5. Buffer Chain delay line.

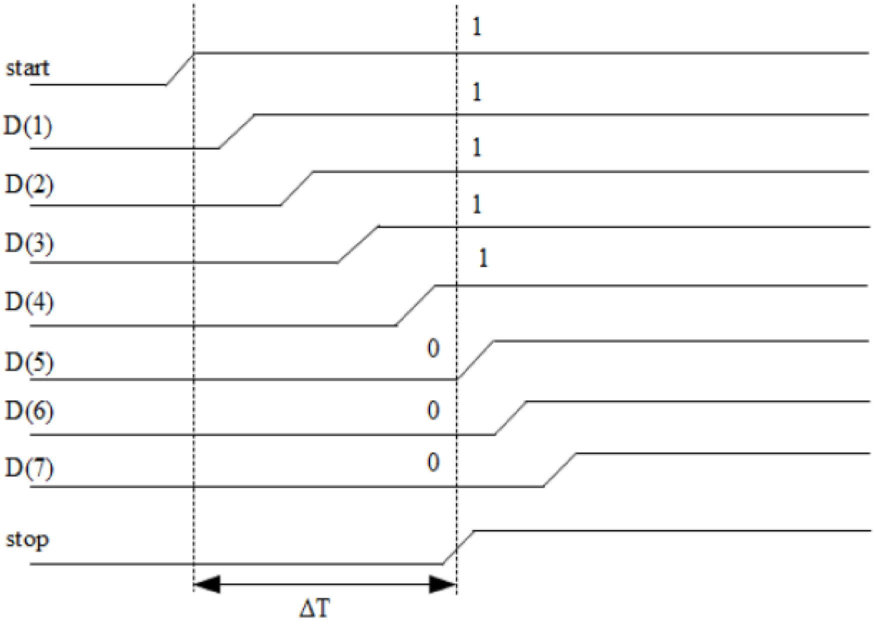


Fig.2.6. Propagation of start signal in the buffer chain [3].

2.3.2. VERNIER DELAY LINE BASED TDC

The vernier delay line is use to improve the resolution of the TDC. The resolution of this TDC is increase by delaying the stop signal. The elements with very small delay difference are used to form a delay lines. The upper delay line from which the starts signal propagates is design with a higher delay than the lower delay line from which the stop signal propagates. The resolution of this TDC is the delay difference ($T_{DEL1} - T_{DEL2}$) of two elements present in the two delay lines.

$$\text{Time Resolution} = (T_{DEL1} - T_{DEL2}) \qquad (2.2)$$

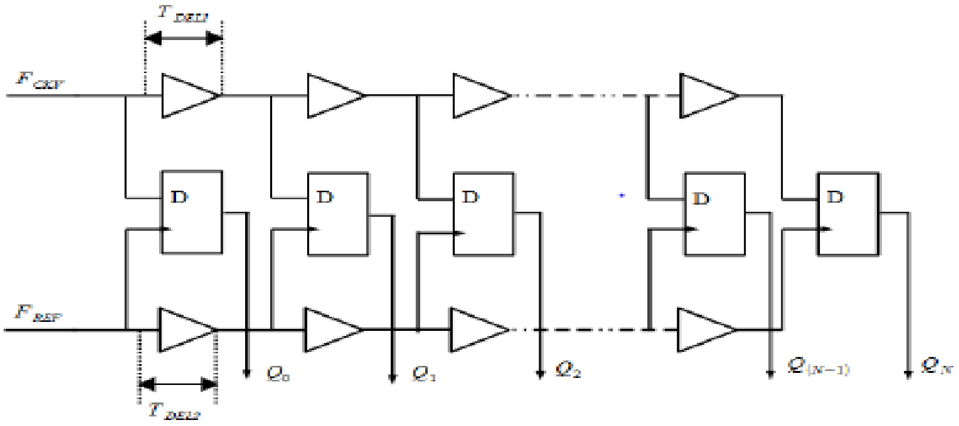


Fig 2.7. Vernier delay line.

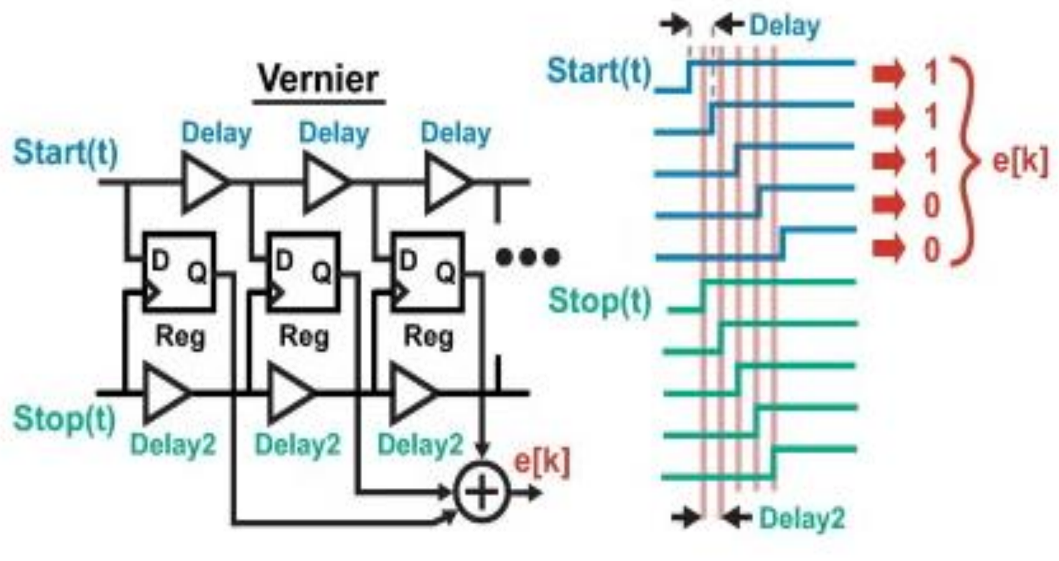


Fig. 2.8. Propagation of start and stop signal through vernier delay line [4].

2.3.3. INVERTER DELAY LINE BASED TDC

Inverter delay line based TDC is also specially designed to increase the resolution compare to Vernier delay line based TDC. As we know the delay of inverter is half of the delay of buffer so the delay line designed with inverter gives double the resolution than vernier type. But implementation of both Vernier delay line and Inverter delay line based TDC is complex.

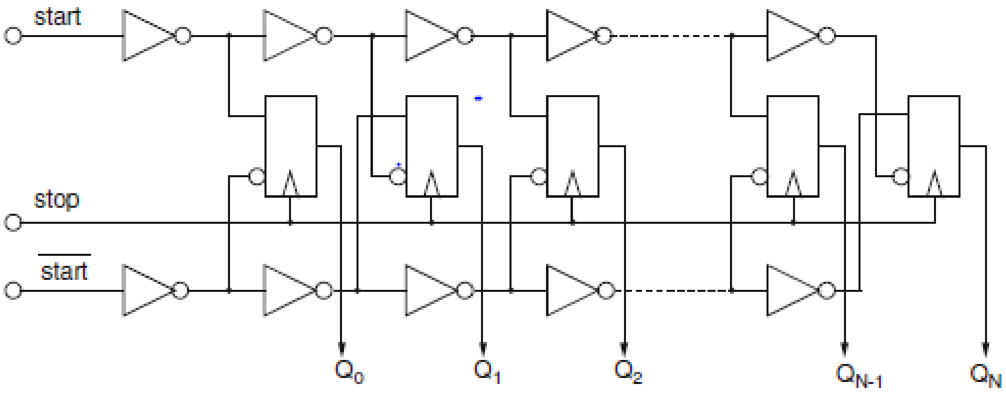


Fig. 2.9. Inverter based Delay line

Using of inverters means use of both the rising and the falling signal transitions are used for measurement [1]. So at the output instead of thermometer code a pseudo thermometer code with alternating ones and zeros will get. The output sample example is shown below:

111111111111111 => 000000000000000 buffer TDC
010101010101010 => 010101010101010 inverter TDC

The length of the interval is indicated not by a HIGH-LOW transition but by a phase change of the alternation HIGH-LOW sequence.

CHAPTER 3

DESIGN AND IMPLEMENTATION OF TDC

3.1. INTRODUCTION OF FPGA EDGE ARTIX 7 AND VIVADO 2018.3

Edge Artix-7 FPGA development board is an upgraded version of EDGE Spartan 6 board. As this board is version 7 we need to use Vivado tool, because ISE tool doesn't work above the version 6 of Spartan, Kintex, Artix etc. Vivado tool is specially design from the scratch for version-7 and above. Edge Artix-7 is a Xilinx XC7A35T IC FPGA which is of price RS 12500. It has many applications like Wireless control, Environment monitor, IoT (Internet of Things), Product Prototyping, Image Processing, Video Processing, and Audio Processing. Its operating frequency is 50 MHz. This FPGA is design using 28nm technology and it has six variable LUT's.

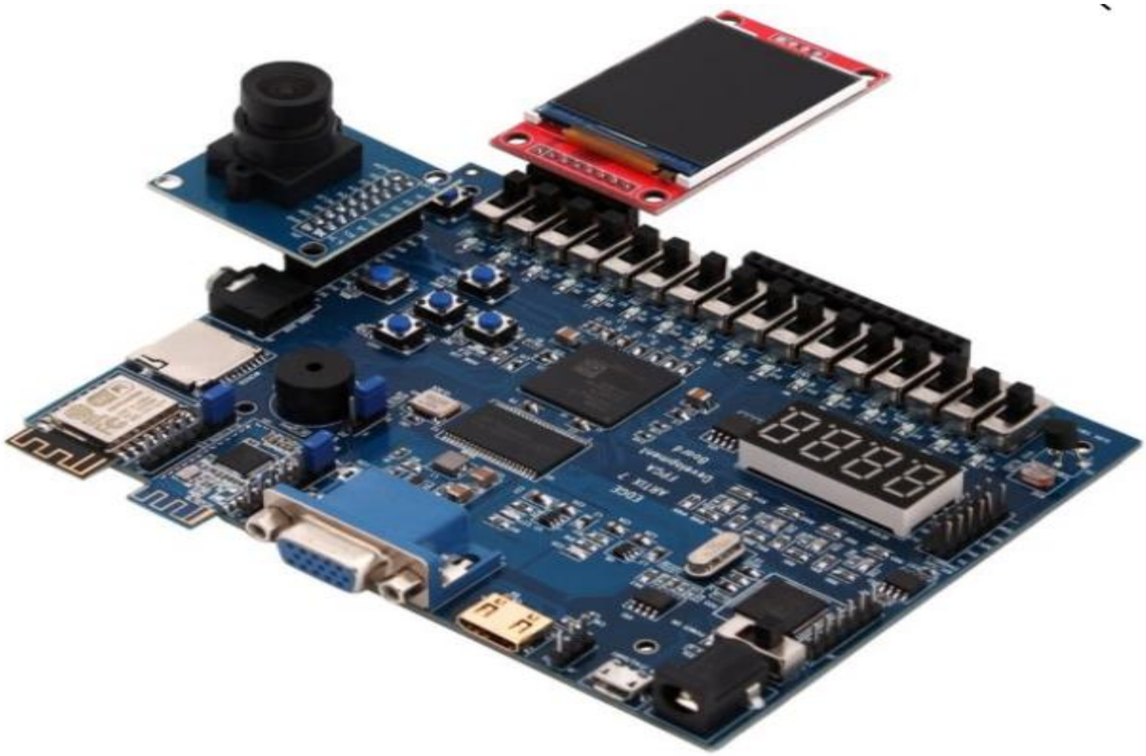


Fig 3.1. Picture of Edge Artix-7 FPGA (Xilinx XC7A35T IC FPGA).

Vivado is a software suite, it is design by Xilinx for synthesis and analysis of HDL designs [5]. Vivado is a well design synthesis tool which is well conceived, tightly integrated, blazing fast, scalable, maintainable, and intuitive. Vivaado software can download freely with vivado HL webPACK option. To learn vivado tool Xilinx provide lot of material and it is easy to learn. Vivado allows developer to synthesize the design and perform simulation with five different options like Run behavioral simulation, Run post-synthesis functional simulation, Run post-synthesis timing simulation, Run post-implementation functional simulation, and Run post-implementation timing simulation.

The post synthesis functional simulation allow designer to checks about functionality of the design after synthesis whether it is functionally working perfect or not and Run post-synthesis timing simulation gives the information about timing constraints. The post implementation simulation is the procedure can be perform after the implementation of design in FPGA which can be selected from the list available in the vivado software or if it is not available in the list then by entering the specification of FPGA designer can select the FPGA board for analysis of functionality of any design.

3.2. DESIGN OF IMPLEMENTED CIRCUIT

The complete design of circuit is shown in the diagram below. The coarse measurement is done by the counter and fine measurement is done by the vernier delay line [6]. The output of the counter and the delay line is added with the quantization error. The output of delay line is thermal code which is encoded with the help of encoder.

Working Principle of the design: When a START pulse come counter starts counting the number of reference cycles whose period is known to us which is 5ns in this project. That is output of clocking wizard. Clocking wizard multiplies the frequency of FPGA board from 50 MHz to 200 MHz to drive the counter and the D flip-flop (Dff) which sends the continuous value logic one to the delay line at each positive edge of the reference frequency. At the input of Dff, AND gate is provided which provides frequency to the counter and Dff only when the START pulse comes it is called gating of frequency. The delay of the AND gate we need to add in the final reading as it delays the frequency after the START pulse comes.

When the STOP pulse comes it stops the counter and the STOP pulse propagates through the buffer line which is provided as a clock to the Dff chain. At the start of experiment the STOP pulse is zero, so the upper latch chain act as a buffer chain, but when the STOP pulse comes it stores the last inputted value in the latch and send it to the Dff chain. It gives the thermal code to the encoder. This encoded value is the fine measurement.

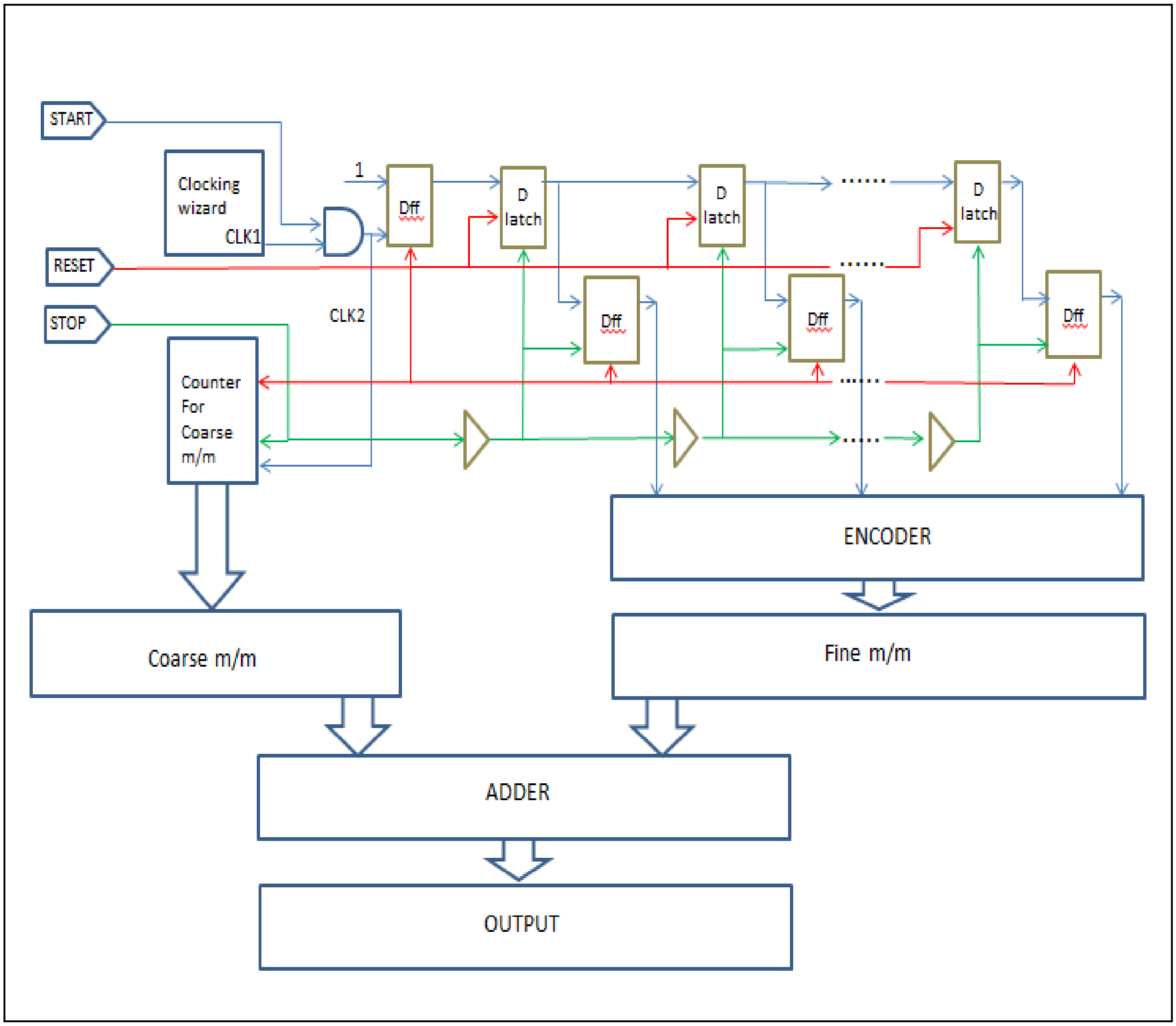


Fig: 3.2. Complete Block diagram of Time-to-Digital converter [6].

3.3 IMPLEMENTATION OF DESIGN

3.3.1. IMPLEMENTATION OF VERNIER DELAY LINE

Delay chain divides the time interval into sub interval that is equal to the propagation delay of the buffer. In this project D-latch is used as a buffer to delay the logic one signal. To work D-latch as a buffer logic zero is supplied to its clock input. After the STOP pulse comes the output values of latch is stored in the D flip-flop's (Dff). A chain of Dff is used to store the output of latch chain. The value in Dff chain is a thermal code which is encoded using ENCODER. The STOP signal is delay by using LUT chain as a buffer chain to get higher resolution. The synthesized and implemented design in the edge Artix-7 can be seen in the picture of vivado tool.

WORKING: The time interval higher than the reference clock cycle (5ns) is measured by the counter but the delay less than the reference clock cycle is measured by the delay line. To delay the start signal and subdivide into small interval, a logic one and not original START signal sends at every positive edge of clock cycle to the delay line. The logic one is sent at every clock cycle because when the STOP signal will come we don't know and we need to measure the time interval in between the STOP signal and a previous positive edge of reference clock cycle.

To ensure that the time interval will be in between STOP signal and the previous positive clock edge, logic one is propagated through delay chain with the help of Dff. STOP signal is passes through a LUT's which work as a buffer to delay the STOP signal for higher resolution. The program is written in "VERILOG HDL", synthesized and implemented in edge atrtix-7. Synthesized and implemented design can be seen in figure 3.3 and 3.4 respectively. Post timing simulation gives the delay of "Elements and Interconnects ". If the post-implemented timing simulation gives the desired delay then we can say that it will work in hardware also.

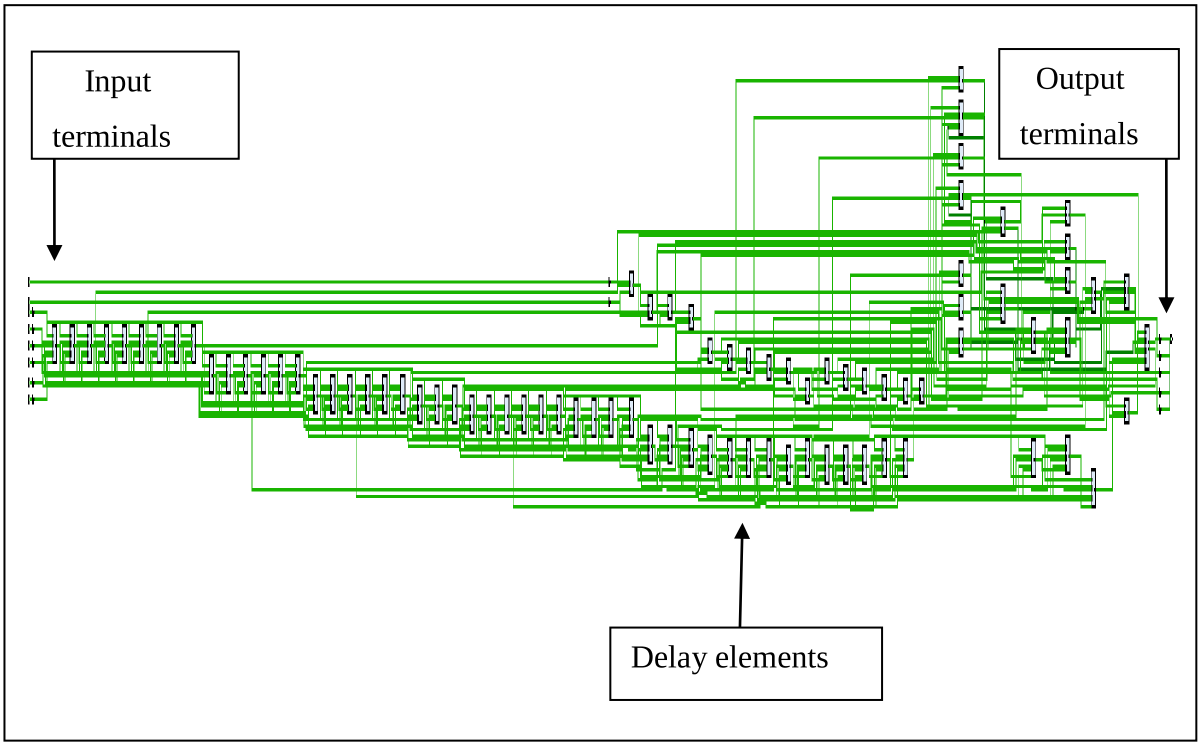


Fig. 3.3. Synthesized design of Vernier delay line in vivado.

The verilog code of the above design is available in the Appendix. Implemented dessign of Vernier delay line in the hardware is shown in the figure 3.4. The implementation has used 64 LUT’s and and 17 flip-flop’s (FF). The interconnects optimezed by the tool can be seen in the fig. 3.4 and the the placement of LUT’s and FF’s are shown in the figure 3.5. The outputs of the design are assign to the LEDs which are avalilable in the FPGA.

The details of harware LEDs which are assign to the variables in verilog code is given in the constraints file available in appendix. Constraints file is the file where we assigns all the variables to the desired harware input-output terminals. All the input and output terminals and their respective variables in the code can be seen in the constraints file.

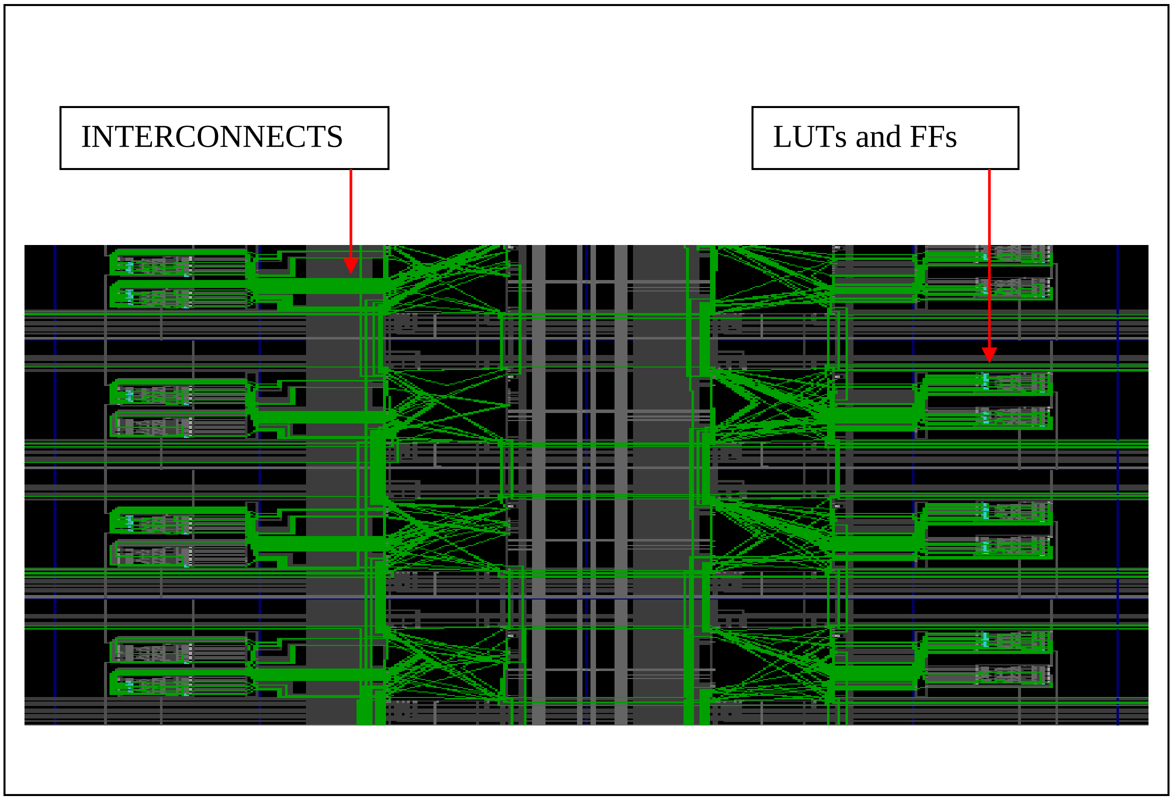


Fig. 3.4 Implemented design of Vernier delay line with interconnets and elements.

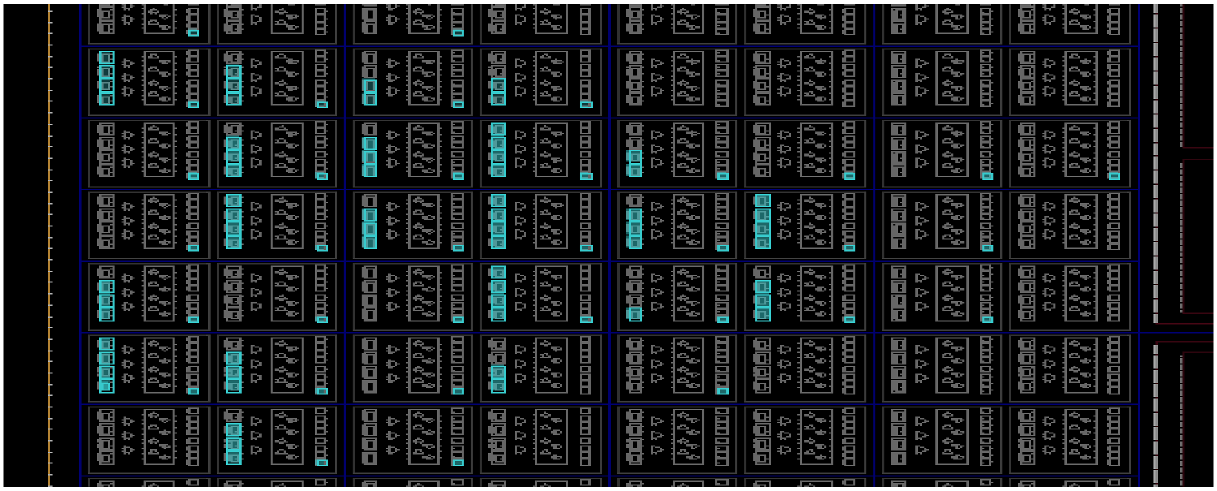


Fig. 3.5. Placement of LUT's and FF's in the FPGA.

After the implementation of design a Run post-implemetation is done and the resolution we got is 413ps which the difference between latch (T_{DEL1}) and the Buffer (T_{DEL2}) i.e. [$T_{DEL1} – T_{DEL2}$].

3.3.2 IMPLEMENTATION OF COUNTER WITH CONTROL SIGNAL

Counter is used for coarse measurement that is the time interval greater than the reference clock cycle. Counter should start counting when a START pulse arrives at start input and it should stop when STOP pulse arrives at stop input terminal. The START and STOP is a control signal for the counter, as it is controlling the counter. The synthesized design of the counter is shown in the figure 3.6. This counter is four bit counter so it can measure up to 80 ns. The range of counter can be increased by increasing the number of steps in the counter. A START, STOP, CLK, RESET, OUTPUT and the LUTs synthesized design is shown below in fig. 3.6.

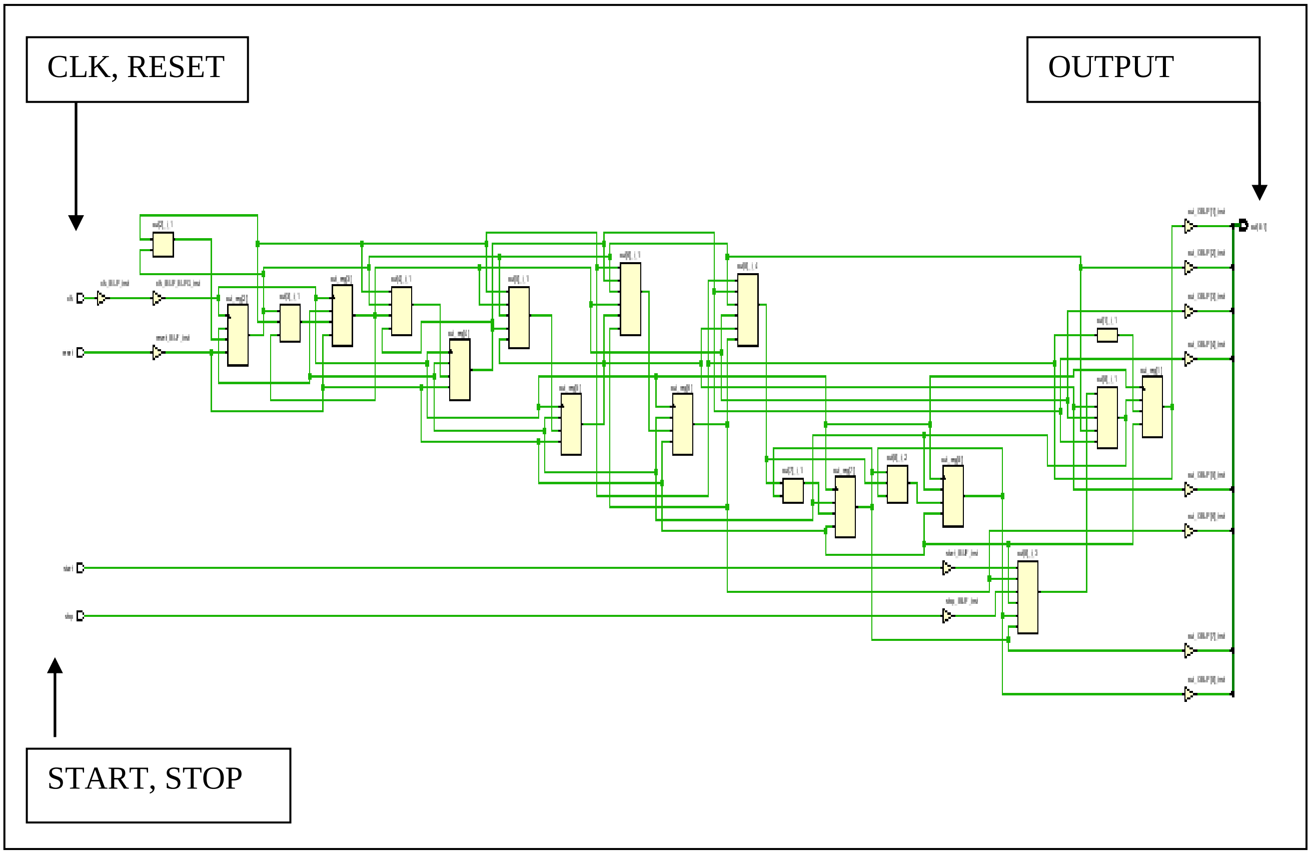


Fig. 3.6 synthesized design of counter by vivado tool.

3.3.3 IMPLEMENTATION OF TDC

The full TDC design contains Counter, Vernier delay line, and Encoder. Two separate outputs fine and coarse measurement has done. In the final reading we need to add the delay of Dff and AND gate which is used for gating of reference clock. The TDC is use 72 LUTs and 22 FFs to implement the design.

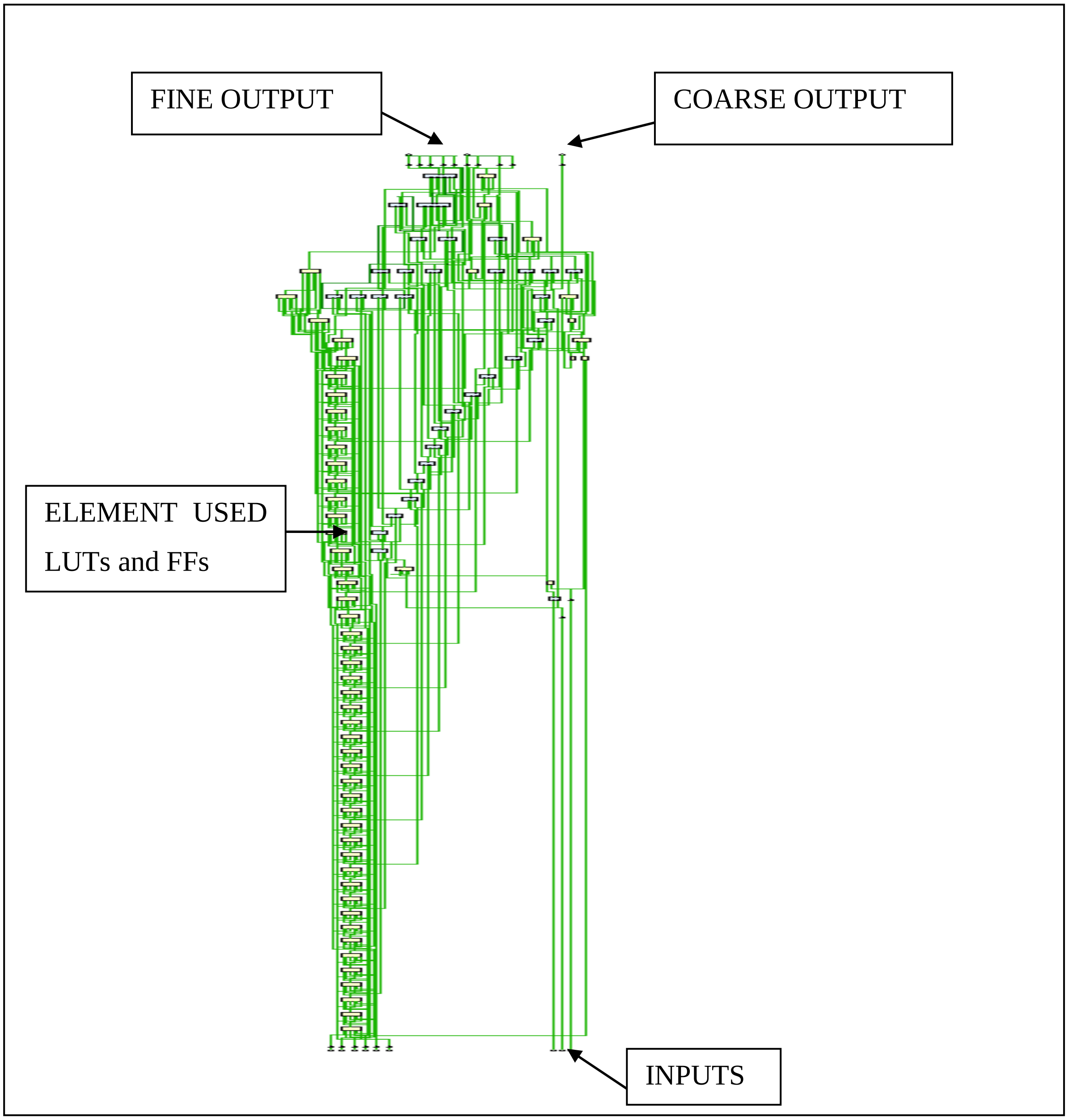


Fig.3.7. Synthesized design of TDC in vivado.

The synthesized design of TDC shown in figure 3.7. The implemented design of TDC can be seen in the figure 3.8. The green part shows the used place for the design in the FPGA. Tool optimize the design and place and route accordingly. The place and route occurs on the basis of many things in that one of the important point is which input and output pin from the design is assigned to which variables. According to pins provided on FPGA board and the same pins applied by the design place and route happens in the optimization. The counter used is of four bit so it can measure maximum 80 ns of time difference between START and STOP signal. The range of this TDC is 500ps to 80ns.

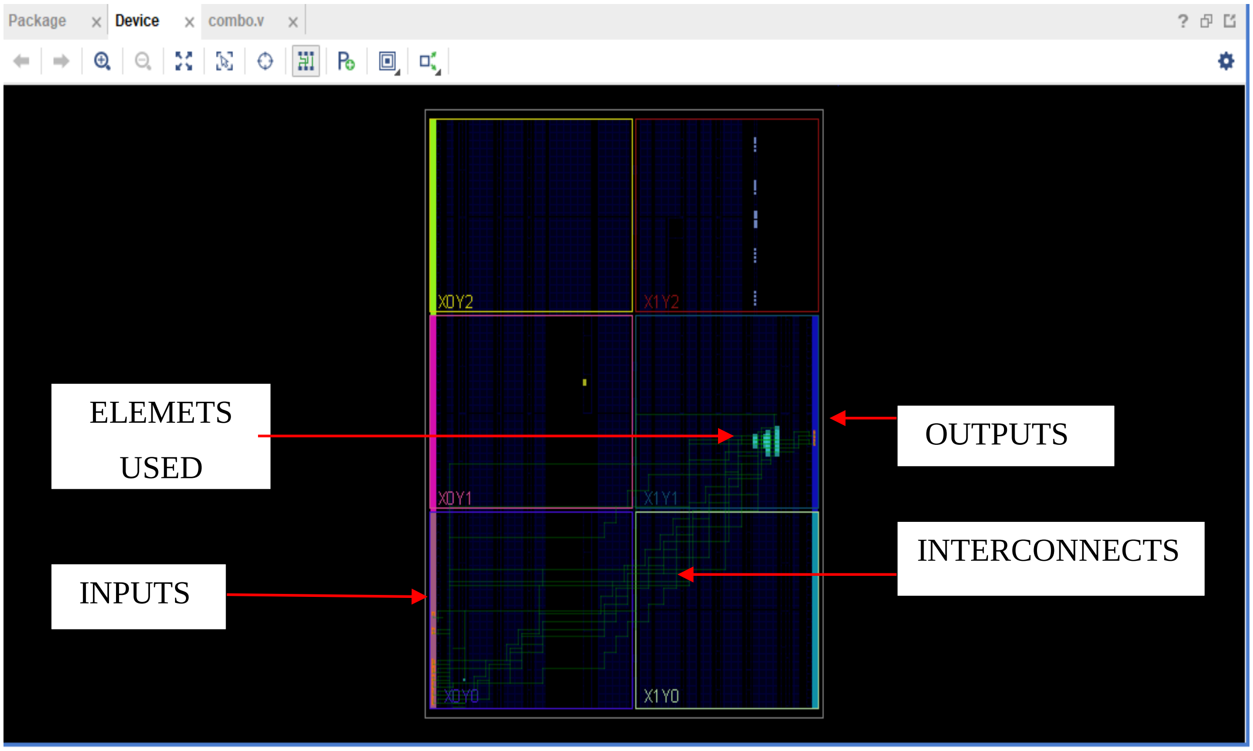


Fig. 3.8 Implemented design of TDC in the Edge Artix-7 FPGA.

TEST SETUP

TDC requires a test setup which is perfectly controllable. A START and STOP signal generated need to be control to get the perfect time testing. For Signal generation we can use another FPGA in which designer can write a HDL code to form a buffer chain of known delay. This setup allows to measure actual delay of a buffer to be measured. The full test setup is shown in the figure 3.9 [7]. Architecture can be used for the buffer chain based characterization is shown in figure 3.9. A single input signal can applied as the input to both start and buffer chain inputs.

The buffer chain is a series of cascaded buffers, with each buffer output introducing a delay of the order of a few hundreds of pico seconds. The TDC is characterized by using the output of the buffer chain as the stop signal. By varying the values of the multiplexer select lines, the delay between start and stop is increased and corresponding readings were observed and plotted. The disadvantage in this method is that the accuracy of the test results depends on the buffer delay as well as the routing delay between buffers. Hence this can yield only an approximate characterization of the TDC.

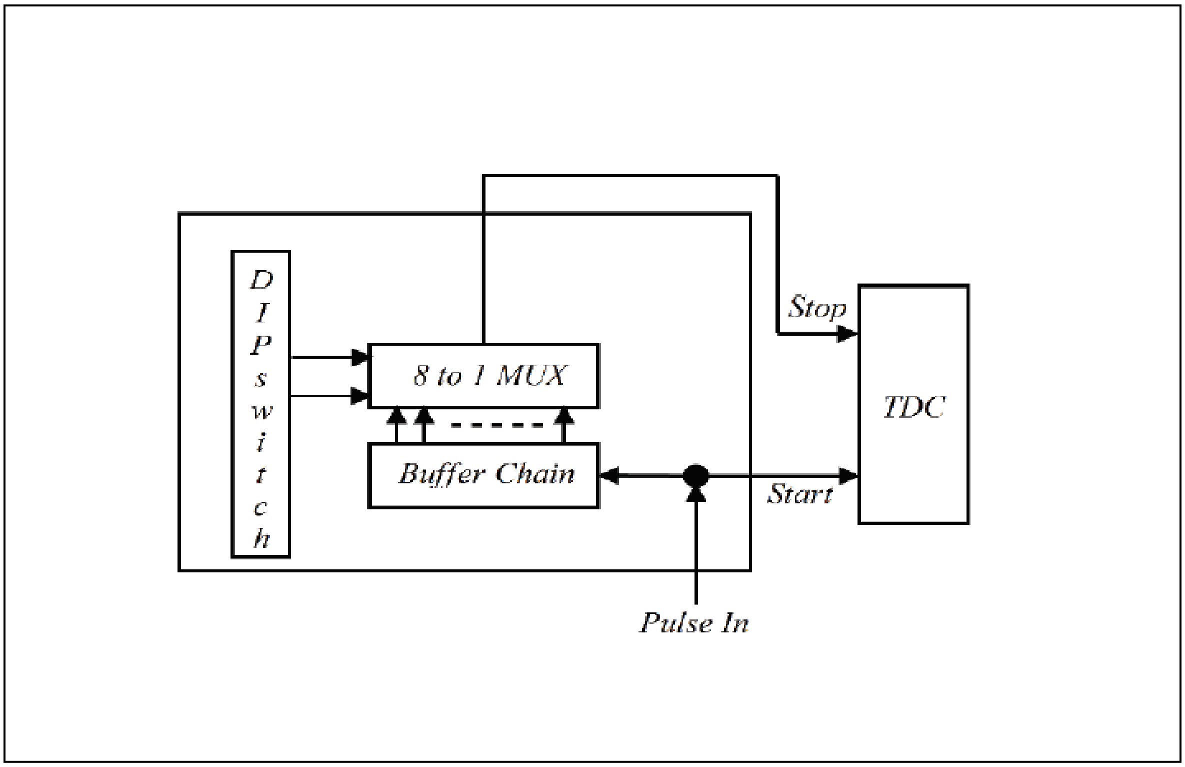


Fig.3.9 Buffer chain based test setup

CHAPTER 4

RESULTS AND CHALLENGES

RESULTS

COARSE TIMING MEASUREMENT

The coarse measurement by a counter in the post-implemented timing simulation is shown in the figure 4.1. The time difference between the START and STOP pulse applied to the design was 40ns. The output shown by the counter was 8 shown in the figure 4.1, which is correct as reference clock cycle applied was of 5ns. Output indicates that counter has taken 8 steps before the STOP pulse arrives. One step is of 5ns and 8 steps means 40ns time difference. In the same way different time interval is applied and the output of counter is observed.

If there is a requirement of time interval measurement in the resolution range of 3 to 5 ns, single counter can be used as a TDC to measure the time interval, because it can achieve this range of resolution easily. If the resolution requirement is in pico-second then delay line need to be designed. The clocking wizard which is available in tool IP Catalog is used to multiply the frequency from 50MHz to 200MHz.

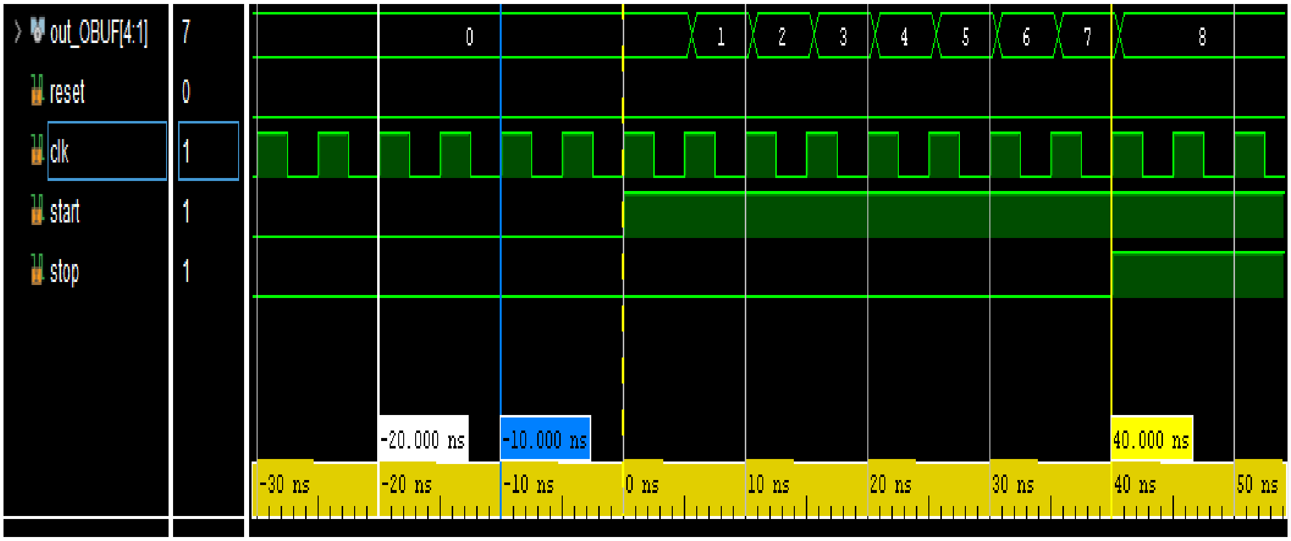


Fig.4.1. Output of counter with time interval 40ns.

FINE TIMING MEASUREMENT

Fine measurement is experimented for the range of 500ps to 5000ps with difference of 500ps. The resolution observed is 413ps which is the time difference of latch and buffer. The output for respective input is shown in the table 4.1. The output shown by the vivado tool after post-implementation timing simulation is shown in the figure 4.2.

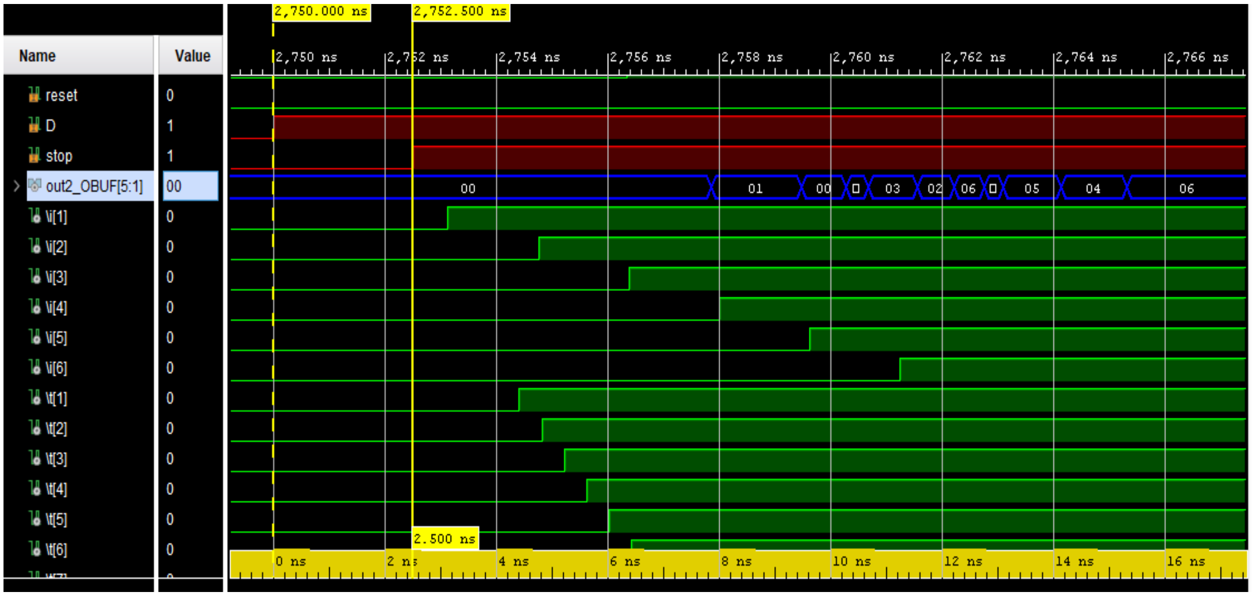


Fig.4.2.Delay line Simulation result after post implementaton timing simulation.

Time difference applied (pico-second)	Output of Encoder	Measured value (pico-second)	Deviatiaton (pico-second)
500	1	413	87
1000	2	816	184
1500	4	1652	-152
2000	5	2065	-65
2500	6	2478	22
3000	7	2891	109
3500	8	3304	196
4000	10	4130	-130
4500	11	4543	-43
5000	12	4956	44

Table 4.1.Output of the Delay line for a applied time Interval.

The Highest deviation in the measurement is 196 ps canbe seen in the table 4.1. The linearity of designed vernier line is shown in the figure 4.2.

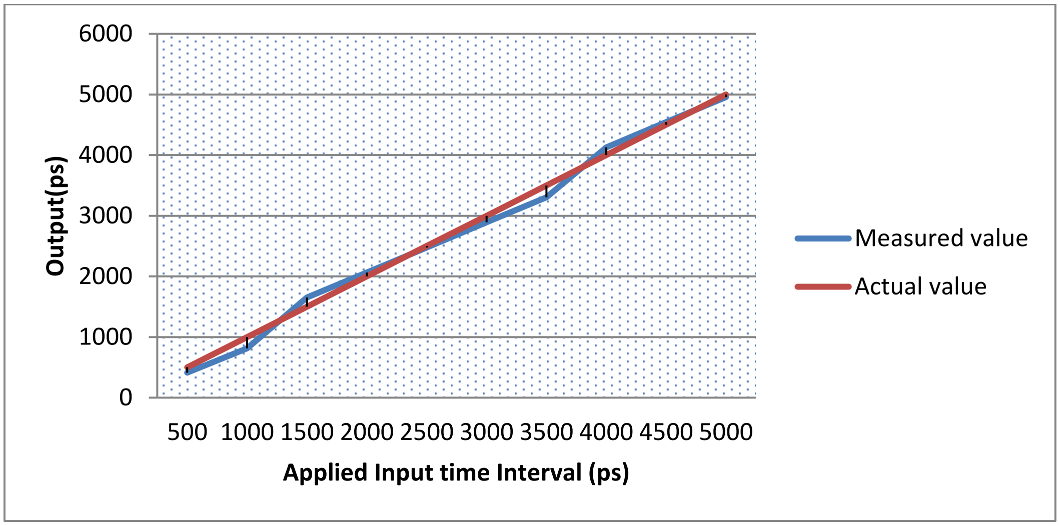
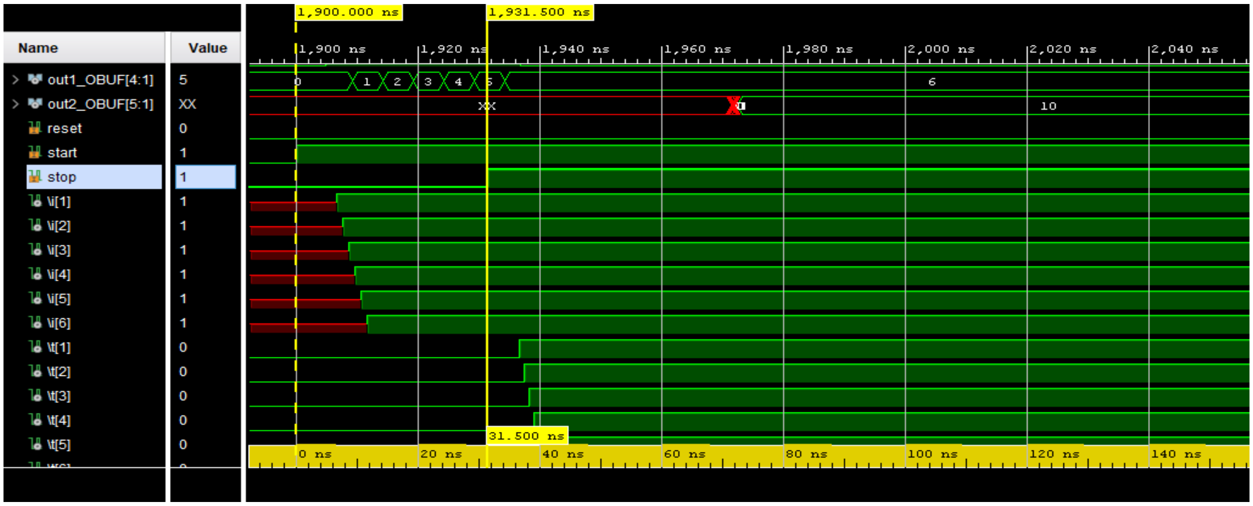


Fig.4.3. Input Output curve of measured and actual values.

COARSE AND FINE MEASUREMENT

Finally all modules instantiated together for coarse and fine measurement together. The Output result when a time difference between START and STOP signal is used is shown in the figure 4.3. the time difference applied is 31.5ns the output of the TDC is shown in figure4.3.



Name	Value
out1_OBUF[4:1]	5
out2_OBUF[5:1]	XX
reset	0
start	1
stop	1
\i[1]	1
\i[2]	1
\i[3]	1
\i[4]	1
\i[5]	1
\i[6]	1
\t[1]	0
\t[2]	0
\t[3]	0
\t[4]	0
\t[5]	0

Fig.4.4. Coarse and fine measurement of time interval

CHALLENGES

The name TDC itself gives an idea about challenges in the project of FPGA. Project on FPGA which deals with timing constraints have many challenges which are discussed here. This project totally focussed on timing constraints which are mainly depeds on the elements which are readily available in the FPGA board. FPGA forces the designer to work according to bord as there is no control of designer on the delay of elements.

There are different hardware challenges as the number of elements changes the resolution of design changes. The small change in the load of delay chain can be result in huge change in the resolution. The inputs and outputs assigns to the variables defined in the code can also change the resolution, as it has a resistance and capacitance. Solving all the issue for better resolution is a big challlenge.

CHAPTER 5

CONCLUSIONS

The TDC implemented in this project can be used for the application where the resolution of 500ps and above is requires. Resolution of implemented TDC is 413ps and its range of measurement is 80ns with maximum deviation 196ps. The range of measurement can be easily increased by increasing the number of steps of the counter. If the resolution requirement is in 3 to 5ns then only counter is enough to measure the time interval between START and STOP signal. If the measurement range is very less, then, only vernier delay line can be used to measure the interval. Combine module of counter and delay line gives the higher range and higher resolution. The resolution of TDC can increase by trying the implementation in different way with different combination of elements with LUT and Registers available in the FPGA board.

APPENDIX

VERILOG PROGRAM

All the text starting with the two forward slash (//) indicates the comment to understand the code by the reader easily.

1.COUNTER

```
module counter (clk,reset,start,stop,out);
         input     clk, reset,start,stop;  //input decalration
        output [4:1] out ;             //output declaration
        reg [4:1] out1;
    assign out = out1;               // output reg  can't be used as out =out, because output can't
                                      //assign to output so "reg out1"  is defined and then it assigned.
    always @(posedge clk)
       begin
           if (reset)           //resets the counter to zero
                  out1 = 0;
          else if(start & ~stop) //if start and not of stop is
                  out1= out1 + 1'b1;   // then counter is incremented by one.
          Else if(stop)
                  Out1 = out1;          // output is displayed if above conditions fail
       end
endmodule
```

CONSTRAINTS FILE

```
set_property PACKAGE_PIN K2 [get_ports {out[4]}]
set_property PACKAGE_PIN K3 [get_ports {out[3]}]
set_property PACKAGE_PIN L2 [get_ports {out[2]}]
set_property PACKAGE_PIN L3 [get_ports {out[1]}]
set_property IOSTANDARD LVCMOS33 [get_ports {out[4]}]
set_property IOSTANDARD LVCMOS33 [get_ports {out[3]}]
set_property IOSTANDARD LVCMOS33 [get_ports {out[2]}]
set_property IOSTANDARD LVCMOS33 [get_ports {out[1]}]
set_property IOSTANDARD LVCMOS33 [get_ports clk]
set_property IOSTANDARD LVCMOS33 [get_ports reset]
set_property IOSTANDARD LVCMOS33 [get_ports start]
set_property IOSTANDARD LVCMOS33 [get_ports stop]
set_property PACKAGE_PIN N11 [get_ports clk]
set_property PACKAGE_PIN L14 [get_ports reset]
set_property PACKAGE_PIN R12 [get_ports start]
set_property PACKAGE_PIN T12 [get_ports stop]
```

2.VERNIER DELAY LINE

```
module Dchain (D,stop ,reset,a,b,c,d,e,out2);
    input stop,reset,a,b,c,d,e,D;  //input decalration
    output reg [5:1] out2 ;          //output declaration
    (*keep=1*) wire t [51:1];     //wire inter connect to observe output at each delay element
     (*keep=1*) wire i [17:1];
 wire  q [17:1];
 begin
            buff g1(stop,a,b,c,d,e,t[1]);//1  // lower delay chain element
            buff g19(t[1],a,b,c,d,e,t[2]);      // buffer element is design by connecting 3 LUT's
            buff x28(t[2],a,b,c,d,e,t[3]);     //To increase the resolution (1 buffer = 3 LUT's)
            latch g2(D,t[3],reset,i[1]);         //Upper  delay chain element
            Dff f1(t[3],reset,i[1],q[1]);         // D flip-flop to store the value of output of upper delay chain


            buff g20(t[3],a,b,c,d,e,t[4]);//2  // total 17 elements used to cover 5ns time interval
            buff x29(t[4],a,b,c,d,e,t[5]);
            buff g3(t[5],a,b,c,d,e,t[6]);
            latch g4(i[1],t[6],reset,i[2]);
```

```
Dff f2(t[6],reset,i[2],q[2]);

buff g5(t[6],a,b,c,d,e,t[7]);//3
buff g21(t[7],a,b,c,d,e,t[8]);
buff x30(t[8],a,b,c,d,e,t[9]);
latch g6(i[2],t[9],reset,i[3]);
Dff f3(t[9],reset,i[3],q[3]);

buff g7(t[9],a,b,c,d,e,t[10]);//4
buff g22(t[10],a,b,c,d,e,t[11]);
buff x31(t[11],a,b,c,d,e,t[12]);
latch g8(i[3],t[12],reset,i[4]);
Dff f4(t[12],reset,i[4],q[4]);

buff g9(t[12],a,b,c,d,e,t[13]);//5
buff g23(t[13],a,b,c,d,e,t[14]);
buff x32(t[14],a,b,c,d,e,t[15]);
latch g10(i[4],t[15],reset,i[5]);
Dff f5(t[15],reset,i[5],q[5]);

buff g11(t[15],a,b,c,d,e,t[16]);//6
buff g24(t[16],a,b,c,d,e,t[17]);
buff x33(t[17],a,b,c,d,e,t[18]);
latch g12(i[5],t[18],reset,i[6]);
Dff f6(t[18],reset,i[6],q[6]);

buff g13(t[18],a,b,c,d,e,t[19]);//7
buff g25(t[19],a,b,c,d,e,t[20]);
buff x34(t[20],a,b,c,d,e,t[21]);
latch g14(i[6],t[21],reset,i[7]);
```

```
Dff f7(t[21],reset,i[7],q[7]);

buff g15(t[21],a,b,c,d,e,t[22]);//8
buff g26(t[22],a,b,c,d,e,t[23]);
buff x35(t[23],a,b,c,d,e,t[24]);
latch g16(i[7],t[24],reset,i[8]);
Dff f8(t[24],reset,i[8],q[8]);

buff g17(t[24],a,b,c,d,e,t[25]);//9
buff g27(t[25],a,b,c,d,e,t[26]);
buff x36(t[26],a,b,c,d,e,t[27]);
latch g18(i[8],t[27],reset,i[9]);
Dff f9(t[27],reset,i[9],q[9]);

buff g28(t[27],a,b,c,d,e,t[28]);//10
buff g29(t[28],a,b,c,d,e,t[29]);
buff x37(t[29],a,b,c,d,e,t[30]);
latch g30(i[9],t[30],reset,i[10]);
Dff f10(t[30],reset,i[10],q[10]);

buff g31(t[30],a,b,c,d,e,t[31]);//11
buff g32(t[31],a,b,c,d,e,t[32]);
buff x38(t[32],a,b,c,d,e,t[33]);
latch g33(i[10],t[33],reset,i[11]);
Dff f11(t[33],reset,i[11],q[11]);

buff g34(t[33],a,b,c,d,e,t[34]);//12
buff g35(t[34],a,b,c,d,e,t[35]);
buff x39(t[35],a,b,c,d,e,t[36]);
latch g36(i[11],t[36],reset,i[12]);
```

```
Dff f12(t[36],reset,i[12],q[12]);

buff g37(t[36],a,b,c,d,e,t[37]);//13
buff g38(t[37],a,b,c,d,e,t[38]);
buff x40(t[38],a,b,c,d,e,t[39]);
latch g39(i[12],t[39],reset,i[13]);
Dff f13(t[39],reset,i[13],q[13]);

buff g40(t[39],a,b,c,d,e,t[40]);//14
buff g41(t[40],a,b,c,d,e,t[41]);
buff x41(t[41],a,b,c,d,e,t[42]);
latch g42(i[13],t[42],reset,i[14]);
Dff f14(t[42],reset,i[14],q[14]);

buff g43(t[42],a,b,c,d,e,t[43]);//15
buff g44(t[43],a,b,c,d,e,t[44]);
buff x42(t[44],a,b,c,d,e,t[45]);
latch g45(i[14],t[45],reset,i[15]);
Dff f15(t[45],reset,i[15],q[15]);
buff g46(t[45],a,b,c,d,e,t[46]);//16
buff g47(t[46],a,b,c,d,e,t[47]);
buff x48(t[47],a,b,c,d,e,t[48]);
latch g49(i[15],t[48],reset,i[16]);
Dff f16(t[48],reset,i[16],q[16]);

buff g50(t[48],a,b,c,d,e,t[49]);//17
buff g51(t[49],a,b,c,d,e,t[50]);
buff x52(t[50],a,b,c,d,e,t[51]);
latch g53(i[16],t[51],reset,i[17]);
Dff f17(t[51],reset,i[17],q[17]);
```

```
end
 always @ (stop or reset or q)              //Encoder to encode the thermal code
     begin
         if ( reset)
                     out2 = 5'b00000;
         else if (   q[17] ) out2 = 5'b10001;
       else if (   q[16] ) out2 = 5'b10000;
       else if (   q[15] ) out2 = 5'b01111;
       else if (   q[14] ) out2 = 5'b01110;
       else if (   q[13] ) out2 = 5'b01101;
       else if (   q[12] ) out2 = 5'b01100;
       else if (   q[11] ) out2 = 5'b01011;
       else if (   q[10] ) out2 = 5'b01010;
      else if (   q[9] ) out2 = 5'b01001;
      else if (   q[8] ) out2 = 5'b01000;
      else if (   q[7] ) out2 = 5'b00111;
      else if (   q[6] ) out2= 5'b00110;
      else if (   q[5] ) out2 = 5'b00101;
      else if (   q[4] ) out2 = 5'b00100;
      else if (   q[3] ) out2 = 5'b00011;
      else if (   q[2] ) out2 = 5'b00010;
      else if (   q[1] ) out2 = 5'b00001;
      else   out2 = 5'b00000;
   end
 endmodule
```

CONSTRAINTS FILE

```
set_property IOSTANDARD LVCMOS33 [get_ports {out2[5]}]
set_property IOSTANDARD LVCMOS33 [get_ports {out2[4]}]
set_property IOSTANDARD LVCMOS33 [get_ports {out2[3]}]
set_property IOSTANDARD LVCMOS33 [get_ports {out2[2]}]
set_property IOSTANDARD LVCMOS33 [get_ports {out2[1]}]
set_property PACKAGE_PIN T9 [get_ports {out2[5]}]
set_property PACKAGE_PIN T10 [get_ports {out2[4]}]
set_property PACKAGE_PIN R5 [get_ports {out2[3]}]
set_property PACKAGE_PIN T5 [get_ports {out2[2]}]
set_property PACKAGE_PIN R6 [get_ports {out2[1]}]
set_property IOSTANDARD LVCMOS33 [get_ports a]
set_property IOSTANDARD LVCMOS33 [get_ports b]
set_property IOSTANDARD LVCMOS33 [get_ports c]
set_property IOSTANDARD LVCMOS33 [get_ports D]
set_property IOSTANDARD LVCMOS33 [get_ports d]
set_property IOSTANDARD LVCMOS33 [get_ports e]
set_property IOSTANDARD LVCMOS33 [get_ports reset]
set_property IOSTANDARD LVCMOS33 [get_ports stop]
set_property PACKAGE_PIN T8 [get_ports a]
set_property PACKAGE_PIN R8 [get_ports b]
set_property PACKAGE_PIN N6 [get_ports c]
set_property PACKAGE_PIN R12 [get_ports D]
set_property PACKAGE_PIN T7 [get_ports d]
set_property PACKAGE_PIN P8 [get_ports e]
set_property PACKAGE_PIN M6 [get_ports reset]
set_property PACKAGE_PIN T12 [get_ports stop]
```

3. TDC CODE

```
 module combo (clk,start,stop,reset,a,b,c,d,e,x,out1,out2);
   input     clk, reset,start,stop;  //Input declaration
   output  [4:1] out1 ;              //Output declaration
```

```
    output x;
    reg Q;
    wire clk1;
    input a,b,c,d,e;
    output reg [5:1] out2;

  (*keep=1*) wire t[51:1];       //(*keep = 1*)=>not allow tool to optimize the buffer chain
  (*keep=1*) wire i [17:1];
   wire   [17:1]q;
   wire clk2;

  clk_wiz_1  dut( clk1,reset,x,clk);  //clocking wizard to multiply freq from 5oMHz to 200MHz
  and y1(clk2,start,clk1);        //AND gate for gating the clock frequncy activated with START

  reg [4:1] out11;
  assign out1 = out11
  always @(posedge clk2)   //Counter
  begin
         if (reset)
                out11 = 0;
       else if(start & ~stop)
               out11 = (out11 + 1'b1);
        else if(stop)
              out11 = out11;
   end
always @ ( posedge clk2 or posedge reset)    //Dff for providing start signal to delay line
  begin
     if (reset)
            Q <= 0;
   else
```

```
        Q <= 1;
end
        buff g1(stop,a,b,c,d,e,t[1]);//1  //vernier delay line
        buff g19(t[1],a,b,c,d,e,t[2]);
        buff x28(t[2],a,b,c,d,e,t[3]);
        latch g2(Q,t[3],reset,i[1]);
        Dff f1(t[3],reset,i[1],q[1]);

        buff g20(t[3],a,b,c,d,e,t[4]);//2
        buff x29(t[4],a,b,c,d,e,t[5]);
        buff g3(t[5],a,b,c,d,e,t[6]);
        latch g4(i[1],t[6],reset,i[2]);
        Dff f2(t[6],reset,i[2],q[2]);

        buff g5(t[6],a,b,c,d,e,t[7]);//3
        buff g21(t[7],a,b,c,d,e,t[8]);
        buff x30(t[8],a,b,c,d,e,t[9]);
        latch g6(i[2],t[9],reset,i[3]);
        Dff f3(t[9],reset,i[3],q[3]);

        buff g7(t[9],a,b,c,d,e,t[10]);//4
        buff g22(t[10],a,b,c,d,e,t[11]);
        buff x31(t[11],a,b,c,d,e,t[12]);
        latch g8(i[3],t[12],reset,i[4]);
        Dff f4(t[12],reset,i[4],q[4]);

        buff g9(t[12],a,b,c,d,e,t[13]);//5
        buff g23(t[13],a,b,c,d,e,t[14]);
        buff x32(t[14],a,b,c,d,e,t[15]);
        latch g10(i[4],t[15],reset,i[5]);
```

```
Dff f5(t[15],reset,i[5],q[5]);

buff g11(t[15],a,b,c,d,e,t[16]);//6
buff g24(t[16],a,b,c,d,e,t[17]);
buff x33(t[17],a,b,c,d,e,t[18]);
latch g12(i[5],t[18],reset,i[6]);
Dff f6(t[18],reset,i[6],q[6]);

buff g13(t[18],a,b,c,d,e,t[19]);//7
buff g25(t[19],a,b,c,d,e,t[20]);
buff x34(t[20],a,b,c,d,e,t[21]);
latch g14(i[6],t[21],reset,i[7]);
Dff f7(t[21],reset,i[7],q[7]);

buff g15(t[21],a,b,c,d,e,t[22]);//8
buff g26(t[22],a,b,c,d,e,t[23]);
buff x35(t[23],a,b,c,d,e,t[24]);
latch g16(i[7],t[24],reset,i[8]);
Dff f8(t[24],reset,i[8],q[8]);

buff g17(t[24],a,b,c,d,e,t[25]);//9
buff g27(t[25],a,b,c,d,e,t[26]);
buff x36(t[26],a,b,c,d,e,t[27]);
latch g18(i[8],t[27],reset,i[9]);
Dff f9(t[27],reset,i[9],q[9]);

buff g28(t[27],a,b,c,d,e,t[28]);//10
buff g29(t[28],a,b,c,d,e,t[29]);
buff x37(t[29],a,b,c,d,e,t[30]);
latch g30(i[9],t[30],reset,i[10]);
```

```
Dff f10(t[30],reset,i[10],q[10]);

buff g31(t[30],a,b,c,d,e,t[31]);//11
buff g32(t[31],a,b,c,d,e,t[32]);
buff x38(t[32],a,b,c,d,e,t[33]);
latch g33(i[10],t[33],reset,i[11]);
Dff f11(t[33],reset,i[11],q[11]);

buff g34(t[33],a,b,c,d,e,t[34]);//12
buff g35(t[34],a,b,c,d,e,t[35]);
buff x39(t[35],a,b,c,d,e,t[36]);
latch g36(i[11],t[36],reset,i[12]);
Dff f12(t[36],reset,i[12],q[12]);


buff g37(t[36],a,b,c,d,e,t[37]);//13
buff g38(t[37],a,b,c,d,e,t[38]);
buff x40(t[38],a,b,c,d,e,t[39]);
latch g39(i[12],t[39],reset,i[13]);
Dff f13(t[39],reset,i[13],q[13]);

buff g40(t[39],a,b,c,d,e,t[40]);//14
buff g41(t[40],a,b,c,d,e,t[41]);
buff x41(t[41],a,b,c,d,e,t[42]);
latch g42(i[13],t[42],reset,i[14]);
Dff f14(t[42],reset,i[14],q[14]);

buff g43(t[42],a,b,c,d,e,t[43]);//15
buff g44(t[43],a,b,c,d,e,t[44]);
buff x42(t[44],a,b,c,d,e,t[45]);
```

```
        latch g45(i[14],t[45],reset,i[15]);
        Dff f15(t[45],reset,i[15],q[15]);

        buff g46(t[45],a,b,c,d,e,t[46]);//16
        buff g47(t[46],a,b,c,d,e,t[47]);
        buff x48(t[47],a,b,c,d,e,t[48]);
        latch g49(i[15],t[48],reset,i[16]);
        Dff f16(t[48],reset,i[16],q[16]);

        buff g50(t[48],a,b,c,d,e,t[49]);//17
        buff g51(t[49],a,b,c,d,e,t[50]);
        buff x52(t[50],a,b,c,d,e,t[51]);
        latch g53(i[16],t[51],reset,i[17]);
        Dff f17(t[51],reset,i[17],q[17]);
always @(stop or reset or q)  //Encoder
 begin
      if ( reset) out2 <= 5'b00000;
  else if (   q[17] ) out2 <= 5'b10001;
  else if (   q[16] ) out2 <= 5'b10000;
  else if (   q[15] ) out2 <= 5'b01111;
  else if (   q[14] ) out2 <= 5'b01110;
  else if (   q[13] ) out2 <= 5'b01101;
  else if (   q[12] ) out2 <= 5'b01100;
  else if (   q[11] ) out2 <= 5'b01011;
  else if (   q[10] ) out2 <= 5'b01010;
  else if (   q[9] ) out2 <= 5'b01001;
  else if (   q[8] ) out2 <= 5'b01000;
  else if (   q[7] ) out2 <= 5'b00111;
  else if (   q[6] ) out2 <= 5'b00110;
  else if (   q[5] ) out2 <= 5'b00101;
```

```
   else if (   q[4] ) out2 <= 5'b00100;
   else if (   q[3] ) out2 <= 5'b00011;
   else if (   q[2] ) out2 <= 5'b00010;
   else if (   q[1] ) out2 <= 5'b00001;
   else   out2 <= 5'b00000;
  end
 endmodule
```

CONSTRAINTS FILE

```
set_property IOSTANDARD LVCMOS33 [get_ports {out1[4]}]
set_property IOSTANDARD LVCMOS33 [get_ports {out1[3]}]
set_property IOSTANDARD LVCMOS33 [get_ports {out1[2]}]
set_property IOSTANDARD LVCMOS33 [get_ports {out1[1]}]
set_property IOSTANDARD LVCMOS33 [get_ports {out2[5]}]
set_property IOSTANDARD LVCMOS33 [get_ports {out2[4]}]
set_property IOSTANDARD LVCMOS33 [get_ports {out2[3]}]
set_property IOSTANDARD LVCMOS33 [get_ports {out2[2]}]
set_property IOSTANDARD LVCMOS33 [get_ports {out2[1]}]
set_property PACKAGE_PIN K2 [get_ports {out1[4]}]
set_property PACKAGE_PIN K3 [get_ports {out1[3]}]
set_property PACKAGE_PIN L2 [get_ports {out1[2]}]
set_property PACKAGE_PIN L3 [get_ports {out1[1]}]
set_property PACKAGE_PIN T9 [get_ports {out2[5]}]
set_property PACKAGE_PIN T10 [get_ports {out2[4]}]
set_property PACKAGE_PIN R5 [get_ports {out2[3]}]
set_property PACKAGE_PIN T5 [get_ports {out2[2]}]
set_property PACKAGE_PIN R6 [get_ports {out2[1]}]
set_property PACKAGE_PIN T8 [get_ports a]
set_property PACKAGE_PIN R8 [get_ports b]
set_property PACKAGE_PIN N6 [get_ports c]
set_property PACKAGE_PIN T7 [get_ports d]
set_property PACKAGE_PIN P8 [get_ports e]
set_property PACKAGE_PIN M6 [get_ports reset]
set_property PACKAGE_PIN T12 [get_ports start]
set_property PACKAGE_PIN R12 [get_ports stop]
set_property IOSTANDARD LVCMOS33 [get_ports a]
set_property IOSTANDARD LVCMOS33 [get_ports b]
set_property IOSTANDARD LVCMOS33 [get_ports c]
set_property IOSTANDARD LVCMOS33 [get_ports d]
set_property IOSTANDARD LVCMOS33 [get_ports e]
#set_property IOSTANDARD LVCMOS33 [get_ports Q]
set_property IOSTANDARD LVCMOS33 [get_ports reset]
set_property IOSTANDARD LVCMOS33 [get_ports start]
set_property IOSTANDARD LVCMOS33 [get_ports stop]
set_property PACKAGE_PIN N11 [get_ports clk]
set_property IOSTANDARD LVCMOS33 [get_ports clk]
set_property IOSTANDARD LVCMOS33 [get_ports x]
set_property PACKAGE_PIN R7 [get_ports x]
```

SUB MODULE CODE

1.D LATCH

```
module latch (   d,        // 1-bit input pin for data
              clk1,       // 1-bit input pin for enabling the latch
              rst,       // 1-bit input pin for active-high reset
               q);    // 1-bit output pin for data output
 input d,clk1,rst;
 output reg q;
   // This always block is "always" triggered whenever en/rstn/d changes
   // If reset is asserted then output will be zero
   // Else as long as enable is high, output q follows input d
   always @ (clk1 or rst or d)
      if (rst)
         q <= 0;
      else
         if (~clk1)
            q <= d;
endmodule
```

2.D FLIP-FLOP

```
module Dff (clk,reset,data,q);
      input data, clk, reset ; //input declaration
      output q;                //output declaration
      reg q;                   //starage element defined as a reg
         always @ ( posedge clk or posedge reset)     //sensitivity list
            if (reset)
                q <= 1'b0;        // resets the Dff i.e. output is zero (active high reset)
            else
                 q <= data;       //at positive edge of clock input is stored
endmodule
```

3. BUFFER ELEMENT

```
module buff(i,a,b,c,d,e,o);
      input i,a,b,c,d,e;
      output o;
      /*as LUT is of 6 variable, tool was optimizing the design and for 17 buffer
      element it was shoing only one element which is not desired for timing analysis
      so to use one LUT as a one element all 6 dummy variables(a,b,c,d,e) are used.
```

```
   only 'i' is the input and 'o' is the output.( All a,b,c,d,e are always logic 1 */
  wire t[4:1];
  begin
      and (t[1],a,b);  // It's buffer nothing else input is 'i' and output 'o'.
     and (t[2],c,d);
     and (t[3],t[1],t[2]);
     and (t[4],e,i);
     and (o,t[3],t[4]);
   end
endmodule
```

REFERENCES

1. **Stephan Henzler.** Time-to-Digital Converter Basics, pages 13-16, 12 January 2010.
2. **Noble Nii Nortey Narku-Tetteh**. A sub-10ps time-to-digital converter with 204ns dynamic range for time-resolved imaging and ranging applications, pages 8-15, May 2014.
3. **Balamurali Radhakrishnan Naveen Wali.** Design of a Time-to-Digital Converter for an All-Digital Phase Locked Loop for the 2-GHz Band, LiTH-ISY-EX--13/4684-SE, Linkoping, pages 33-37, 2013.
4. **Mootaz ALLAM** PhD Student Pierre & Marie Curie University LIP6 Laboratory, Time based quantizers, page 8, June 2015.
5. https://www.xilinx.com/products/design-tools/vivado.html
6. **Claudio Favi and Edoardo Charbon.** A 17ps Time-to-Digital Converter implemented in 65nm FPGA Technology, page 3, Feb 24, 2009.
7. **Raguvaran Narasimman.** Implementation of a 30 ps Resolution Time to Digital Converter in FPGA, page 14, 2015.
8. **M. Maamoun, Member, IAENG S. Arami, R. Beguenane, A. Benbelkacem and A. Meraghni**. A 3ps Resolution Time-to-digital Converter in Low-cost FPGA for Laser Rangefinder, Proceding of the World Congress on Engineering 2017 Vol1 WCE 2017, July 5-7, page 3, 2017.
9. **C. Ugur, E. Bayer, N. Kurz and M. Traxler**. A 16 channel high resolution (<11 ps) Time-to-Digital Converter in a Field Programmable Gate Array. C Ugur et al 2012 JINST **7** C02004, pages 2-3, 2004.
10. **Robert McIlhenny, Milos D. Ercegovac**. On Using 1-out-of-n Codes for (p,q) Counter Implementations, IEEE transaction, page 2,1996.
11. **G.S.Jovanovic, M. K. Stojcev.** Vernier’s Delay Line Time–to–Digital Converter. APPL. MATH. INFORM. AND MECH. vol. 1, 1 (2009), pages 11-13, 2009.
12. **Henzler,S.** http://www.springer.com/978-90-481-8627-3**,** Time-to-Digital converter.