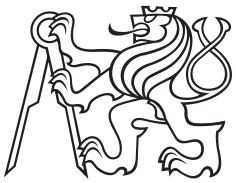
Master Thesis



Czech Technical University in Prague

F3

Faculty of Electrical Engineering
Department of Measurement

Implementation of a method for comparing two time scales on an FPGA circuit

Bc. Michal Špaček

Supervisor: Ing. Radek Sedláček, Ph.D.
Field of study: Cybernetics and Robotics
May 2021

ZADÁNÍ DIPLOMOVÉ PRÁCE

I. OSOBNÍ A STUDIJNÍ ÚDAJE

Příjmení: **Špaček** Jméno: **Michal** Osobní číslo: **466107**

Fakulta/ústav: **Fakulta elektrotechnická**

Zadávající katedra/ústav: **Katedra měření**

Studijní program: **Kybernetika a robotika**

Studijní obor: **Kybernetika a robotika**

II. ÚDAJE K DIPLOMOVÉ PRÁCI

Název diplomové práce:

Implementace metody pro porovnání dvou časových stupnic na FPGA obvodu

Název diplomové práce anglicky:

Implementation of a method for comparing two time scales on an FPGA circuit

Pokyny pro vypracování:

1. Na základě rešerše oborných článků navrhněte a implementujte na FPGA obvodu tzv. TDC převodník (time-to-digital convertor) s cílem dosažení co nejlepšího rozlišení alespoň v řádech desítek ps. Ověřte jeho metrologické parametry (rozlišení a linearitu rozsahu).
2. Navrhněte nový nebo použijte vhodný hardware využívající FPGA obvod pro implementaci metody pro porovnání dvou referenčních časových stupnic založené na vyhodnocení 1 PPS impulzů. Pro měření časových zpoždění použijte vyvinutý TDC převodník uvnitř FPGA obvodu.
3. Vytvořte patřičný firmware pro FPGA s podporou úplné konfigurace SFP/SFP+ modulů pomocí MDIO rozhraní.
4. Vytvořte aplikaci pro PC umožňující uživatelsky přívětivé ovládání celého systému včetně vizualizace a vyhodnocení naměřených dat.

Seznam doporučené literatury:

[1] TANCOCK, Scott, Ekin ARABUL a Naim DAHNOUN: A Review of New Time-to-Digital Conversion Techniques. IEEE Transactions on Instrumentation and Measurement [online]. 2019, 68(10), 3406-3417 [cit. 2021-01-25]. ISSN 0018-9456. Dostupné z: doi:10.1109/TIM.2019.2936717.
[2] Dierikx, E. – Xie, Y.: White Rabbit Good Practice Guide. Dutch Metrology Institute, May 2019.
[3] Grzegorz Daniluk: White Rabbit calibration procedure. CERN BECO-HT, October 2019. version 1.1.
[4] INF-8074: SFP (Small Formfactor Pluggable) Transceiver [online]. In: SNIA, 2020, 2001-05-12 [cit. 2021-01-25]. Dostupné z: https://members.snia.org/document/dl/26184.

Jméno a pracoviště vedoucí(ho) diplomové práce:

Ing. Radek Sedláček, Ph.D., katedra měření FEL

Jméno a pracoviště druhé(ho) vedoucí(ho) nebo konzultanta(ky) diplomové práce:

Datum zadání diplomové práce: **25.01.2021** Termín odevzdání diplomové práce: _____

Platnost zadání diplomové práce:
do konce letního semestru 2021/2022

Ing. Radek Sedláček, Ph.D.		prof. Mgr. Petr Páta, Ph.D.
podpis vedoucí(ho) práce	podpis vedoucí(ho) ústavu/katedry	podpis děkana(ky)

III. PŘEVZETÍ ZADÁNÍ

Diplomant bere na vědomí, že je povinen vypracovat diplomovou práci samostatně, bez cizí pomoci, s výjimkou poskytnutých konzultací.
Seznam použité literatury, jiných pramenů a jmen konzultantů je třeba uvést v diplomové práci.

. _____
Datum převzetí zadání

_____
Podpis studenta

Acknowledgements

I would like to thank my supervisor Ing. Radek Sedláček, Ph.D. for his valuable help during the development of this master thesis and for the provided technical facilities. My grateful thanks are also extended to doc. Ing. Jaroslav Roztočil, CSc. for his helpful participation in the discussions. Finally, I wish to thank my family for their support throughout my study.

Declaration

I declare that the presented work was developed independently and that I have listed all sources of information used within it in accordance with the methodical instructions for observing the ethical principles in the preparation of university theses.

Prague, May 21, 2021 .....
Signature

Abstract

The thesis is aiming at the implementation of the TDC (Time-to-Digital Converter) inside FPGA. The second part of the thesis focuses on time transfer through an optical fiber and its comparison with the local time scale.

The implemented TDC is composed of a delay line and a counter. A combination of these two structures provides excellent resolution and a wide measuring range. The kit Terasic DE10-Standard is used as development hardware. The designed extension board is then connected to the development kit. The extension board aims to provide a suitable interface that is not fulfilled by the standalone development board. Processor Nios controls the whole system for time scales comparison. It also provides access to the SFP module configuration memory and establishes communication with the PC. The user application on PC ensures the visualization of the measured data with the export option and access to the SFP modules configuration.

Keywords: TDC, comparison of time scales, FPGA

Supervisor: Ing. Radek Sedláček, Ph.D.

Abstrakt

Práce se zabývá implementací TDC (Time-to-Digital Converter) uvnitř FPGA obvodu. Druhá část práce se zaměřuje na přenos přesného času prostřednictvím optických vláken a jeho následné porovnání s lokální časovou stupnicí.

Implementovaná TDC struktura je kombinací zpožďovací linky a čítače. Spojení těchto dvou struktur poskytuje výborné rozlišení a široký měřící rozsah. Pro vývoj byl použit kit Terasic DE10-Standard. K němu je následně připojena navržená rozšiřující deska. Ta má za úkol poskytnout vhodné rozhraní, které nám samotný vývojový kit neposkytuje. Celý systém je řízen procesorem Nios. Dále procesor Nios umožňuje přístup ke konfigurační paměti SFP modulů a také zajišťuje komunikaci s PC. Uživatelská aplikace na PC poskytuje vizualizaci měřených dat spolu s jejich následným exportováním a také přístup ke konfiguraci SFP modulů.

Klíčová slova: TDC, porovnání časových stupnic, FPGA

Překlad názvu: Implementace metody pro porovnání dvou časových stupnic na FPGA obvodu

Contents

1 Introduction 1
1.1 Motivation 1
1.2 Comparison of time scales 2
1.2.1 GPS time transfer 2
1.2.2 White Rabbit 4

2 Theoretical analysis of the TDC on FPGA 5
2.1 Delay line 6
2.2 Vernier method 8
2.3 Vernier method using a delay line 10
2.4 Wave union 10
2.4.1 Wave union A 11
2.5 TDC using fine and coarse time measurement 12
2.6 Successive-Approximation TDC 13

3 Proposed time scales comparison 17

4 Realization 21
4.1 Extension board 23
4.1.1 Input signals 24
4.1.2 Output signals 25
4.1.3 SFP modules 26
4.1.4 UART interface 26
4.1.5 Indications 28
4.1.6 Final state 28
4.2 VHDL development 30
4.2.1 TDC design 30
4.2.2 Delay generator 36
4.2.3 Modulation and demodulation of PPS signal 36
4.2.4 Processor Nios II 38
4.3 Processor Nios II firmware 42
4.4 User application 43
4.4.1 UART communication 44
4.4.2 Time measurement 44
4.4.3 Data visualization 44
4.4.4 Delay generator control 44
4.4.5 Export of the data 45
4.4.6 SFP configuration 45
4.4.7 Application customization . . . 46

5 Results 47
5.1 Delay line validation 47
5.2 TDC validation 50

6 Conclusion 55

Bibliography 57

A Meaning of the used shortcuts 61

Figures

1.1 Common-View Method 3
1.2 Clock Transportation 3
1.3 Time and frequency transfer by White Rabbit [1] 4

2.1 Example of an ideal and real transfer function of the ADC 6
2.2 Basic delay line arrangement [2] . 7
2.3 Example of the delay elements outputs state in the time 8
2.4 Block diagram of Vernier method [2] 8
2.5 A timing diagram of the Vernier method [2] 9
2.6 Vernier method using a delay line [3] 10
2.7 A timing diagram of wave union A 11
2.8 Possible structure of the wave union A 12
2.9 Block diagram of TDC 12
2.10 A timing diagram of TDC 13
2.11 SAR ADC block diagram [4] . . 14
2.12 SAR TDC block diagram [4] . . 15

3.1 Block diagram of the complete system 18
3.2 The critical path of the PPS signal in the system 19

4.1 Development kit Terasic DE-10 standard [5] 22
4.2 Peripheries of the development kit Terasic DE-10 standard [5] 22
4.3 Block diagram of the extension board 23
4.4 Reference clock connection 24
4.5 PPS signal connection 25
4.6 Output clock signal connection . 25
4.7 SFP module connection 26
4.8 UART interface connection 27
4.9 UART digital isolator connection 28
4.10 LEDs indication connection . . . 28
4.11 Top view on final PCB design . 29
4.12 Bottom view on final PCB design 29
4.13 The idea of the adder delay line 30
4.14 Chained one-bit full adders . . . 31
4.15 One-bit full adder connection . . 31
4.16 Design of the delay line in Quartus 32
4.17 Prior encoder design in Quartus 33
4.18 Counter design in Quartus 33
4.19 Control logic in Quartus 34
4.20 A timing diagram of the control logic 35
4.21 Normalization design in Quartus 36
4.22 Delay generator in Quartus . . . 36
4.23 Scrambler connection 38
4.24 Descrambler connection 38
4.25 Scrambler and descrambler in Quartus 38
4.26 Nios II configuration in Platform Designer 40
4.27 Nios II in Quartus 41
4.28 Message frame of the communication protocol 42
4.29 Control tab of the user application 43
4.30 SFP configuration in user application 46
4.31 Qt application in light mode . . 46

5.1 Connection of the measurement chain during the evaluation of the delay line 48
5.2 Measured transfer function of the delay line 48
5.3 Transfer function of the delay line (original) 49
5.4 Transfer function of the delay line (shifted) 50
5.5 Connection of the measurement chain during the evaluation of the TDC 51
5.6 Validation of the TDC (before calibration) 52
5.7 Validation of the TDC (after calibration) 53

Tables

1.1 Available TDC solutions overview 2

4.1 The FPGA chip specification [6] 21
4.2 Nios II version properties [7] ... 39

A.1 Shortcuts overview 61

Chapter 1

Introduction

This work aims to create a TDC (Time-to-Digital Converter) implemented inside an FPGA circuit. The converter's interface is supposed to be conventional with two inputs and one output. The measurement subject is the time difference between the rising edges of the signals on the inputs. The mentioned time difference is then presented on the output in binary form. Deployment of the implemented TDC is expected in the second part of this work. The effort is to develop a device that compares two time scales by evaluating their PPS (Pulse Per Second) signal. The development of this entire system also requires a user application that serves control over the whole system. The application should likewise visualize the measured delays with their exportation. It is assumed that an optical fiber will connect the stations with time scales. Therefore, the final hardware design will be equipped with the SFP module interface. Finally, the possibility to access the configuration of the SFP module is expected.

1.1 Motivation

Measuring the time between two events is critical in several applications. An example is laser distance measurement. In this case, it is important to measure the time difference between the transmitted and received beam. Then, together with the knowledge of the speed of propagation of the beam in a given medium, we can determine the absolute distance between the emitter and the object from which the beam was back reflected to the emitter. The more accurately we can measure the time between the transmitted and received beam, the more accurately we can determine the absolute distance of the sensor from the object. Another application of the TDC may be magnetic field measurement. There is a type of magnetometer that measures a magnetic field based on the Larmor frequency of the working medium. This frequency is directly proportional to the magnetic field. A TDC can be used to determine this frequency.

There are few ready-made circuit solutions on the market providing time measurement. A little overview of the available solutions proceeds in Table 1.1 with their basic parameters.

Producer	Texas Instruments	ScioSense	ScioSense
Product number	TDC7201	AS6500	TDC-GPX
Maximum range	8 ms	16 s	10 µs (2 stop channels mode)
Resolution	55 ps	10 ps (High res. mode)	10 ps (2 stop channels mode)
Channels	2	up to 4 stop channels	up to 8 stop channels
Price	≈100 Kč	≈500 Kč	≈3700 Kč

Table 1.1: Available TDC solutions overview.

Performance comparison of each TDC circuit to each other is not that easy. More complex circuits can be set to various modes, improving one parameter and at the same time worsening some other parameters. For example, the last circuit in the above table, ScioSense TDC-GPX, loses six-stop channels in the best mode for its measurement resolution. The maximum range is also lowered beside a few other parameters. Therefore, it depends on the specific usage of the desired TDC.

However, these ready-made solutions cannot be modified according to specific needs or added to an already finished system. From this perspective, are TDCs implemented at the FPGA preferable. According to [8], a TDC with a resolution in the tens of picoseconds can be implemented on the FPGA. According to the mentioned publication, it is possible to build a competitive TDC, which can then be, to some extent, universally deployed on various FPGA circuits without additional requirements for external hardware.

1.2 Comparison of time scales

1.2.1 GPS time transfer

Nowadays, the most frequently used method for comparing time scales is GPS time transfer [9]. The method is based on mutual indirect comparison through the received GPS signal at both measurement locations. The cited publication describes more options from which two of them are most interesting for accurate time transfer.

The first mode is called the Common-View Method. This method using one particular satellite for its function. Direct signal visibility of the involved stations to the used satellite is a necessity. Stations involved in the comparison are receiving GPS data from the same satellite, and they compare the local clock with received time data from the satellite. Then are these local comparison data process across each station. The cited publication states the accuracy of this method about 1 ns. The idea scheme of this method is shown below in Figure 1.1.

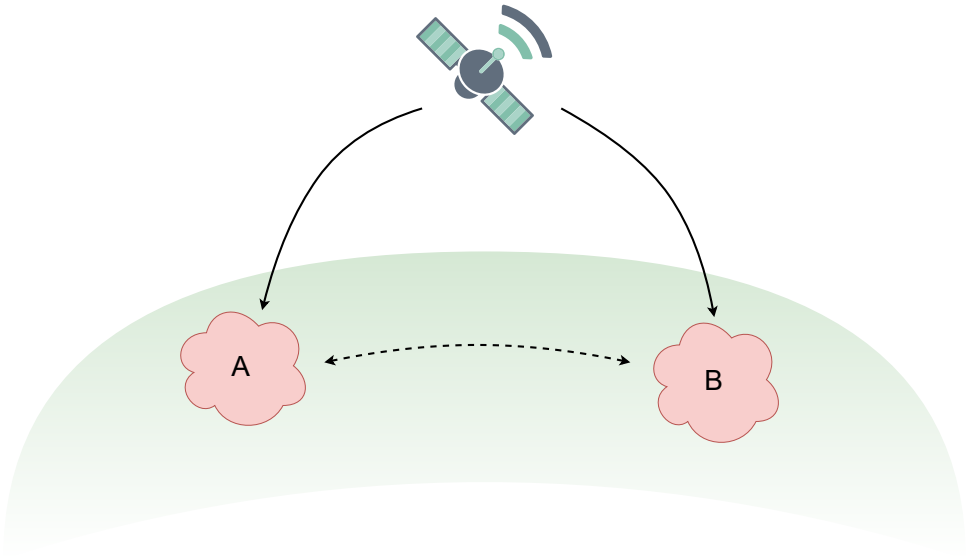


Figure 1.1: Common-View Method.

The second method is called Clock Transportation. The method is based on successive observation and can be used for comparison stations located anywhere on the earth, where is GPS signal of the same satellite or group of the satellite available with a time delay at most 12 h. Unfortunately, this method could suffer from satellite clock instability which should be about 5 ns over 12 h. The scheme of the method configuration is shown in Figure 1.2.

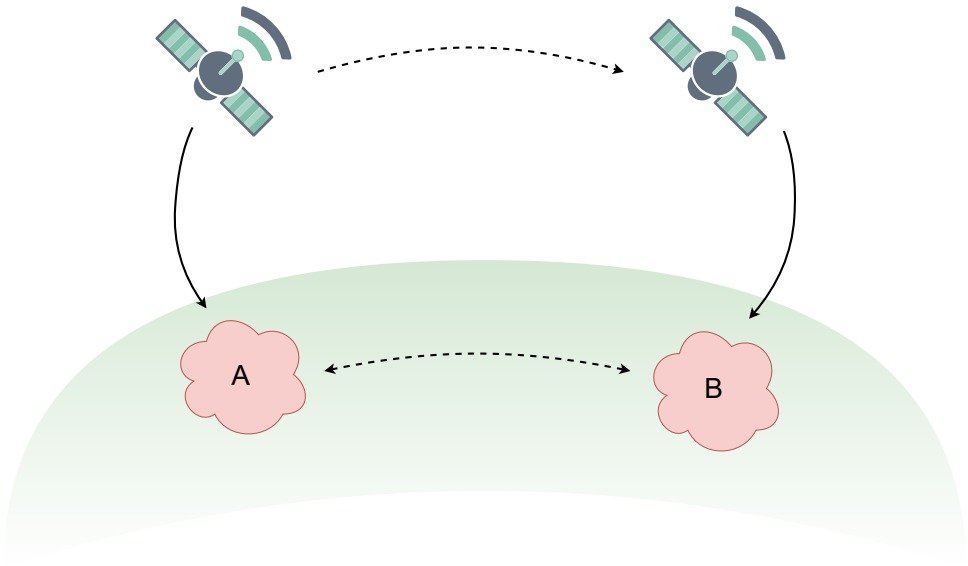


Figure 1.2: Clock Transportation.

1.2.2 White Rabbit

Another method recently used for time synchronization is White Rabbit (WR) [10]. This method was developed at CERN to replace its current timing system. The project is based on open standards, open software and hardware. It improves the original PTP (Precision Time Protocol) itself and fulfils the specification of Synchronous Ethernet. Thanks to that, it is possible to obtain sub-nanosecond accuracy. This excellent accuracy is achieved by covering three main fields. Application of the PTP, syntonization on the physical layer, and precise phase measurement [11]. The simplified example of the time and frequency transfer using White Rabbit is pictured in Figure 1.3. We consider two stations equipped with an atomic clock. The stations are generally located at different places. The WR Grandmaster and WR Slave are linked via optical fiber. Local atomic clock is used to provide the reference clock for Grandmaster. PPS signal provided by the atomic clock is also routed to the Grandmaster. Thanks to the White Rabbit method, the slave node is synchronized with the reference clock of the Grandmaster. Synchronization across the described topology makes it possible to reconstruct the Grandmaster's PPS signal and the clock signal at the slave side. The White Rabbit method should ensure determinism, accuracy and reliability. Finally, we can compare the local PPS signal at station B with the PPS signal generated at station A.

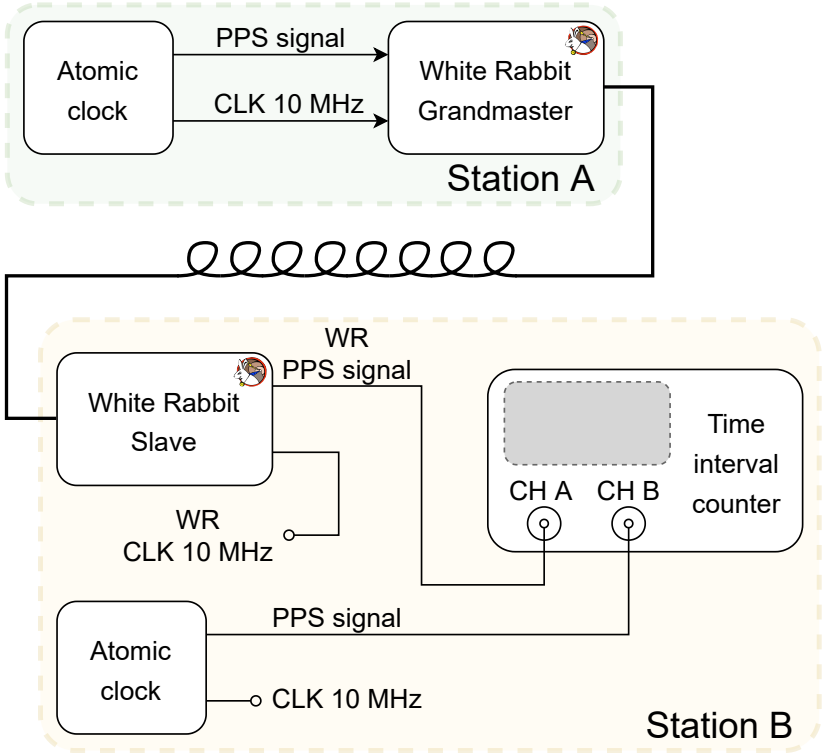


Figure 1.3: Time and frequency transfer by White Rabbit [1].

Chapter 2

Theoretical analysis of the TDC on FPGA

There are several options how to implement a Time-to-Digital Converter on FPGA. When choosing a suitable implementation for a given application, several aspects play a role. One of them is the maximum range that the converter can cover. Other such parameters are the resolution of the given measurement or the dead time of the converter. Dead time determines how many measurements we can perform in a given time interval. Additionally, worth mentioning the differential and integral nonlinearity.

Equation (2.1) defines the differential nonlinearity of the converter. It defines the width deviation of the j-th step of the actual and ideal conversion characteristics.

$$DNL_j = \frac{q_j - q}{q}, \tag{2.1}$$

where q is the ideal width of one LSB, q_j is the actual width of j-th step of the conversion characteristic.

Integral nonlinearity can be described by (2.2). It is the accumulated error of the real conversion characteristic in the j-th step.

$$INL_j = \sum_{i=0}^{j} DNL_i, \tag{2.2}$$

where DNL_i is a differential nonlinearity in i-th step.

For better understanding, Figure 2.1 shows an example of an ideal and real transfer function of the ADC.

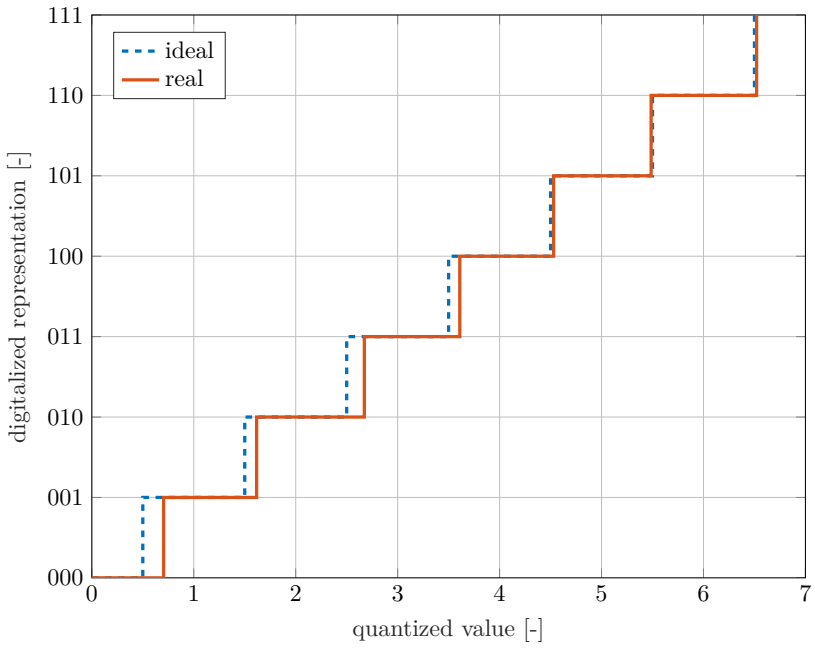


Figure 2.1: Example of an ideal and real transfer function of the ADC.

A comparison of various options for implementing the TDC is described in publication [8]. This chapter describes the principles of the most important structures for TDC implementation.

2.1 Delay line

The fundamental block of this method is the logic member, which allows copying the state of the input to its output with some known delay τ. By grouping these members into a chain structure, we obtain a delay line providing a total delay of $n \cdot \tau$, where n is the number of individual delay members. Moreover, the buffer circuit is connected to the output of each delay element. Buffer circuit can be realized, for example, as a D flip-flop. The final realization is the delay line which output is connected to the buffer line. There is an assumption that we measure the time between two rising edges. Signal with the earlier rising edge is labelled as a start signal, and this signal is routed to the input of the delay line. The second signal is the stop signal and triggers the input clock of the D flip-flop. This connection to the clock input of each buffer element ensures the capture of the state of the delay line at the moment of arrival stop signal's rising edge. The captured state of the delay line then proceeds to further processing. The number of set delay elements in the delay

line is proportional to the time between start and stop rising edge signals. The described structure is captured in Figure 2.2. In this way, it is possible to quantify the time difference between two consecutive events. Theoretically, the resolution of such a converter is given by the delay of one delay element.

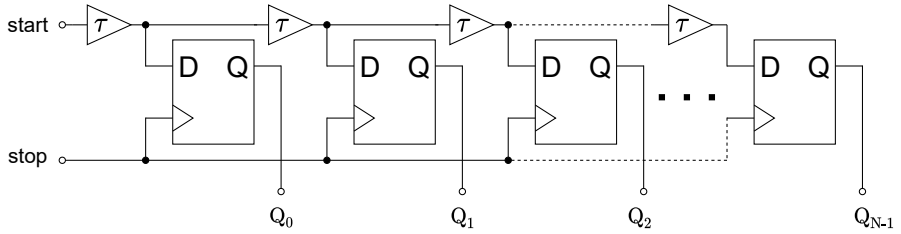


Figure 2.2: Basic delay line arrangement [2].

The resulting delay measured by the above structure is encoded in the thermometer code. This output form is disadvantageous for further processing. In addition, thermometer code representation is also uneconomical from a memory point of view. For example, we have a delay line with 1024 elements. To store the measured delay, we would have to use 1024 bits of memory. This is several times more compared to storing the same value in the classical binary form. We only need 10 bits to store the result in binary form. Therefore, the encoder is an integral part of the TDC design.

The converter can be realized by construction, which detects the transition of the logic level from 1 to 0 on the output of the D flip-flops line after arrival the rising edge of the stop signal. The output of the converter is then the position of this transition in the binary representation. Expression (2.3) then gives the resulting delay.

$$\Delta T = n \cdot \tau, \tag{2.3}$$

where n is the number of delay elements through which the start signal passed before the arrival of the stop signal.

Figure 2.3 clearly shows the signal propagation through the delay line in time. It can be seen how the propagating signal gradually affects the output of the delay elements over a period of τ.

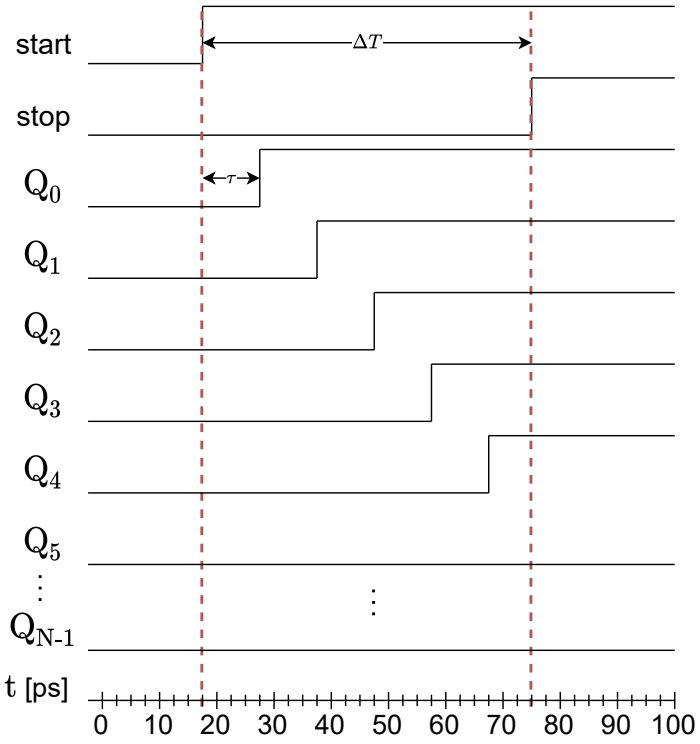


Figure 2.3: Example of the delay elements outputs state in the time.

2.2 Vernier method

Another way to measure the time difference of events within a gate array is the Vernier method. There are several versions of this method that use the various resources provided by the gate array. One of the versions we will present here uses two oscillators with different oscillation frequencies f_1 and f_2 for its function [2]. A block diagram representing the implementation of this structure can be seen in Figure 2.4.

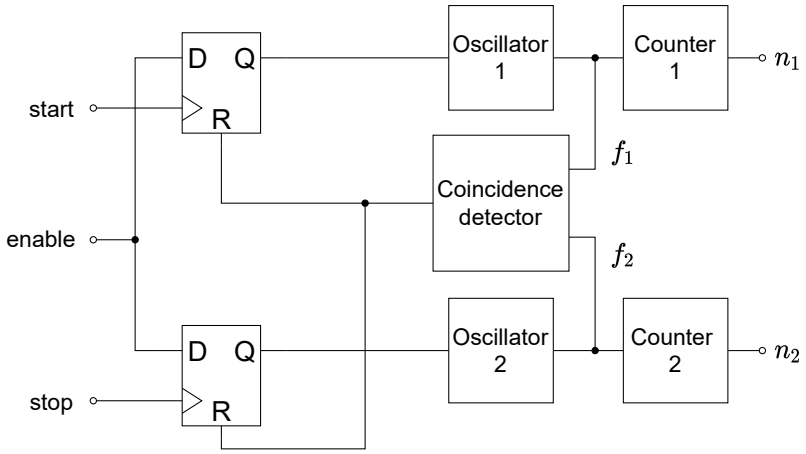


Figure 2.4: Block diagram of Vernier method [2].

The design contains two oscillators. Oscillator 1 is activated at the rising edge of the start signal, and oscillator 2 is activated at the rising edge of the stop signal. The appropriate oscillator generates the signal with a defined frequency after activation. Counter 1 and counter 2 process the output of both oscillators. Furthermore, the coincidence detector is also connected to the output of both oscillators. The coincidence detector ensures the stopping of both oscillators at the moment of concurrence of both signals. If this situation occurs, we have all the necessary information to calculate the resulting time difference according to (2.5). Described sequence of signals is captured in Figure 2.5.

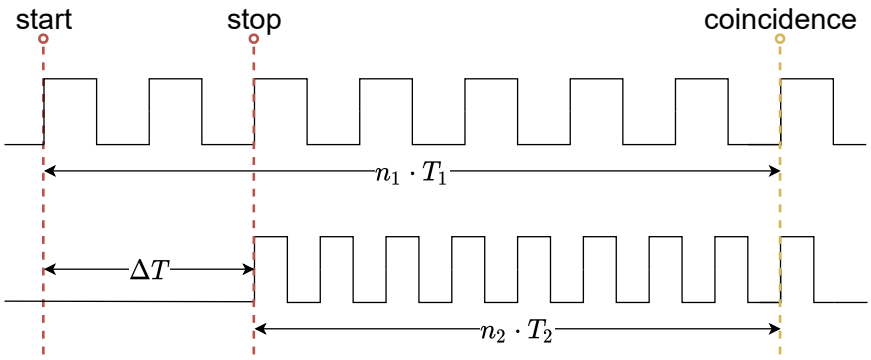


Figure 2.5: A timing diagram of the Vernier method [2].

$$T_1 = \frac{1}{f_1} \qquad T_2 = \frac{1}{f_2} \tag{2.4}$$

$$\Delta T = T_{\text{STOP}} - T_{\text{START}} = (n_1 - 1) \cdot T_1 - (n_2 - 1) \cdot T_2, \tag{2.5}$$

where n_1 and n_2 is the number of recorded signal periods having the frequency f_1 and f_2.

Equation (2.6) gives the theoretical resolution of this method. Equation (2.7) gives the maximum possible time that we can measure in this way.

$$r = T_1 - T_2 \tag{2.6}$$

$$T_{\text{MAX}} = \frac{T_1 \cdot T_2}{r} \tag{2.7}$$

2.3 Vernier method using a delay line

The principle of the Vernier method using the delay line [3] is very similar to the classical Vernier method described in Section 2.2. The basic structure consists of two delay lines and a series of D flip-flops. The elementary delays of lines A and B are denoted as τ_1 and τ_2. Assume a delay of $\tau_1 > \tau_2$. We can see the connection of the described structure of the measuring chain in Figure 2.6. A start pulse is applied to the delay line with the elementary delay τ_1. The output of each delay member of this line is connected to the input respective D flip-flop circuit. The stop pulse is applied to a delay line with a delay of τ_2, and the output of each delay member is applied to the clock input of the given flip-flop circuit. After reaching the concurrence of the start and stop signals, it is possible to evaluate the output state $Q_0, \ldots, Q_{N-1}$.

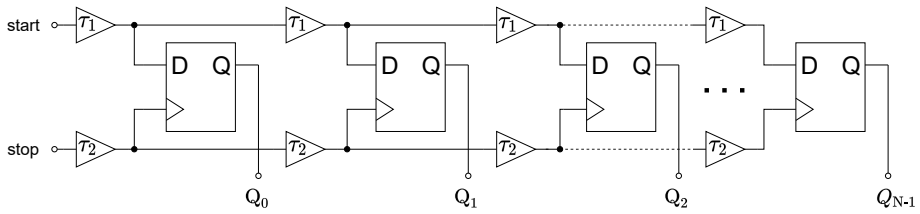


Figure 2.6: Vernier method using a delay line [3].

Equation (2.8) defines the resolution of such a converter. We calculate the resulting delay ΔT according to (2.9).

$$r_\tau = \tau_1 - \tau_2 \tag{2.8}$$

$$\Delta T = n \cdot r_\tau, \tag{2.9}$$

where n is the number of output bits $Q_0, \ldots, Q_{N-1}$ in the logical one state.

Equation (2.10) defines the maximum possible delay that this type of Time-to-Digital Converter can measure.

$$T_{\mathrm{MAX}} = N \cdot r_\tau, \tag{2.10}$$

where N is the number of delay elements of one delay line.

2.4 Wave union

Wave union is a method that improves the properties of a TDC using a classical form of a delay line. The delay line implemented on the FPGA may suffer from the strong nonlinearity of the differential conversion characteristic. This is mainly due to the very different delays of the individual elementary delay members. This problem can be reduced, to some extent, by using the wave union method [12].

2.4.1 Wave union A

Wave union type A uses a wave union pattern called FSR (Finite Step Response). The pattern used to determine the time delay has a finite number of logic level transitions 1-0 and 0-1. The delay line is divided into two parts. The first part stores the pattern and propagates it through the delay line. The second part acts only as a basic delay line.

The prepared pattern is released at the time of the start signal arrival. This pattern is propagating through the delay line until the stop signal arrives. When the stop signal arrives, the state of the line is saved for subsequent processing by the encoder. After capturing the state of the line, the wave union pattern is reset to the first part of the delay line. Subsequently, the structure is again ready for further measurements.

Position of the transitions of the logical levels of the pattern is essential for the evaluation of measured delay. Subsequently, using the known initial position of the pattern, it is possible to determine the absolute shift of the individual logical transitions of the pattern during the measurement period. For a better understanding, we can consider a pattern with two 0-1 transitions. Graphically, this case is processed in Figure 2.7. Equation (2.11) determines the resulting delay of such an arrangement.

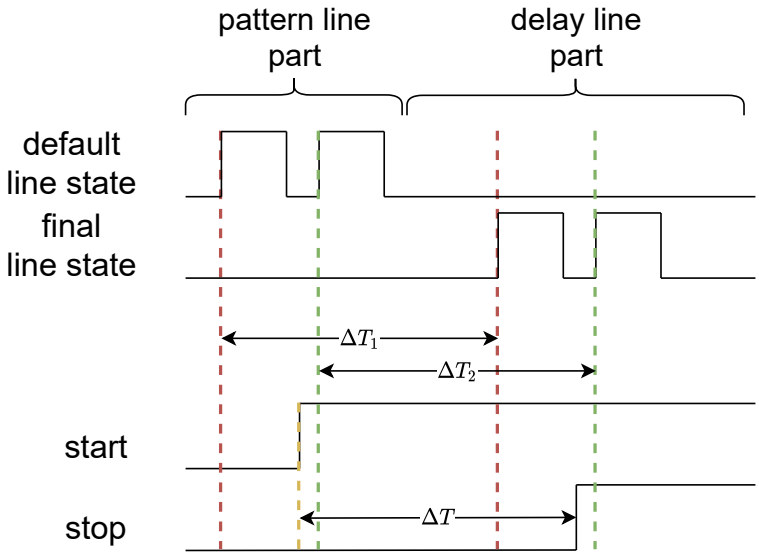


Figure 2.7: A timing diagram of wave union A.

$$\Delta T = \frac{\Delta T_1 + \Delta T_2}{2}, \tag{2.11}$$

where ΔT_1 is the absolute shift of the first logical transition and ΔT_2 is the absolute shift of the second logical transition of the pattern in the delay line.

An example of a possible implementation of this described structure at the hardware level is shown in Figure 2.8.

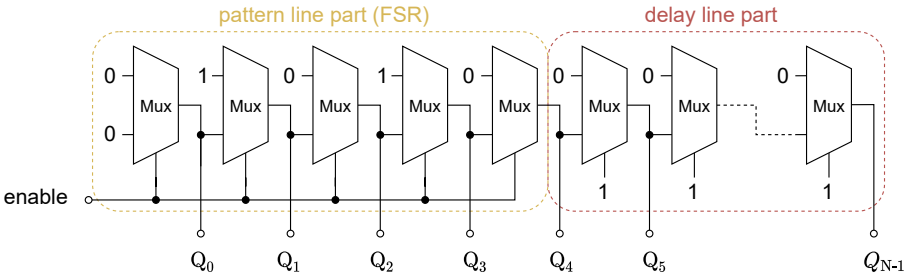


Figure 2.8: Possible structure of the wave union A [8].

2.5 TDC using fine and coarse time measurement

The resolution of the classical delay line is given by the delay of each delay member in the chain structure. The aim is to use an element with the least possible delay to achieve a good resolution. On the other hand, this reduces the maximum possible measurable time. Dividing the measurement into two parts can solve the problem. One part would measure long time events with relatively low resolution, and the second part would provide a short measurement with relatively high resolution. Combining these two parts creates a design that would measure long-time events with relatively high resolution.

If we proceed to a specific implementation, the described coarse time measurement can be provided by a counter, which counts the rising edges of the clock signal. In this case, the resolution is given by the clock signal period, and the size of the counter gives the maximum possible measured time. A classical delay line realizes the part ensuring fine time measurement. The described structure can be seen in Figure 2.9. The timing diagram of this structure is shown in Figure 2.10.

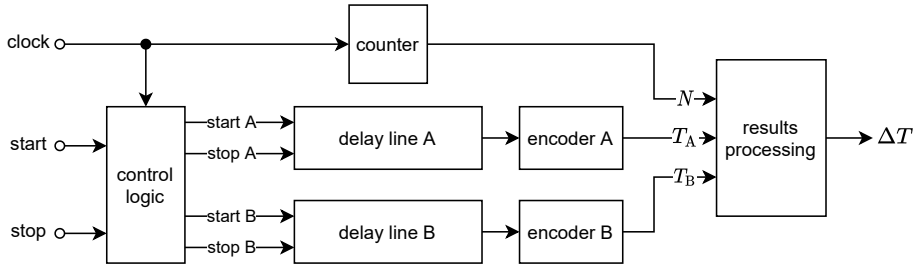


Figure 2.9: Block diagram of TDC.

In general, the start and stop signals do not have to be synchronous with the clock pulse. Therefore, two delay lines are considered in the design. Delay line A is activated at the arrival of the start signal and captured at the moment of the rising edge of the clock signal. We measure the time difference between the rising edge of the clock and the start signal. We denote this time difference as time T_{A}. We will keep the resulting value for later processing. Next comes a counter, which records with a coarse step the number of passed rising edges of the clock signal. The clock signal period is defined as T_{clk}. At the rising edge of the stop signal, the delay line B is activated, and the time difference between the stop signal and the following rising edge of the clock pulse is measured. We denote this time difference as time T_{B}. The resulting time difference ΔT is then obtained from (2.12).

$$\Delta T = T_{\mathrm{A}} + N \cdot T_{\mathrm{clk}} - T_{\mathrm{B}} \tag{2.12}$$

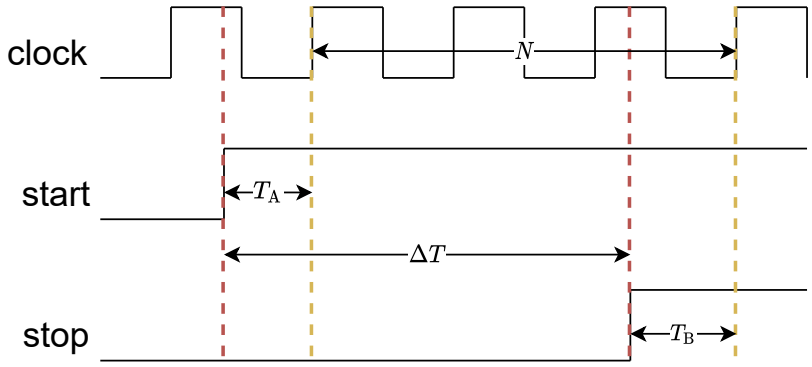


Figure 2.10: A timing diagram of TDC.

2.6 Successive-Approximation TDC

An analogy of the Successive-Approximation (SAR) TDC structure is a SAR Analog-to-Digital Converter (ADC). The function of the SAR ADC we can use for a better explanation of its TDC alternative. In Figure 2.11, we can see the structure of the SAR ADC. The key part of the SAR ADC is a Digital-to-Analog Converter (DAC). The input circuit S&H (Sample and Hold) takes a sample of the measured signal and retains the sampled signal level at its output until a new sample is taken. Subsequently, this signal level is compared with the output level of the signal from the DAC. The evaluation logic, which sets the digital word for the DAC, is driven by this comparison. After finding the closest possible output value of the DAC to the input one, the digital input word of the DAC is declared as the resulting digital representation of the quantized signal. Searching the correct combination of a digital word follows the steps of a binary search algorithm [4].

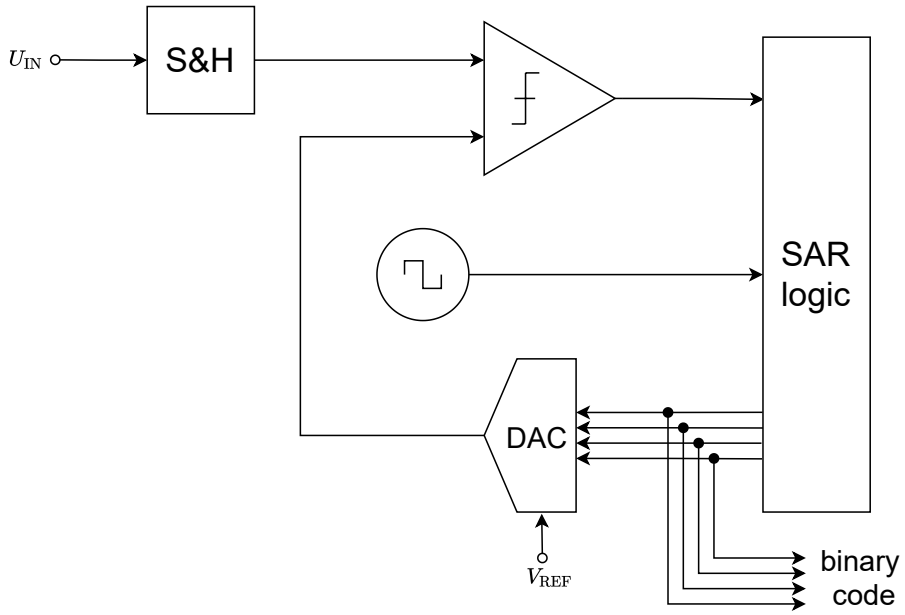


Figure 2.11: SAR ADC block diagram [4].

In the first step, the MSB bit is set to a logic one, which generates a signal of amplitude $\frac{1}{2}V_{REF}$ at the output of the DA converter. This level is then compared with the sampled level of the signal. If it is less than the quantized signal level, logic 1 remains in the MSB bit and moves to the next bit in the sequence. Otherwise, the MSB bit is set to logical zero. By the same process is evaluated each bit of the resulted number.

Now to the SAR TDC itself. The DAC is here represented by the delay line, in which the multiplexer selects the particular output. The delay line input is connected to the start signal. A D flip-flop is used here for comparison. The output of one of the delay members is connected to its data input multiplexer. A stop signal is connected to its clock input. If the start signal is delayed at the output of the selected delay member by more than the delayed stop signal, the D circuit output will be 0. Otherwise, its output will be 1. Again, as in SAR ADC, searching for a suitable digital approximation of the present delay between a start and a stop signal is performed by the binary search method [4]. Figure 2.12 shows the structure of the SAR TDC.

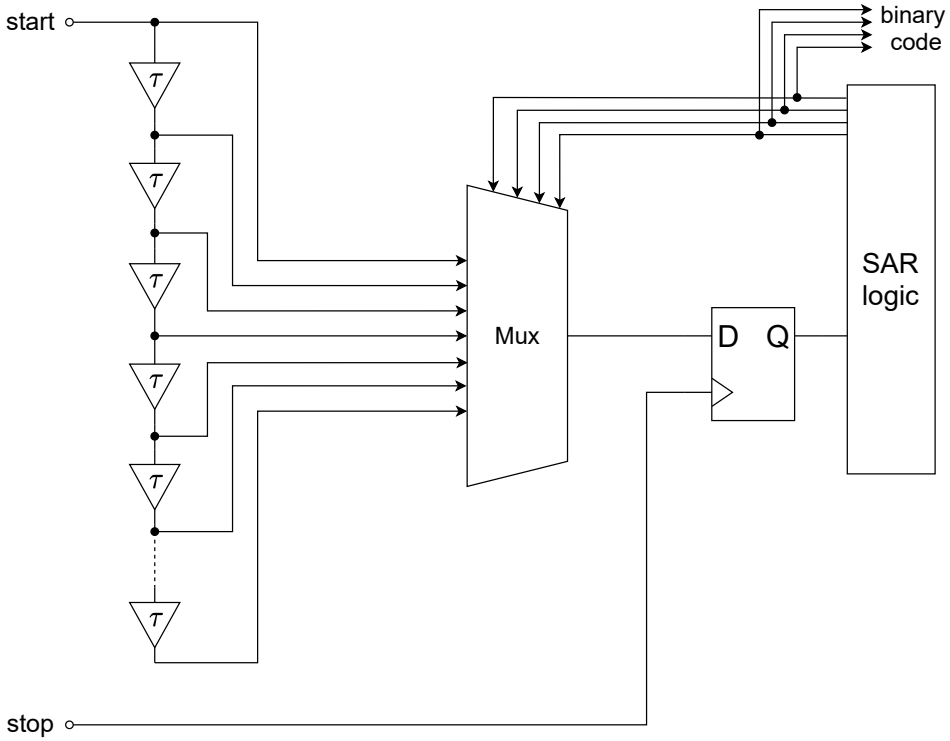


Figure 2.12: SAR TDC block diagram [4].

The evaluated domain puts different requirements on the input signal. In the case of SAR ADC, the input signal can be sampled by S&H unit, and for several comparisons, in the measurement period, we can suppose a constant input value. On the other hand, this assumption can not be made in the case of the SAR TDC. We cannot hold the input signal timing for all comparisons performed in one measurement period. Therefore, this type of TDC is suitable for the stable repeating delay, which has no radical change over the short time horizon.

Chapter 3

Proposed time scales comparison

The idea is to bring the PPS signal from the distant station to the local one. Then should be performed a comparison of the local PPS rising edge signal and the rising edge of the received PPS signal from a distant station. The block diagram in Figure 3.1 provides better insight into the whole intended measurement chain.

Each station has to be equipped with an atomic clock which we want to compare. The FPGA is there as a central control unit. A dedicated high-speed interface connects the PPS signal and the reference clock signal from an atomic clock to the FPGA. The reference clock is then used as the main clock signal for FPGA for its excellent stability. The SFP module is used in the design for conversion between the optical and electrical domain. The last part is the UART interface which provides a connection to the PC application. A user application manages the control over the whole system. The connection by optical fiber between stations is assumed.

The user from the PC application sets the role of each station. The function of the slave station is to provide its PPS signal to the master station through an optical fiber. The master station should compare its local PPS signal from the atomic clock with the received slave PPS signal. The result of the comparison is the measured delay between rising edges of the PPS signals. These results should then master station provide to the PC application. Measured data can be then exported and analyzed in more detail.

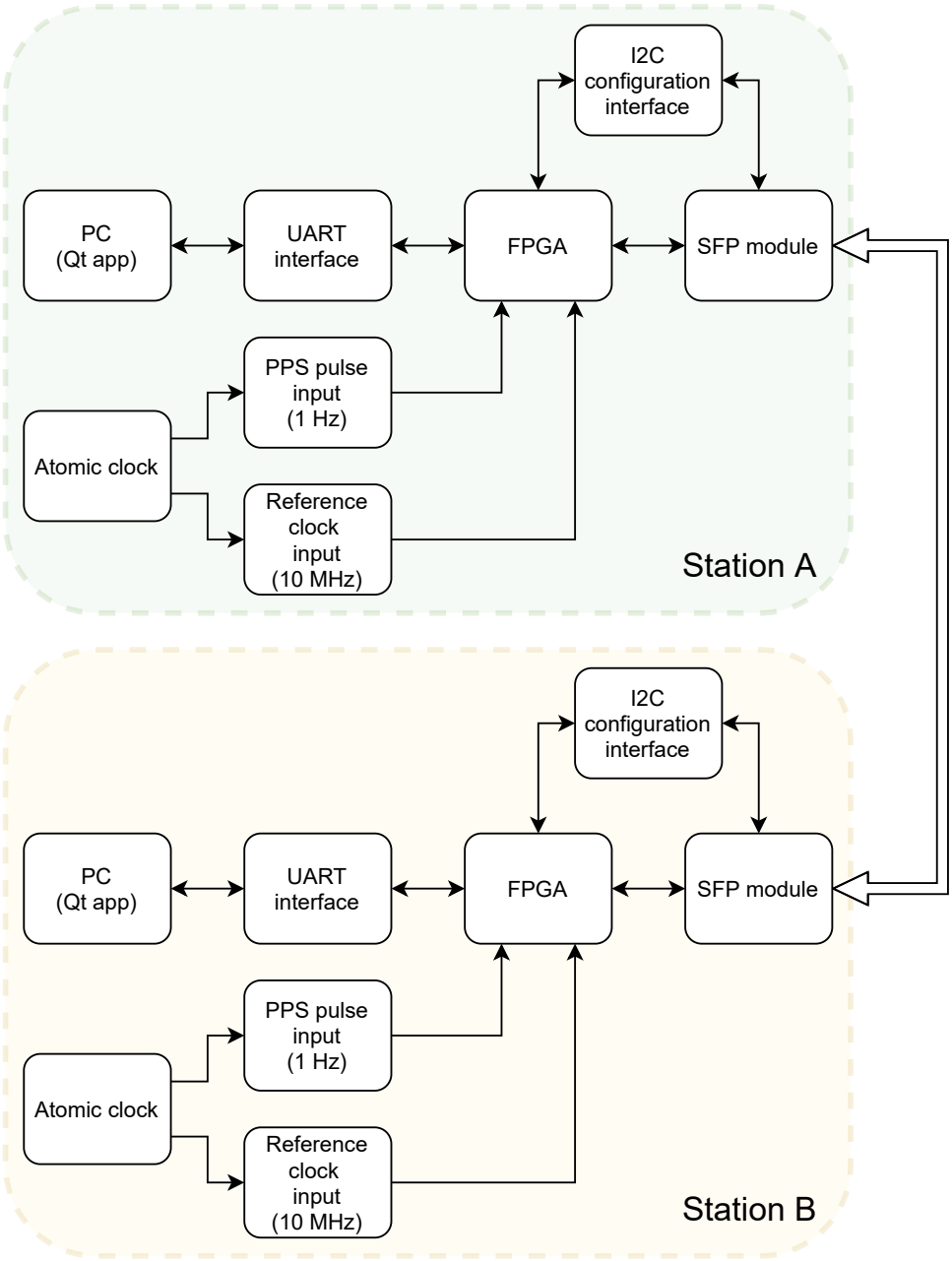


Figure 3.1: Block diagram of the complete system.

There are several particular delays in the PPS signal path that cannot be ignored during the comparison. Therefore, we rearrange the original block diagram from Figure 3.1 into the new diagram to highlight the most crucial path of the design. Each particular significant delay on the path has its label. The crucial path with the delays is shown in Figure 3.2.

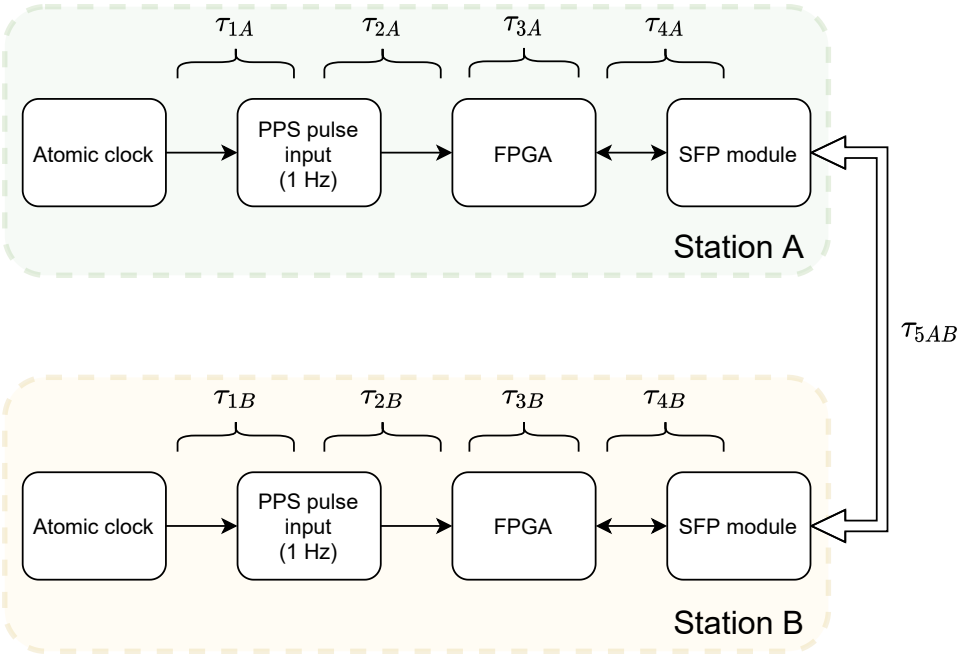


Figure 3.2: The critical path of the PPS signal in the system.

Station A is supposed to be a master and station B slave. In that case, station B task is to provide its PPS signal to the master for the final comparison. The delays τ_{1A} and τ_{1B} represent the signal delay on the path from the atomic clock to the PPS input interface. The input circuits and the routing of the signal on PCB are represented by delays τ_{2A} and τ_{2B}. Another delay, appearing on both sides of the stations, is the delay τ_{3A} and τ_{3B}. This delay is strongly dependant on FPGA design. The last delay that occurs on both stations is τ_{4A} and τ_{4B}, representing routing between the FPGA and SFP module on PCB and the delay on the SFP module itself.

If we manage constant temperature conditions, mentioned delays should stay almost constant. There is also delay τ_{5AB}, which represents a link between the stations in the form of optical fiber. We can suppose that this connection will be exposed to the environment with periodic temperature change. These temperature changes can result in signal propagation delay change in fiber optic up to tens of $\frac{\mathrm{ps}}{\mathrm{km\,K}}$ [13]. Some type of periodic compensation of these delay changes would be appropriate.

Chapter 4

Realization

For the development of the Time-to-Digital Converter on FPGA, we choose the development kit Terasic DE10-Standard. The main part of the kit is the chip Intel Cyclone V SE 5CSXFC6D6F31C6N. A little overview of the chip specification is summarized in Table 4.1.

Technology	28 nm
Logic Elements	110 000
ALM	41 509
ALM registers	166 036
DSP Blocks	112
PLLs	6
Maximum Embedded Memory	6.191 Mbit
Maximum User I/O	288
Maximum LVDS Pairs	144
Maximum NRZ Transceivers	9

Table 4.1: The FPGA chip specification [6].

The chip is separated into two parts, the FPGA and the processor ARM Cortex-A9 Dual-core. Moreover, the kit has many other periphery resources that may be helpful in the development process. We can find many switches, push-buttons, LED indicators, and 7-segment displays as a basic user input-output interface. These resources are connected directly to the FPGA. Furthermore, on the board are located other peripheries, but these parts are connected to the hardware processor. It is possible to route these periphery parts to the FPGA, but it requires a little more effort to achieve this. The kit provides many more resources that we will not describe here because we will not use them in our work. In Figure 4.1, we can see our development board, and Figure 4.2 shows the interconnection of kit peripheries.

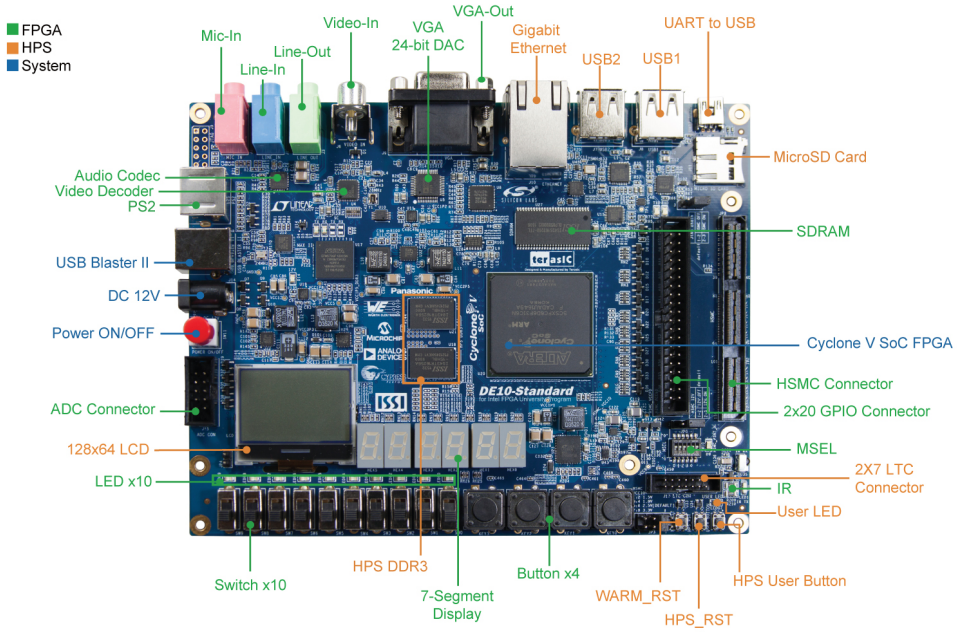


Figure 4.1: Development kit Terasic DE-10 standard [5].

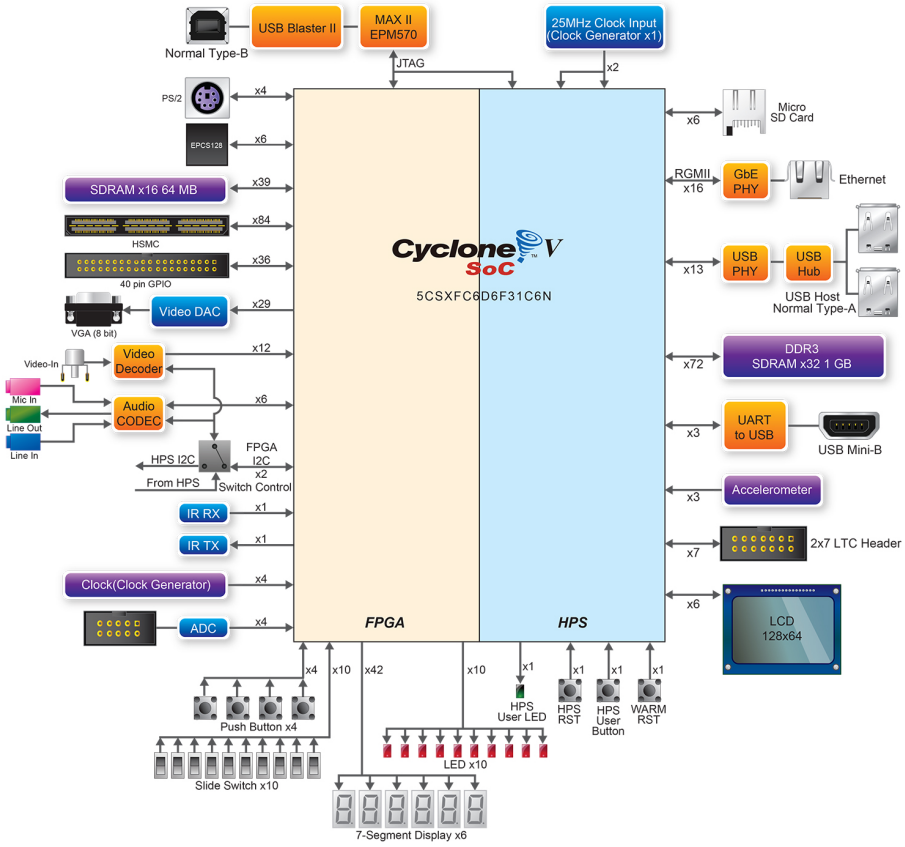


Figure 4.2: Peripheries of the development kit Terasic DE-10 standard [5].

There is an assumption that the compared stations will be connected by optical fiber. Here comes a demand for the optical connection interface on our development board. Beyond this, we need to connect the local PPS pulse from to the device. Unfortunately, there is no suitable way, how to bring these signals to our development board. These requirements encourage us to develop an extension board that provides required interfaces. The kit has a 160-pins HSMC connector for a high-speed signal, which we can use as our extension board's entry point.

4.1 Extension board

The created extension board has to ensure all needed operations, which allows bringing crucial signals to the FPGA. The requirements ensured by the extension board are listed below.

- Connection to PC using USB through UART interface.
- Connection SFP module to establish communication through an optical fiber.
- Input for PPS pulse and reference clock of the time scale.
- User-defined output with the high-frequency specification. It can be used as a debug output.
- LEDs indication interface of the state.

The intended logical connections of the developed extension board are shown in Figure 4.3.

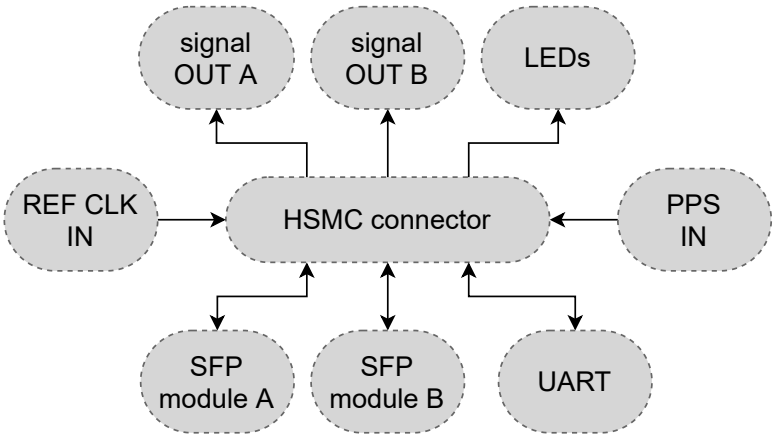


Figure 4.3: Block diagram of the extension board.

4.1.1 Input signals

The time scale provides a PPS signal and a reference clock with frequency 10 MHz. The PPS signal is compatible with classical TTL logic. The reference clock is in the form of a sine wave which is not compatible with FPGA's digital logic. Therefore, there is a need to convert a sine wave into a square wave compatible with FPGA input logic. For this purpose, we used a circuit from Microchip PL133-37 [14].

It is a circuit classified as a fanout buffer. We used it mainly for conversion from a sine wave to a square wave with promised low phase noise and low additive jitter on the output. The PL133-37 has 3 LVCMOS inverted outputs with 12 mA output drive strength, which should help for better rise time edge on each logic level change.

Furthermore, in the path of the described reference clock signal is added as a driver for conversion from LVCMOS to LVDS interface. The circuit DSLVDS1001 from Texas Instruments manufacture [15] has the task to perform the conversion from a single-ended clock line to the differential clock line. Thanks to differential signaling, the connection should be more reliable and protected from influences in the form of external noise. Even in the opposite way, this can be said that the signal path influences its surrounding components much less thanks to differential signaling. The reference clock path's schematic connection on our extension board can be seen in Figure 4.4.

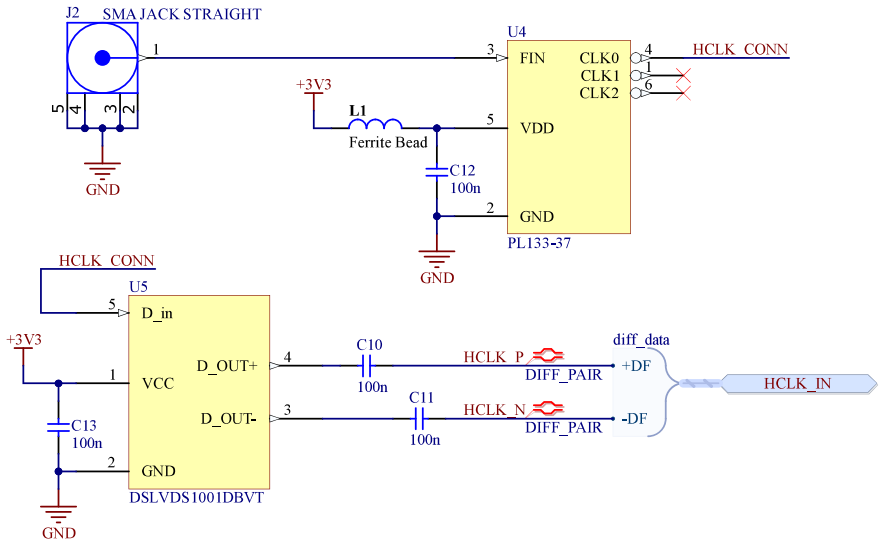


Figure 4.4: Reference clock connection.

The circuit 5PB1102PGGK8 from Renesas provides a PPS signal input interface [16]. There is no problem with logic-level compatibility. The circuit has been added only as a clock buffer to improve the rise and fall time of the PPS signal. It has ultra-low additive jitter and very low pin-to-pin clock skew. As we can see in Figure 4.5, The circuit's connection requires only one external component, which is a blocking capacitor for the power supply.

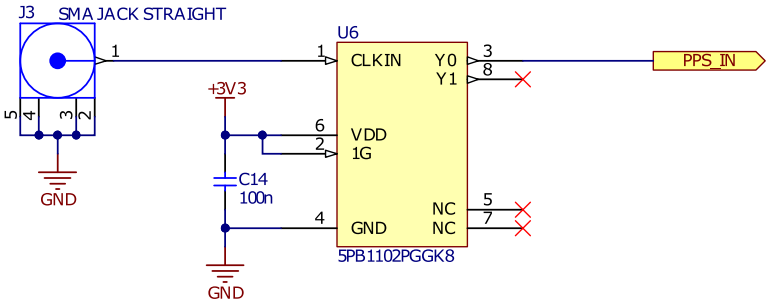


Figure 4.5: PPS signal connection.

4.1.2 Output signals

This part of the hardware design is not too crucial for the measurement itself. However, it can be handy in the sense of debugging the VHDL design. The purpose of this design part is not dedicated to some specific function. It can be used, for example, as an output for the delay generator, which may be synthesized on FPGA.

Hardware resources used for this part is again circuit Renesas 5PB1102PGGK8. It improves the steepness of the logical change on its output. A simple connection can be seen in Figure 4.6.

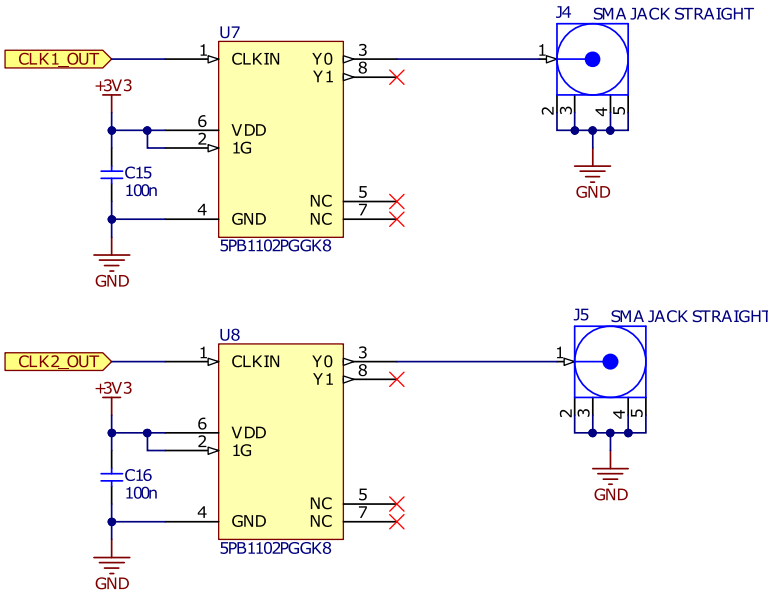


Figure 4.6: Output clock signal connection.

4.1.3 SFP modules

SFP modules are the most important part of the PCB design. It provides a connection to a second device located far away from the first one. An optical fiber establishes this connection. Therefore, it is needed to convert an optical signal to an electrical signal and vice versa. This function will handle an encapsulated device known as the SFP module.

SFP module is a compact hot-pluggable device that provides a communication interface. There are the types of SFP modules that ensure the conversion of the electrical and optical domain. We can find multiple kinds of modules with different features, for example, communication speed [17]. In our design, we have to prepare a suitable interface for the SFP module. The connection is inspired by the recommendation in SFP specifications document [18], and it is shown in Figure 4.7.

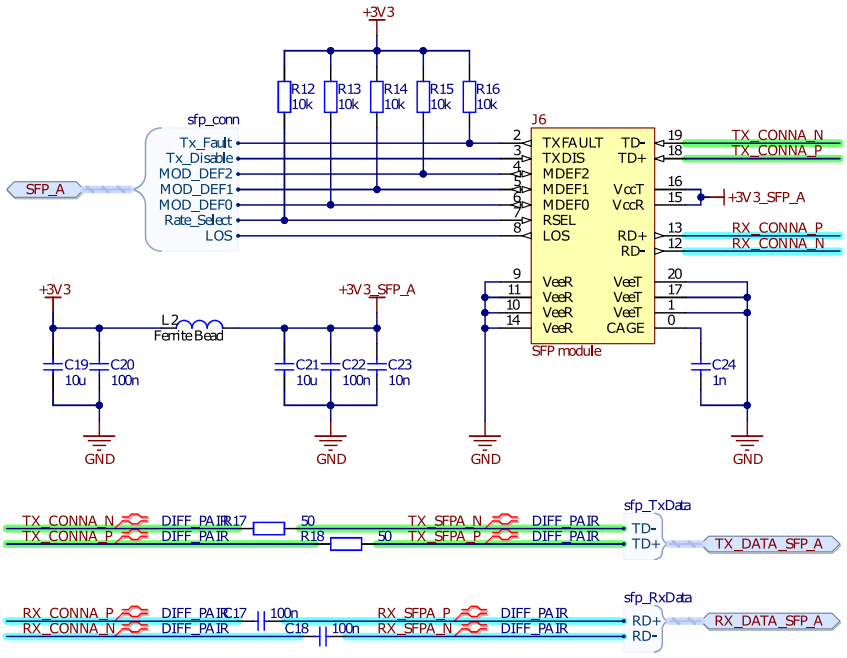


Figure 4.7: SFP module connection.

4.1.4 UART interface

The UART interface is one of the essential things that ensures communication with PC. The development kit itself has a USB to UART converter but, unfortunately, is connected to the HPS part of the chip. There was an attempt to connect the HPS UART periphery to the FPGA part of the chip, but after all, it turned out that it is much more time-efficient to add another UART to USB converter just for the FPGA. The true nature of this work is in the development Time-to-Digital Converter and comparison of two time scales. Therefore, there is no need to spend a considerable amount of time

with not related topic. The described problem resulted in the addition of the USB to UART converter to our developed extension board.

For this purpose is used chip FTDI FT230XS-R [19]. The chip itself requires a few external components for a fully functional state. As an input data connector, we choose USB-C for its reliability, robustness, and nowadays availability. The ongoing communication is indicated by two LEDs, each for one-way data transfer. The whole connection of the UART converter system is shown in Figure 4.8.

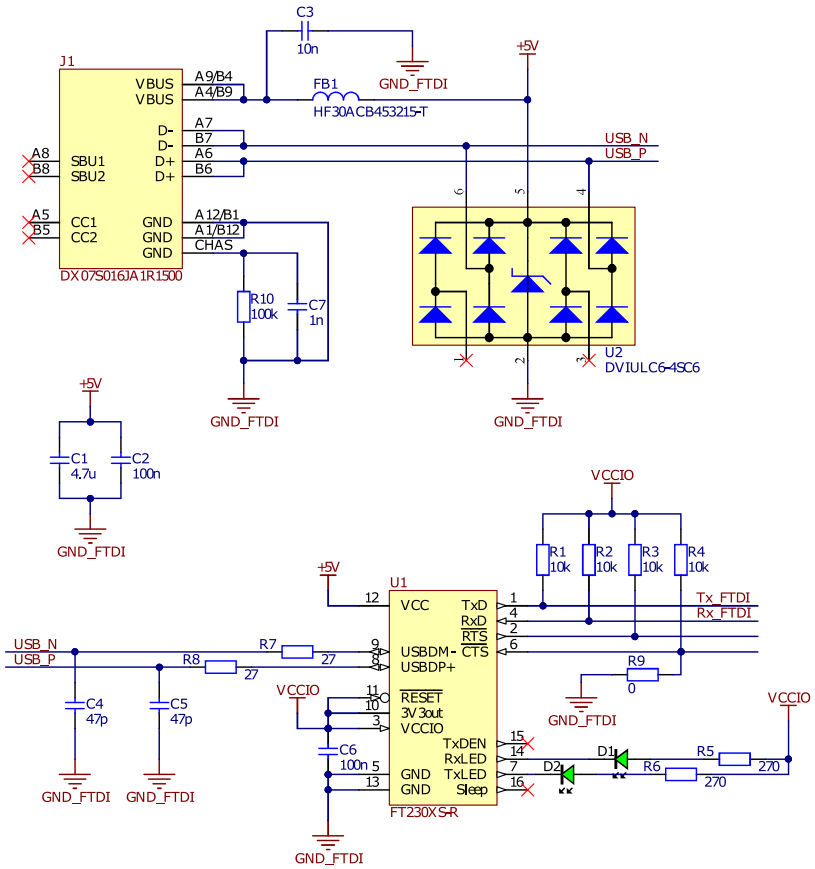


Figure 4.8: UART interface connection.

Between the UART converter and the rest of the design is added a digital isolator. For this purposes, we used the circuit from Analog Devices, ADUM226N0WBRIZ [20]. An integrated circuit does not need any additional components except blocking capacitors on the power supply. The inductive coupling is here used to establish the separation in the communication channel. The manufacturer promises that thanks to the spread spectrum of the modulated carrier, the noise emission from the coupler should be minimal. This part of the simple design can be seen in Figure 4.9.

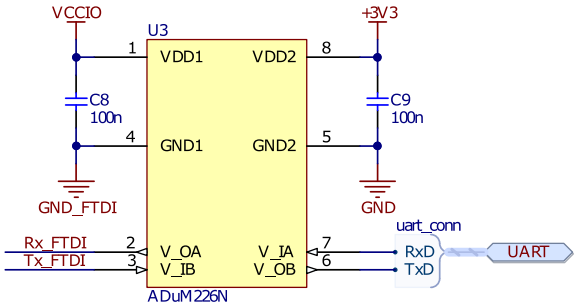


Figure 4.9: UART digital isolator connection.

4.1.5 Indications

The small part of the design is a set of LEDs which are providing status information. On PCB, we can find LED labelled as “USB”, which indicates the present power supply of the UART converter coming from the host PC. PCB label “BOARD” is associated with LED indicating the power supply granted by the development board through the HSMC connector. The other three LEDs are user-defined. The state of these LEDs can be set by FPGA. The Connection of the described LEDs is pictured in Figure 4.10.

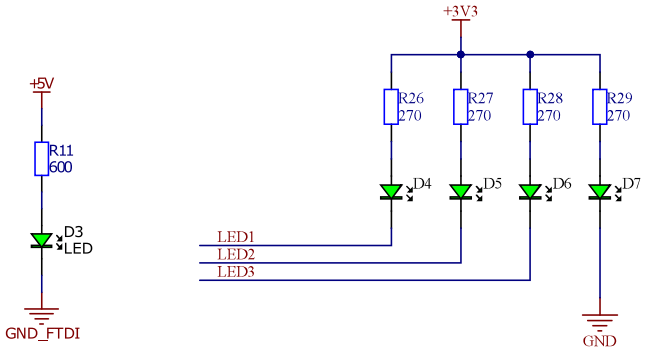


Figure 4.10: LEDs indication connection.

4.1.6 Final state

The PCB uses four-layer technology. The top and bottom layers are used for signal connection. The second and third middle layers are used for power distribution on the whole PCB design. It worth mentioning the different way of PCB routing in the inner and outer layers. Inner layers are routed inversely to the outer layers. In these layers are supposed copper planes covering the whole PCB area. By the designer is defined only separation line of these planes. This approach gives us excellent conductance for power distribution in any place of the PCB. The final form of the PCB design is captured in Figures 4.11 and 4.12.

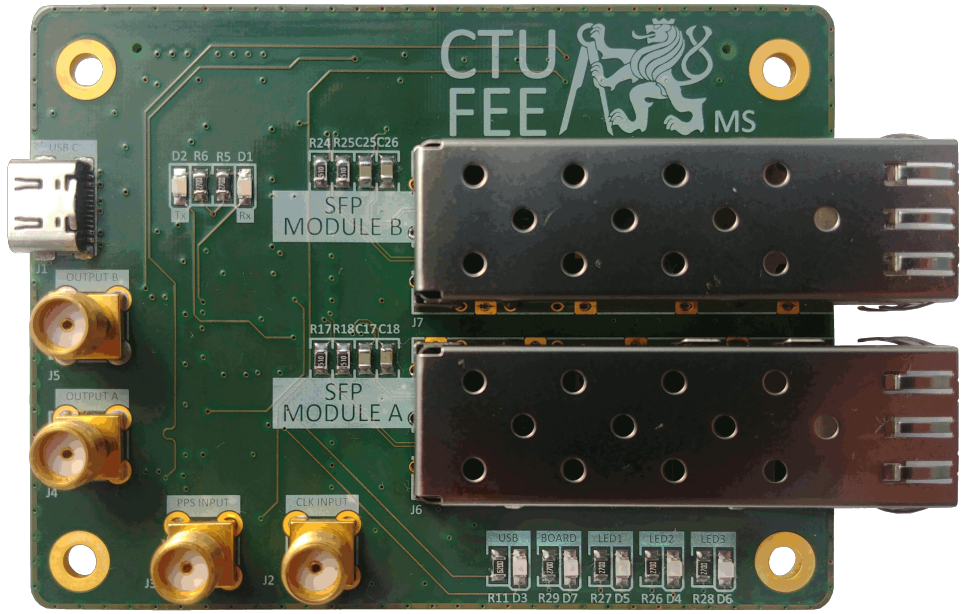


Figure 4.11: Top view on final PCB design.

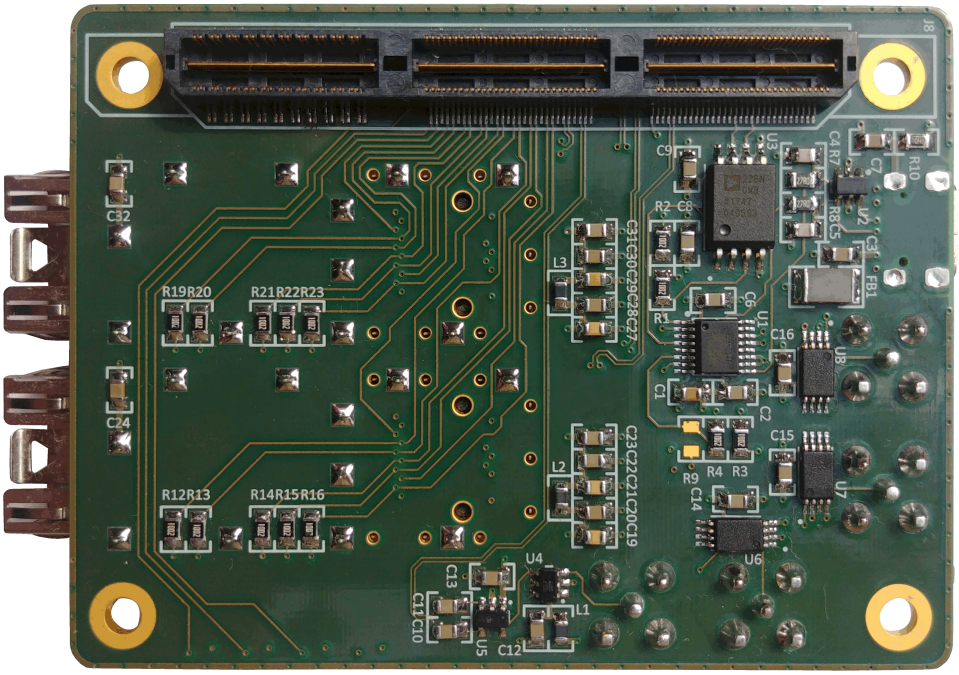


Figure 4.12: Bottom view on final PCB design.

4.2 VHDL development

VHDL language belongs to the family HDL (Hardware Description Language). It is a technology and vendor-independent language describing the behaviour or structure of an electronics circuit. Thanks to VHDL, we can synthesise source codes into the hardware structures on FPGA, which fulfils the desired functionality. The VHDL can also be used for the simulation of a circuit's design [21]. The used FPGA chip was programmed in the Intel Quartus Prime software [22], which the manufacturer of the FPGA recommends as a development environment.

4.2.1 TDC design

The main section of this VHDL design is the Time-to-Digital Converter itself. We decided to implement the form using a delay line and counter. Counter for coarse measurement and delay line for fine measurement. The principal function of this type of TDC structure is described in Section 2.5.

Delay line

It shows that the best hardware resource that we can use on our Cyclone V FPGA for delay line purposes is an adder. Based on the documentation, the adder's chained carry-in and carry-out signals provide the most stable and the smallest delay of the ongoing signal. One of the adder's modes is arithmetic mode - carry chain [23]. Figure 4.13 describes a little more the used hardware for delay line purposes.

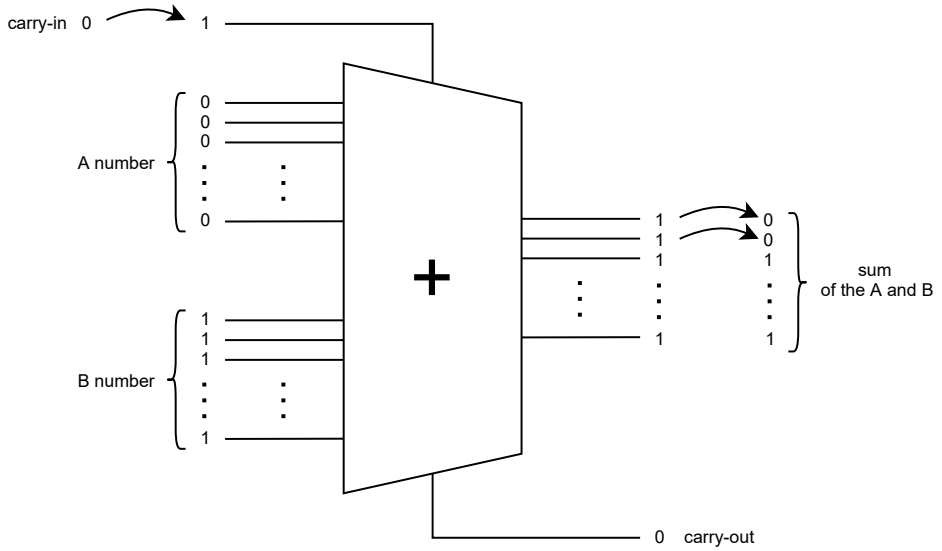


Figure 4.13: The idea of the adder delay line.

The carry-in input is used as a starting point for changing the adder output bit by bit. One of the adder's input has to be filled by zeros, and the second input by ones to obtain the described behavioural. The inputs A and B are constant throughout the adder operation. The initial state of the adder carry-in input is low logic level. The inputs state of the adder forces the output state to all ones. Described input combination can be marked as a default state of the delay line. The structure is prepared for the incoming trigger signal, which starts propagating the ongoing signal through the adder. Trigger signal routed to carry-in input will change the low state to the high. The performed change will cause a successive change of the adder output bits. Bit by bit will change its high state to a low state.

The internal structure of the adder in delay line mode consists of one-bit full adders, as shown in Figure 4.14. Except the last adder is each carry-out output of the adder connected to the next adder carry-in input. Delay of the carry signal between each adder is an essential property of this structure.

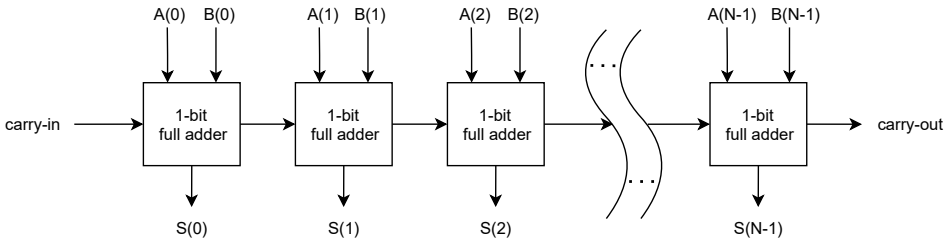


Figure 4.14: Chained one-bit full adders.

For completeness, the internal structure of the 1-bit full adder, used inside the adder structure, is shown in Figure 4.15.

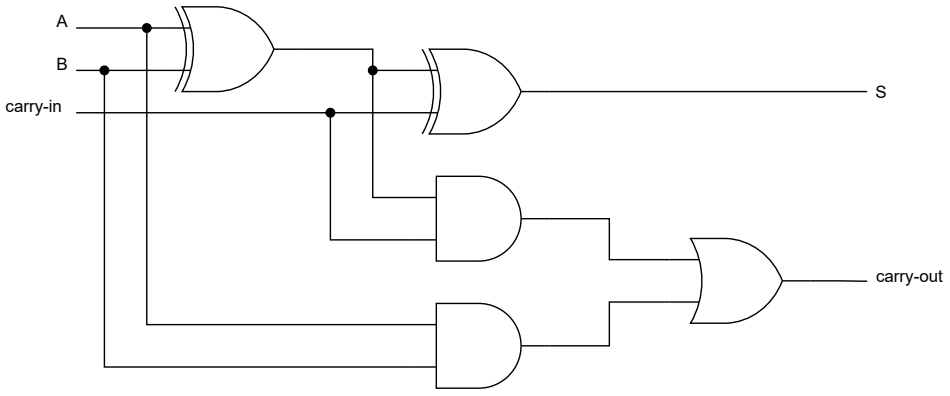


Figure 4.15: One-bit full adder connection.

In practice, we struggled with the synthesis of described structure in Quartus. The constant values assignment to adder inputs caused problems. The described use of the adder does not make sense in the case of regular design. During the optimization stage, the Quartus evaluates the structure as useless and removes it from the final design. Occurred problem is solved

using memory on adder inputs instead of direct constant values assignment. Memory content can be any set of numbers and may be changed. Therefore, Quartus can not discard the adder from the design because the content of the memory can be changed during the operation. Afterwards, the memory content has to be set to comply with the initial state of the adder.

To complete the delay line design, we have to connect the output bits of the described adder delay line with the D flip-flop chain. D flip-flop chain should store the delay line's state at the rising edge of the stop signal. The block of the design can be seen in Figure 4.16. To achieve the best performance, we had to enclose the delay line design into the so-called logic lock region. Thanks to the logic lock region directive, we can lock the design on the desired location on the chip FPGA with a defined size. Without the logic lock region, Quartus spread the design across chip resources, which leads to malfunction of the design.

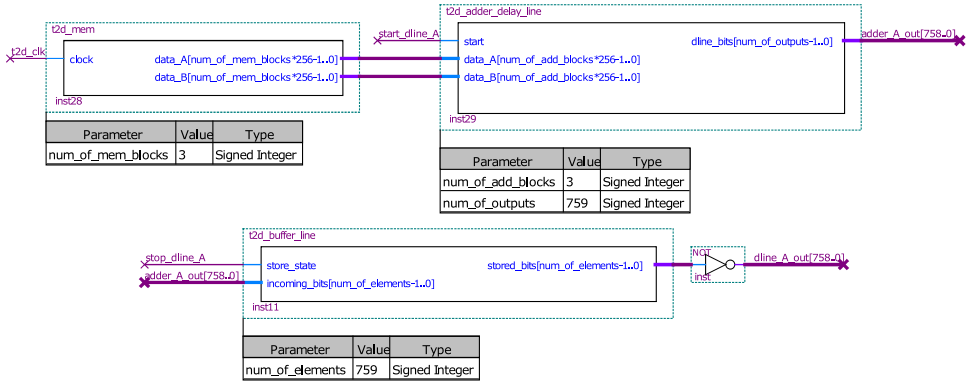


Figure 4.16: Design of the delay line in Quartus.

Encoder

The encoder is a part of the design, which is tightly connected with the delay line implementation. It converts the raw representation of the delay in thermometer code to the classical binary representation of the number. The encoder evaluates the delay line state after the arrival of the stop signal and finds its output position of the logic level transition. This position represents the number of delay elements altered by the ongoing start signal until the stop signal arrives. Unfortunately, error bubbles, which represents the inconsistency in thermometer code, may occur. Therefore, a prior encode was used instead of a basic encoder, which encodes the highest position of the logic level transition on its output. This approach should eliminate the bubble error issue. The created prior encoder is shown in Figure 4.17.

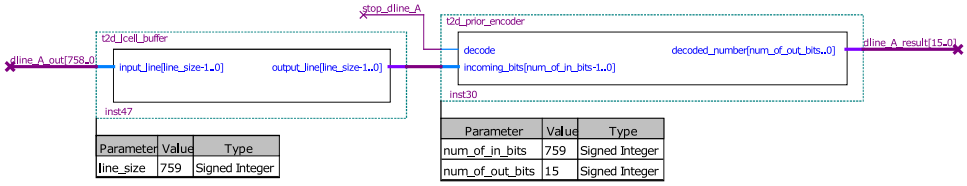


Figure 4.17: Prior encoder design in Quartus.

Counter

The counter design is responsible for counting the number of passed clock edges. Counter's state is incremented by one on the incoming clock signal's rising edge in enabled mode. The control logic defines the mode of the counter. At the end of the enable mode, the state of the counter is stored in the buffer. Then the counter is reset to zero initial state as preparation for a new measurement. The maximal measurement time range is defined by the width of the digital number representing the counter's value and the input clock frequency. In our case, the data width is 16 bits. Theoretically, this width should provide a measurement window defined by (4.1). We suppose $T_{clk} = 4\,\text{ns}$, which should give us 262.14 µs window size.

$$T_{\text{MAX}} = (2^n - 1) \cdot T_{\text{clk}}, \tag{4.1}$$

where n represents a number of data bits, and T_{clk} is a clock signal period. Quartus design can be found in Figure 4.18.

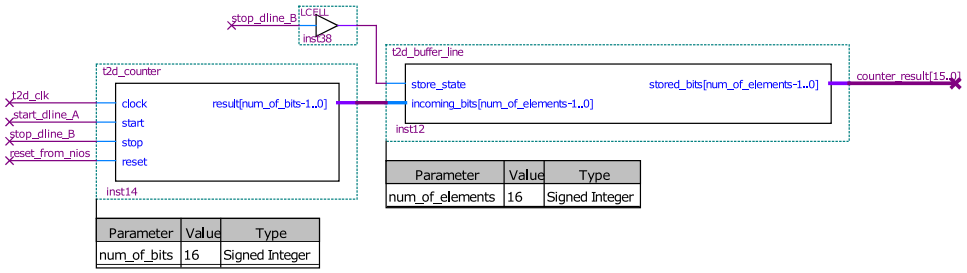


Figure 4.18: Counter design in Quartus.

Control logic

The crucial part of the TDC design is control logic. It ensures proper triggering of each delay line by start and stop signal and enable or disable counter. The block diagram in Figure 2.9 depicts the role of this design. The control logic in Quartus shows Figure 4.19.

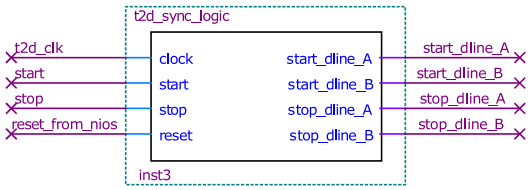


Figure 4.19: Control logic in Quartus.

Ensure proper function is a little tricky task. We faced the problem that the registered sequence was not the same as the incoming sequence of the input signals. As a problematic situation, we evaluated the state when the clock's edge and stop signal came close to each other. This particular state is typically manifested by a periodically varying number of registered clock pulses when measuring a constant delay. Synchronization is constructed by three concurrent design using the VHDL process. The first two are changing their state to a high level synchronously with a start and stop signal. The third design is sensitive to the rising edge of the clock signal. If the clock signal's rising edge is present and the delay line is activated by the first or second design, a signal for the stop appropriate delay line is generated. All these mentioned designs are equipped with asynchronous reset input. The timing diagram of the signals in Figure 4.20 provides a closer look into the function of the synchronization logic. Each action of the logic is labelled in ascendant order.

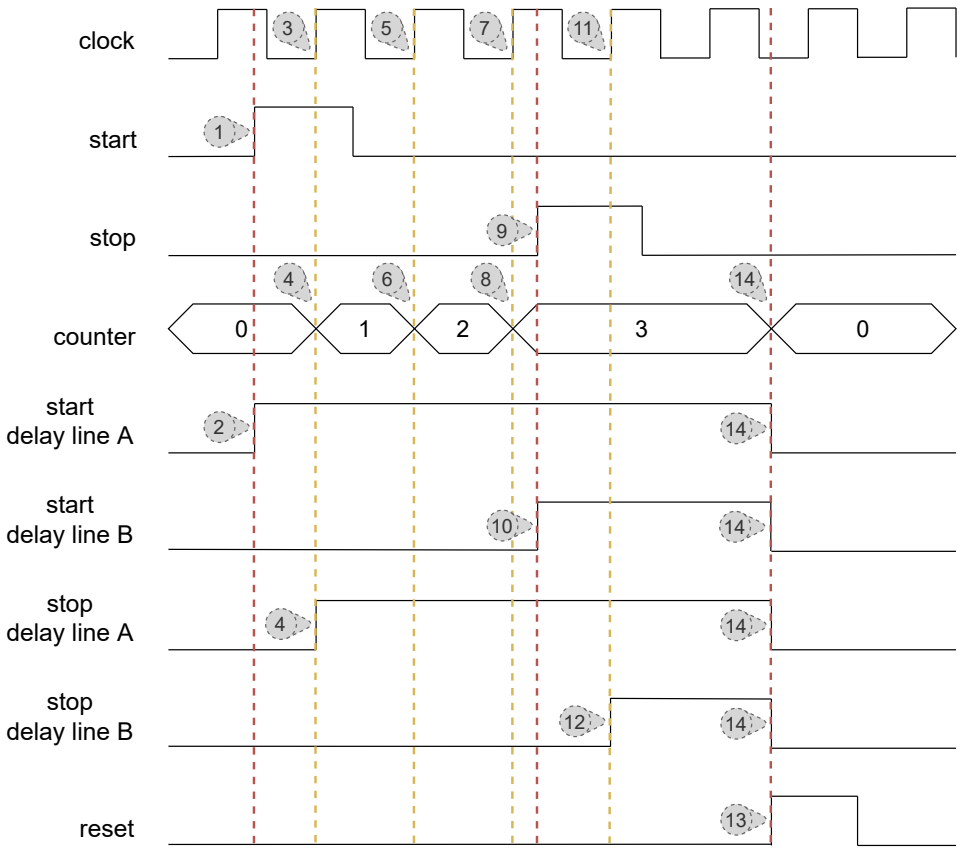


Figure 4.20: A timing diagram of the control logic.

Normalizer

The order of the rising edges of the PPS signal from the local and remote time scale is not guaranteed. Therefore, a normalizing design is needed. Normalization ensures that on start input of the TDC will always be rising edge earlier than on stop input. Without described normalization, measurement probably leads to the overflow due to its maximal measurement range. The normalizing block from the Quartus project is shown in Figure 4.21. The Figure also shows the switching logic of the normalizer input. We can switch between PPS evaluation and delay generator, which is described in Section 4.2.2. The selected source is also provided on dedicated debugging outputs of the extension board, which was described in Section 4.1.2.

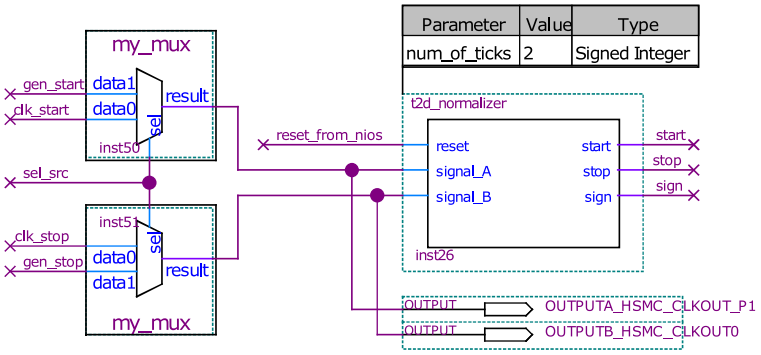


Figure 4.21: Normalization design in Quartus.

4.2.2 Delay generator

The design of the delay generator was used during the development of the TDC. At the final stage of the development, the generator can be used for calibration purposes. The delay generator can be used for the comparison of the proposed TDC and the external time meter. Its implementation follows the reverse idea of the used delay line (Section 2.1). The start signal is generated by the rising edge of the connected clock signal. The clock signal is also connected to the delay line. The second delayed signal, labelled as a stop signal, is generated by selecting the appropriate delay element of the delay line by a multiplexer. The LCELL primitive was used as the delay element of the delay line in this case. Quartus project design of the delay generator is in Figure 4.22.

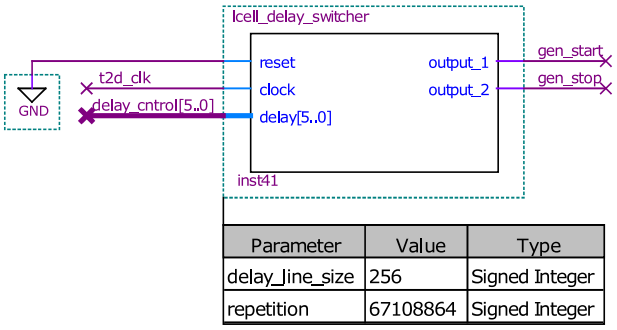


Figure 4.22: Delay generator in Quartus.

4.2.3 Modulation and demodulation of PPS signal

Transferring the PPS signal through the optical channel was another issue that we were facing. PPS signal is characterized by a period of one second and by its highly unproportional duty cycle. The high state is typically present for several tens of microseconds. SFP modules are designed for very high-speed communication. Therefore this relatively low-speed PPS signal is not suitable for sending in its pure form. Because of the mentioned problem,

the PRBS (Pseudo-Random Binary Sequence) modulation was introduced. It is crucial for reliable signal transfer.

The PRBS is a binary sequence that seems to be randomly generated but actually is generated deterministically [24]. Its statistical properties are close to the truly random sequence. The Pseudo-Random Binary Sequence generator is characterized by its n-degree polynomial. One of the polynomial dependent property is the repetition period. The sequence is repeated after N bits, where N is defined by (4.2). MLS type (Maximum-Length Sequence) of the PRBS should provide duty cycle within one sequence period $c = \frac{1}{2}$. The duty cycle c is defined by (4.3). This means that in one period of the PRBS should be 2^{n-1} ones and $2^{n-1} - 1$ zeros [25].

$$N = 2^n - 1, \tag{4.2}$$

where n represents the degree of the characteristic polynomial of the PRBS generator.

$$c = \frac{m - 1}{N - 1}, \tag{4.3}$$

where m is a number of ones in the sequence of the length N.

The structure which encodes input data using a PRBS generator is called a scrambler. Sent data are encoded at the transceiver side according to the applied polynomial in the scrambler structure. With knowing scrambler polynomial, the data are decoded into the original state by descrambler at the receiver. There are several reasons why this approach is suitable for data communication. At first, it breaks down the long sequence of the constant logic level. Unchanging logic level for a relatively long time on the communication channel may cause issues. The second reason is the improved power spectrum of the communication channel. Scrambler provides a spread spectrum in the communication channel, which reduces power spectral density. For example, thanks to the spread spectrum, communication is more difficult to disturb. There are two types of scrambler additive and multiplicative [26].

The multiplicative version of the scrambler was used in our case. The connection of the multiplicative scrambler and descrambler structure is shown in Figure 4.23 and Figure 4.24. Design follows polynomial PRBS15 = $x^{15} + x^{14} + 1$. Scrambler and descrambler structure is designed using a shift register and XOR function.

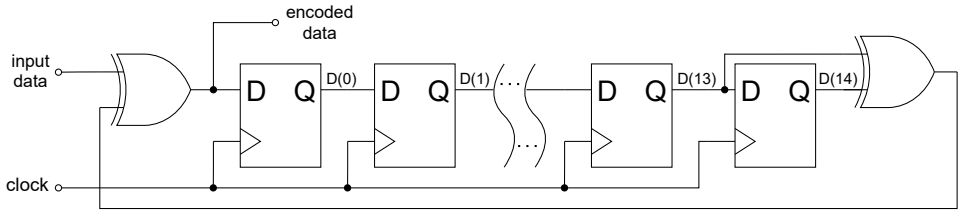


Figure 4.23: Scrambler connection.

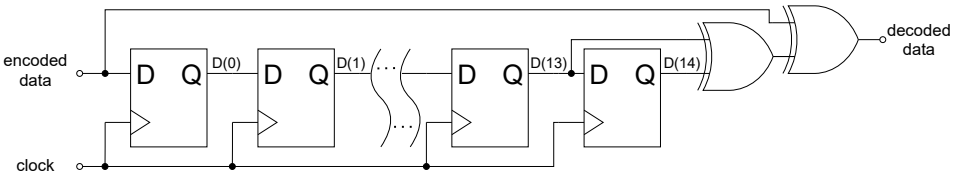


Figure 4.24: Descrambler connection.

Ports labelled, in Figure 4.23 and Figure 4.24, as encoded data are in the final design connected by fiber optics. The final design of the channel for the PPS signal in Quartus can be seen in Figure 4.25.

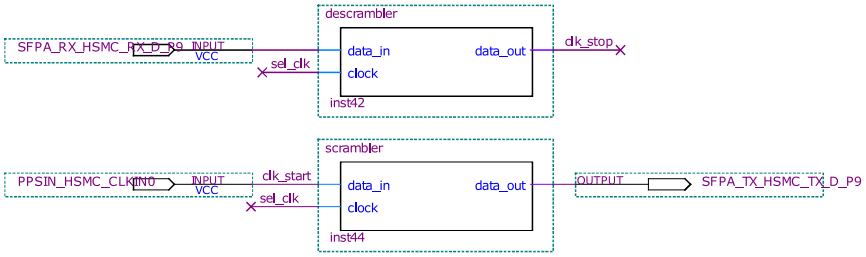


Figure 4.25: Scrambler and descrambler in Quartus.

4.2.4 Processor Nios II

The whole described design on FPGA needs to be controlled by the user from the PC. By these requirements, we have to ensure processing more complex tasks, which are specified below.

- connection with PC through UART interface
- read and write SFP module memory through the I^2C bus interface
- read the output of the TDC
- provide reset pulse for measurement chain
- control delay generator

The processor is more than suitable for this type of tasks. Therefore, we deployed the processor Nios II [7] in our project. The Nios II is a soft-core processor, which means that its structure is described by HDL code. The architecture of the processor is RISC. The main advantage of this approach is the large variability of the processor. A developer can configure a processor to fit exactly the projects needs. All configuration of the Nios II processor is done by Quartus build-in tool named Platform Designer.

Platform Designer offers three different versions of the processor. Versions are labelled as *e* (economy), *s* (standard) and *f* (fast). The performance and properties of the processor also increase in the given order. An overview of the Nios II version properties can be found in Table 4.2. Our design uses version *f*.

Nios II type	*e*	*s*	*f*
Processor pipeline	1 stage	5 stage	6 stage
Branch prediction	-	static	dynamic
Multiplication	software	3-cycle multiplier	1-cycle multiplier
Shift	software	3-cycle barrel shifter	3-cycle barrel shifter
Size [LEs]	540	1030	1600
Maximal clock rate [MHz]	195	110	140
Performance [MIPS]	18	55	145

Table 4.2: Nios II version properties [7].

Platform Designer is a tool with a semi-graphical configuration interface, where various designs can be built. Prepared forms provide the configuration of each component. The user-defined junction establishes the connection between the components via a dedicated bus. The HDL code is auto-generated by the Platform Designer. An example of the Platform Designer project is shown in Figure 4.26. This example also shows the configuration of our processor that is used in the Quartus project.

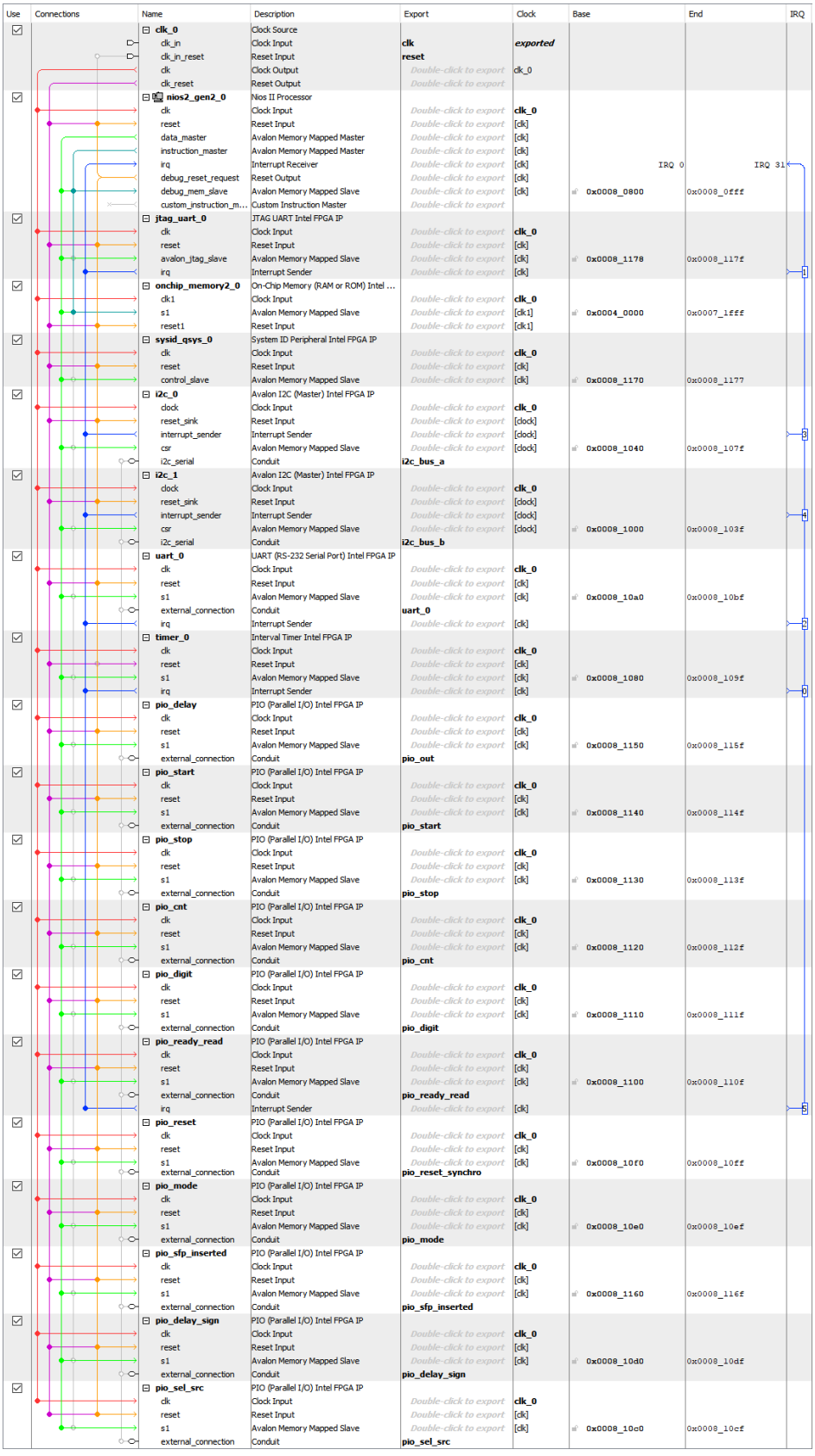


Figure 4.26: Nios II configuration in Platform Designer.

Generated processor, specified by the Platform designer project in Figure 4.26, is shown in Figure 4.27. The figure also shows the buffers for the I^2C interface.

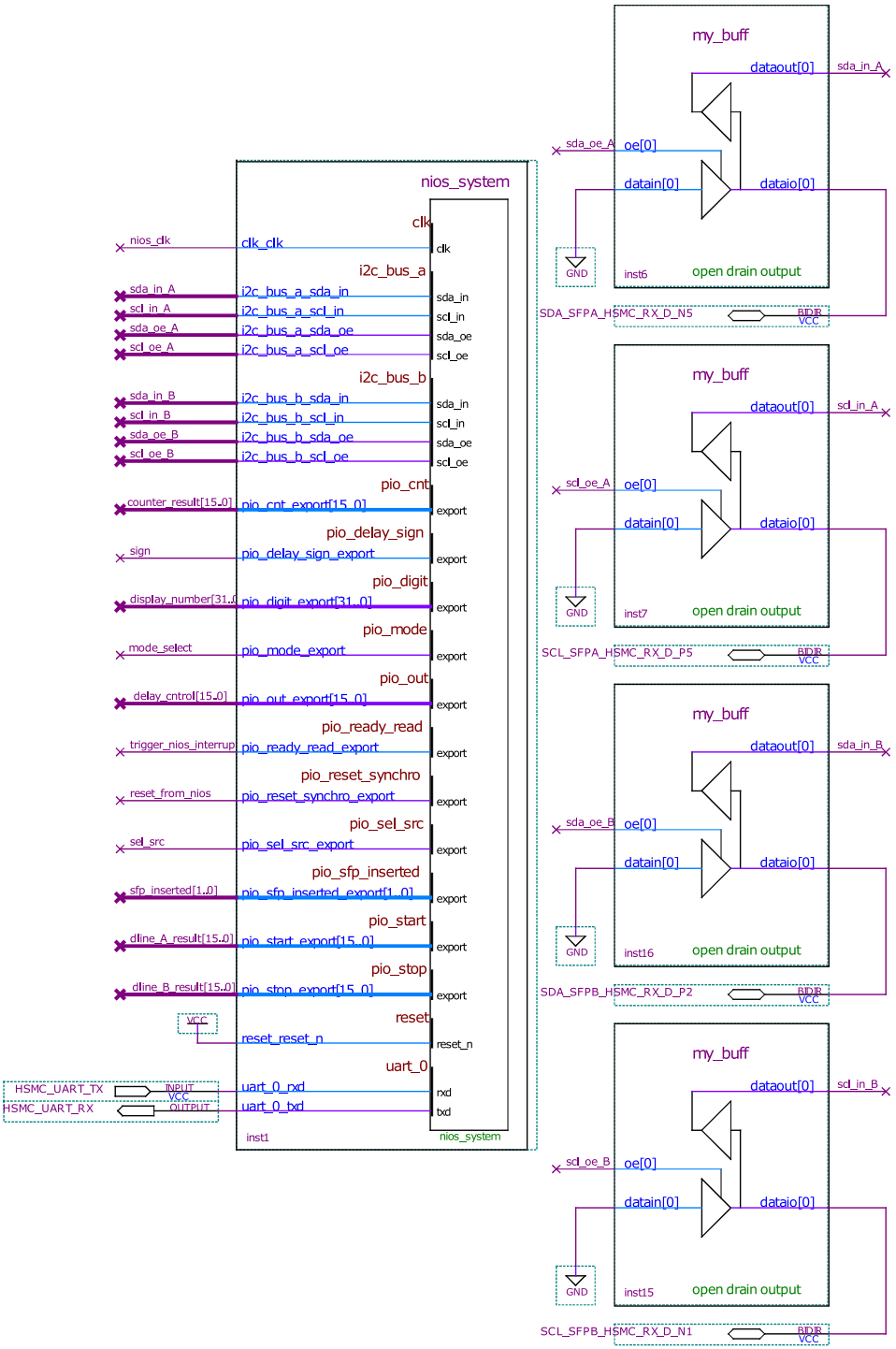


Figure 4.27: Nios II in Quartus.

4.3 Processor Nios II firmware

The firmware for the processor Nios II is written in C language. A key function of the processor on the FPGA is to provide a communication channel from the user at PC application to the FPGA design. Therefore is necessary to establish UART communication with PC. The UART operates in a non-blocking mode which allows us to do some other necessities. UART communication is operating with the following settings.

- baud rate 115200 Bd
- frame data size 8 bit
- no parity
- one stop bit

There is a need to send and receive multiple types of messages between PC application and firmware. This requirement leads to the creation of our communication protocol. The frame of the protocol is shown in Figure 4.28.

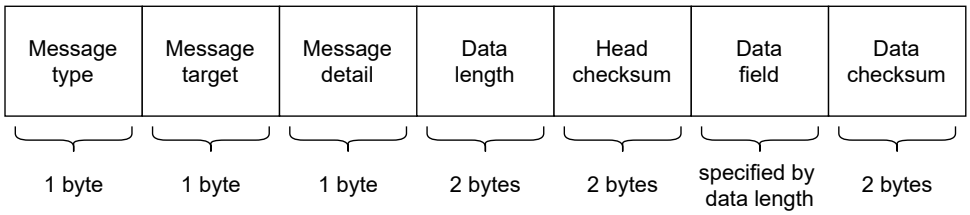


Figure 4.28: Message frame of the communication protocol.

The message protocol has three specification blocks in the head of the message frame, precisely determining the message purpose. The following listing shortly explains the purpose of each block of the message frame.

- **Message type** - determines the primary purpose of the message
- **Message target** - determines to what part of the design is a message dedicated
- **Message detail** - additional specification related to the previous message target block
- **Data length** - specify the number of bytes in the data field
- **Head checksum** - checksum of the previous four blocks of the message frame (head of the message frame)
- **Data field** - raw data
- **Data checksum** - checksum of the message frame

Sometimes the data frame may be damaged. It is good to know if that situation comes and make an appropriate response to that situation. Therefore we add to the message frame two blocks of the 2-bytes checksum. It is an instrument for detecting the corrupted messages to prevent unwanted behaviour caused by them.

Another periphery that we have to configure and make accessible is the I^2C bus for SFP modules configuration. Based on the reference manual, the SFP modules can handle clock frequency up to 100 kHz [18]. An appropriate initial configuration of the I^2C interface has been done.

In the default state, the processor only waiting for UART commands and generates reset pulses for FPGA control logic after reading the value of measured delay. Delays are not locally stored. By changing this default mode to the elevated mode, we force the processor to send delay data to the PC application without requesting each sample.

4.4 User application

All control over the FPGA design is done using a PC application. The application is written in Qt framework [27] using C++ language. Qt is a cross-platform framework for application development. The layout of the main application tab is shown in Figure 4.29.

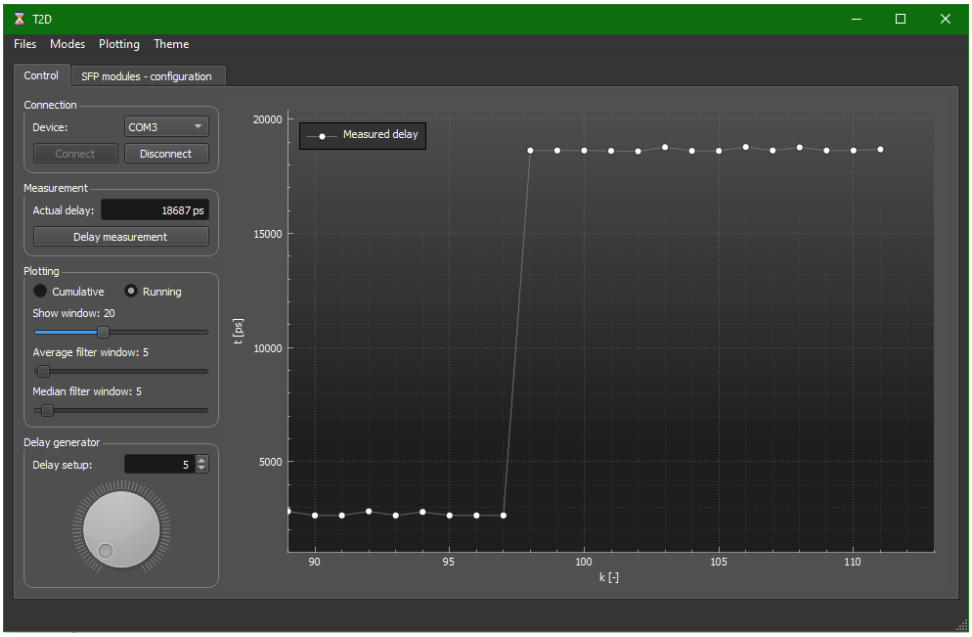


Figure 4.29: Control tab of the user application.

4.4.1 UART communication

The Control field for serial connection is located on the "Control" tab of the application. It is a default tab that shows up at the startup of the application. For connection to the extension board, the user has to specify the COM port, where the extension board is connected to the PC and then click on the button "Connect". There is also the "Disconnect" button to leave a UART session.

4.4.2 Time measurement

The time measurement can be started after the successful connection to the extension board. At First, the user should specify the mode of the device. In the "Modes" bar menu can be chosen master or slave role of the device. If the user select master mode, the second device has to select the slave mode and vice versa. After that, the device is ready for measurement. Button "Delay measurement" will request the measurement data from the TDC.

4.4.3 Data visualization

Measured delays are immediately stored and plotted in a graph after receiving. There are multiple features of data plotting. The user can choose the cumulative or running data plotting. The cumulative option visualize all stored data in the graph. On the contrary, the running option will show only the last n samples. The length of the showing window can be customized in a range from 1 to 50 by a slider labelled as "Show window". There is also features providing basic data filtering. They can be enabled in the menu bar named "Plotting". User can select an average filter and median filter. The data items range used during filtering can also be selected by the user. The last feature connected with data handling is "Clear plot" located in the plotting menu bar. Pressing it, the user deletes all measured data stored in the application.

4.4.4 Delay generator control

Delay generator design has to be enabled by the user before use. This feature is disabled by default. Only one source of the TDC input can be used at one time. Therefore, by enabling the delay generator, we disable the evaluation of the PPS signals. Two objects in the application can be used to control the delay generated by the FPGA design. First is a spin-box, where the user can write an exact number or change the value of the delay by one by little arrows. The second way is to use a rotating knob and change its value by the cursor. The value is propagated through UART and processor Nios II to the FPGA delay generator design.

4.4.5 Export of the data

The data export is crucial for further processing in the specialized software dedicated to analyzing the measured delay data. The CSV file format was chosen for its simple construction and wide acceptability across analysis software. The file is organized into two columns. The first column defines the index of the measured delay. The second column directly represents measured delay.

The data for exportation has to be already stored in the application. After click on the "Export to CSV" button in the "Files" menu bar, the prompt for specification of the location where exported file will be saved, and then hit the "Save" button. The plot can also be exported to the PDF file format in the same menu bar.

4.4.6 SFP configuration

On the second applications tab named "SFP modules - configuration", we can find a user interface for read and write operations to SFP modules registers. The tab is mainly covered by two tables prepared for storing the memory content of the SFP module. The first table represents the ID field, and the second table represents the Diagnostics field of the SFP module memory content. Parsing and decoding of the configuration follows the SFP modules reference manual [28].

By the combo box object, we can directly switch between SFP module A and B. All configuration actions are related to the selected module in the mentioned combo box. By clicking to "Read all items", we send a request to read the selected SFP module's ID and Diagnostics memory field. Received data are parsed to the prepared table cells. Content of the tables can also be deleted by the button "Clear table content".

The application also has the possibility to read and write data at a specified memory address. For these purposes is dedicated application group control labelled as "Custom configuration". The Layout of the configuration tab can be seen in Figure 4.30.

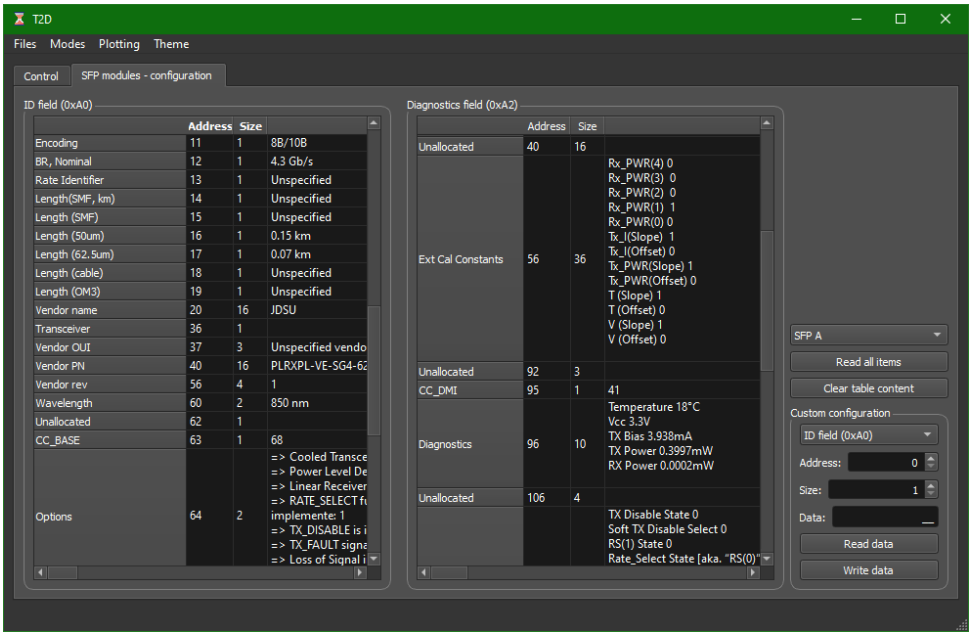


Figure 4.30: SFP configuration in user application.

4.4.7 Application customization

The application has the option to switch between the dark and light theme mode. The default theme is dark. Application appearance in light mode is shown in Figure 4.31.

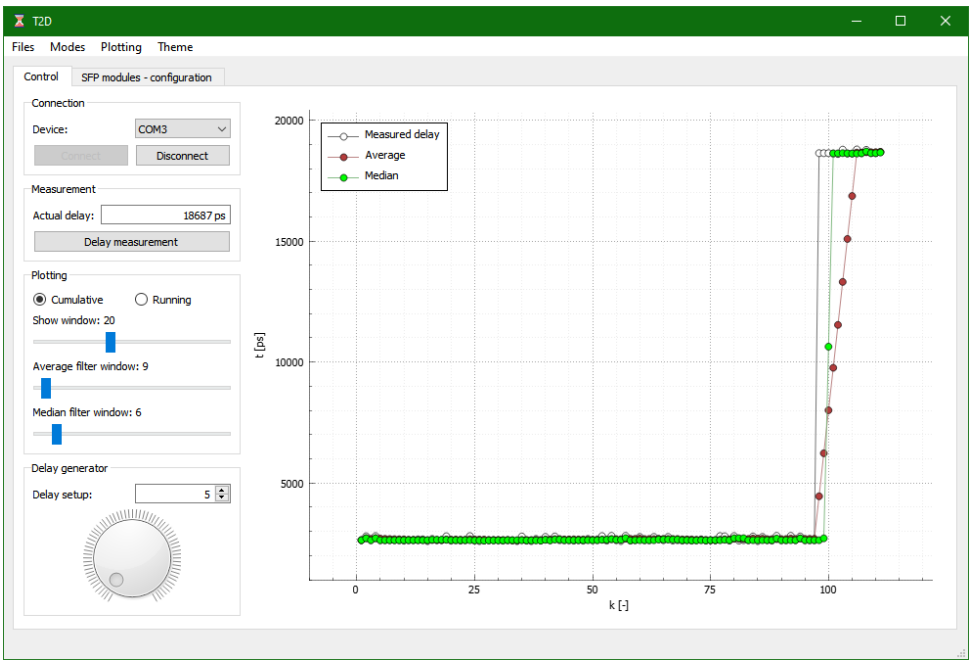


Figure 4.31: Qt application in light mode.

Chapter 5

Results

The developed resources need to be verified and proven their proper function at the last stage. We will show the results of several crucial experiments, which should provide a decent functionality illustration of the developed system.

5.1 Delay line validation

The delay line is an essential part of the final TDC design. Its adding sub-nanosecond resolution to the final measured delay. Therefore, it is good to know the properties of the implemented delay line.

To measure the transfer function of the delay line, we have to generate two signals delayed towards each other in range 0 to few nanoseconds with step about tens of picoseconds. Achieve this requirement is very challenging. The first approach was to generate these signals by the hardware resources of the FPGA. The Delay generator seems to be suitable at the beginning. After the first few attempts, we found out that we cannot generate a sufficiently small delay with an acceptable delay step. Quartus optimization and routing probably caused discovered issue.

We came with another method using a two-channel function generator. Generator Agilent 33522A provide the possibility to set phase shift between two waves in the range 0° to 360° with a resolution of 0.1° [29]. If we supposed square wave on each output channel with maximum possible frequency 30 MHz, we are able by this way set the delay between two output signals in range 0 to $\approx$ 33.3 ns with resolution $\approx$ 9.26 ps. These parameters are satisfying for our purposes. Generation of the very short variable delay was prepared. Measurement was verified together with precise time interval counter SRS SR620 [30]. Moreover, to improve the reference clock stability of the used function generator and counter, we used rubidium frequency standard SRS FS725 [31] as an external clock reference. Block diagram of the measurement chain can be seen in Figure 5.1.

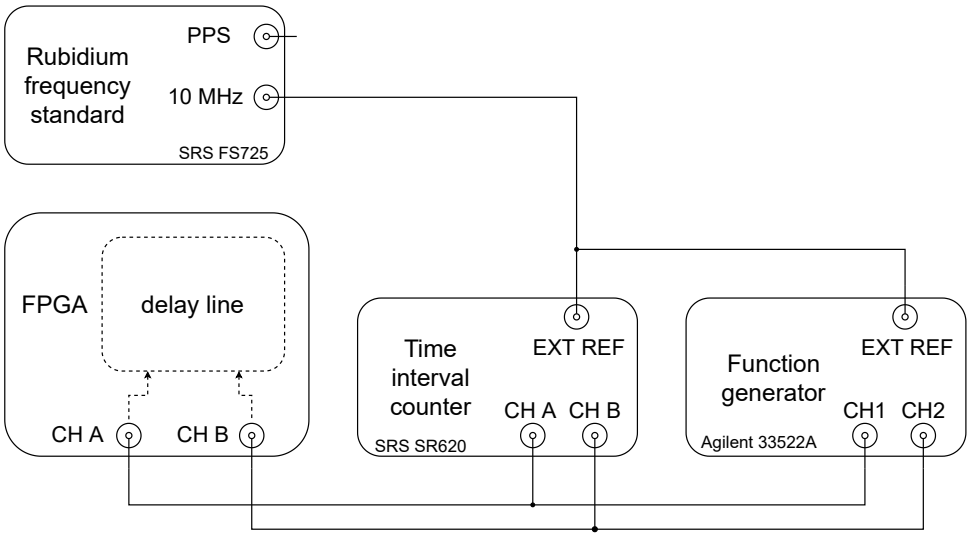


Figure 5.1: Connection of the measurement chain during the evaluation of the delay line.

The transfer function was measured in few representative points through the whole range. Each point of the transfer function was measured as a single sample without any averaging. Results of the measurements can be seen in Figure 5.2. The measured delay t by time interval counter is on the x-axis. The number of the delay elements at high state n is on the y-axis.

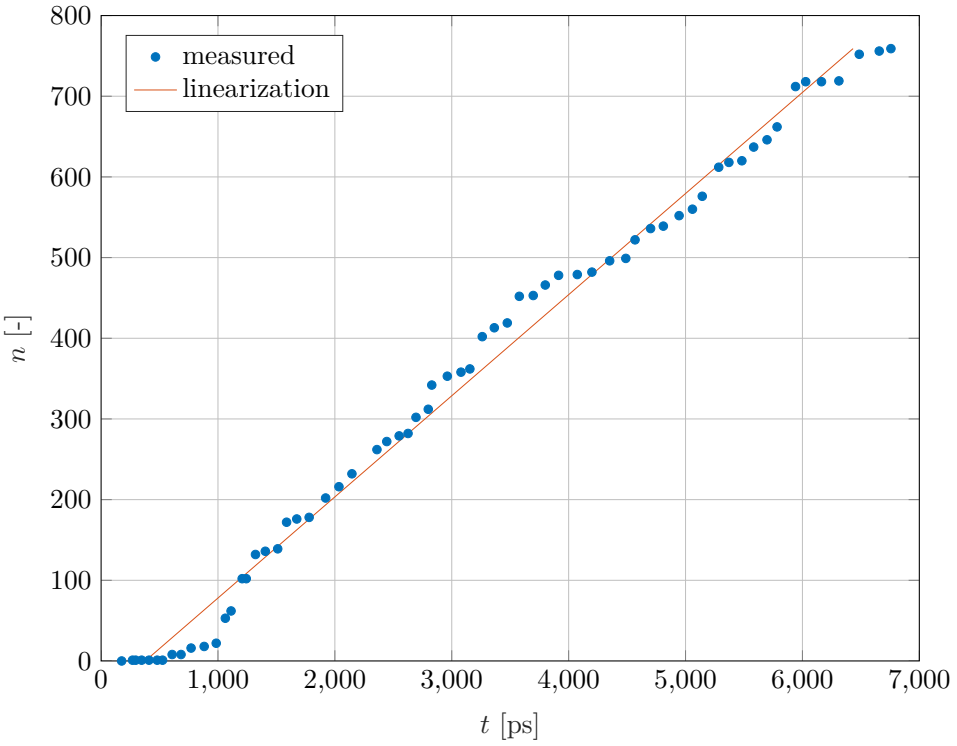


Figure 5.2: Measured transfer function of the delay line.

To reduce the negative influence of the delay fluctuation, we perform measurement over multiple samples at few fixed delays. The processed transfer function with averaged results is in Figure 5.3. The meaning of the axis is still the same as in Figure 5.2.

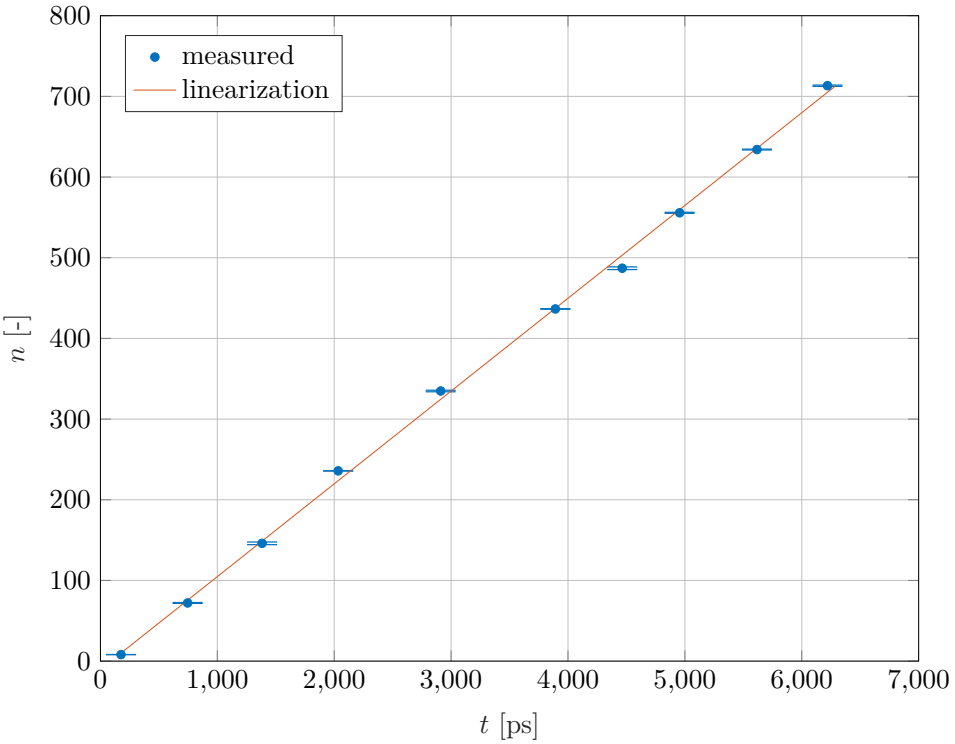


Figure 5.3: Transfer function of the delay (original).

Based on the above transfer function, we tried to explore the source of the particular deviation from the linearized delay dependence. We shifted the physical placement of the delay line on the FPGA chip. The Delay line was moved about 200 blocks down, keeping the same vertical line of the hardware resources. The result of the described experiment can be seen in Figure 5.4. Again, the meaning of the axis is still the same as in Figure 5.2.

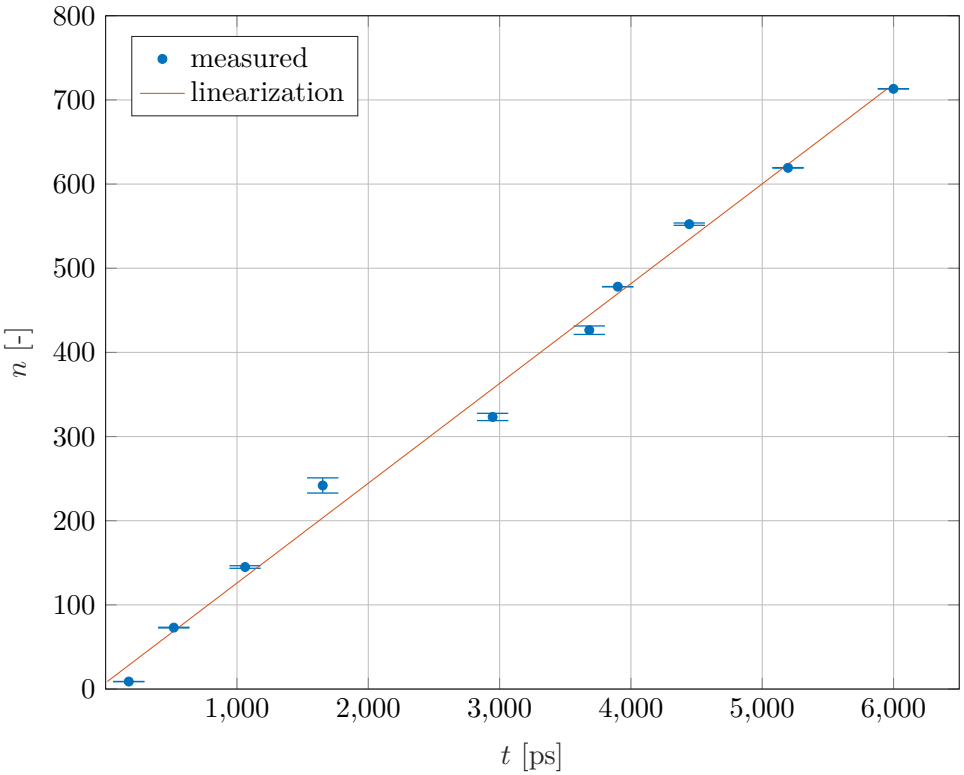


Figure 5.4: Transfer function of the delay line (shifted).

If we look at both last graphs a little bit closer and compare them, we can find some hint of shaky pattern around the linearization line. On the shifted version, the pattern moved roughly about 200 elements lower. It seems that nonuniform delay across the whole line manifests here.

The information about the average delay of one delay element is extracted from the measurement with averaging where the delay line was located in the original place. This value is now used as a calibration parameter for the calculation of the final delay measurement. Using whole measured range we determined average delay to the value $\tau_{\text{avg}} \approx 8.59\,\text{ps}$.

5.2 TDC validation

The next experiment is already with a complete TDC. In the measurement setup, we used rubidium frequency standard SRS FS725 as a source of the stable reference clock and the PPS signal for delay evaluation. The time interval counter SRS SR620 was used for measurement validation. Block diagram of the established measurement chain is shown in Figure 5.5.

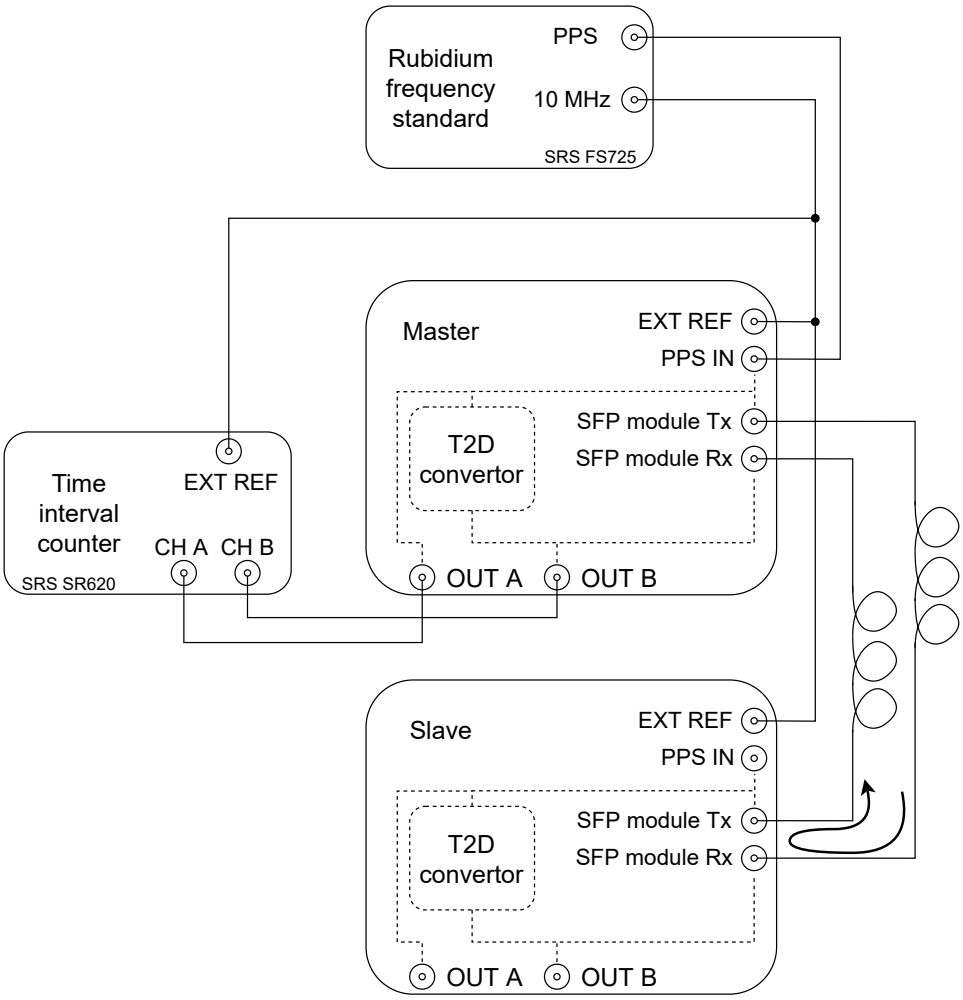


Figure 5.5: Connection of the measurement chain during the evaluation of the TDC.

To ensure as similar as possible measurement chain of the intended future usage, we split the chain into master and slave parts. Master received the PPS signal, sent the signal through an optical fiber to the slave device. On the slave part was implemented loopback to the master side. This setup should sufficiently simulate the future measurement chain setup. All described effort resulted in a comparison of the PPS signal received directly through a dedicated SMA connector and the PPS signal received through an optical fiber. Various delays were emulated by arranging multiple lengths of the optical fibers. Three lengths were available for measurement purposes. Time differences Δt between the our implemented TDC and time interval counter are presented in Figure 5.6. Varying fiber optic length is denoted on the x-axis labelled as l. Each particular point in the graph is the average result of the 500 samples.

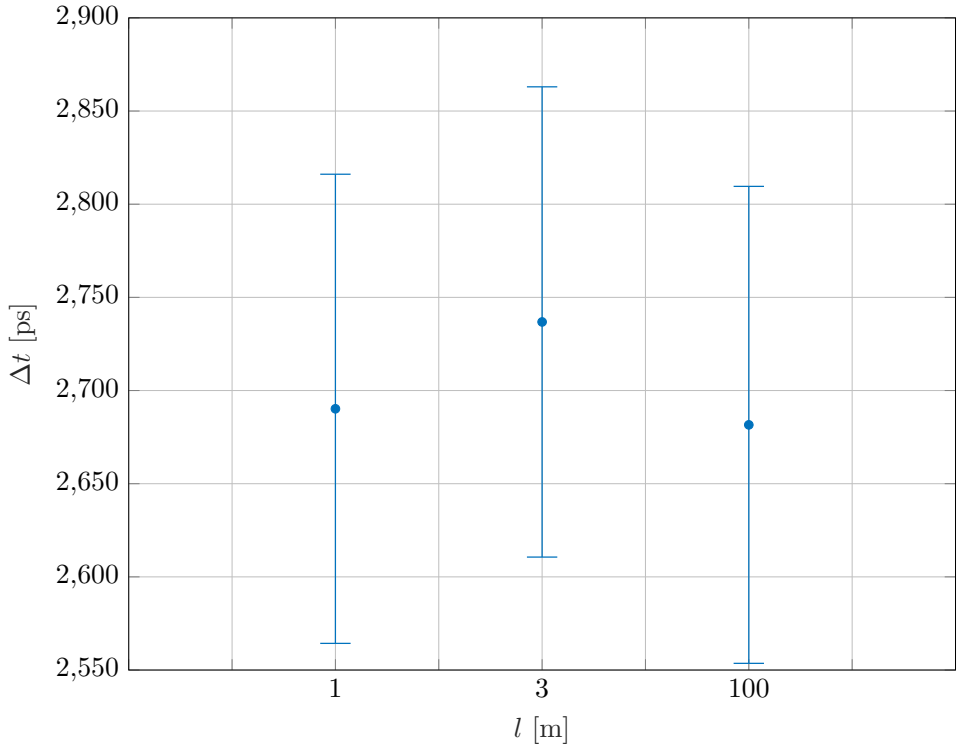


Figure 5.6: Validation of the TDC for three different lengths of optical fiber (before calibration).

The calibration of our TDC is needed due to the presented offset. Therefore, a new measurement with the calibrated device was performed. Its results can be seen in Figure 5.7.

Figure 5.7 shows that calibration improves delay differences to the range $\approx \pm 150\,\text{ps}$. A deeper investigation is needed for a more accurate specification of the implemented measurement design. However, even these results suggest the good performance of the implemented TDC.

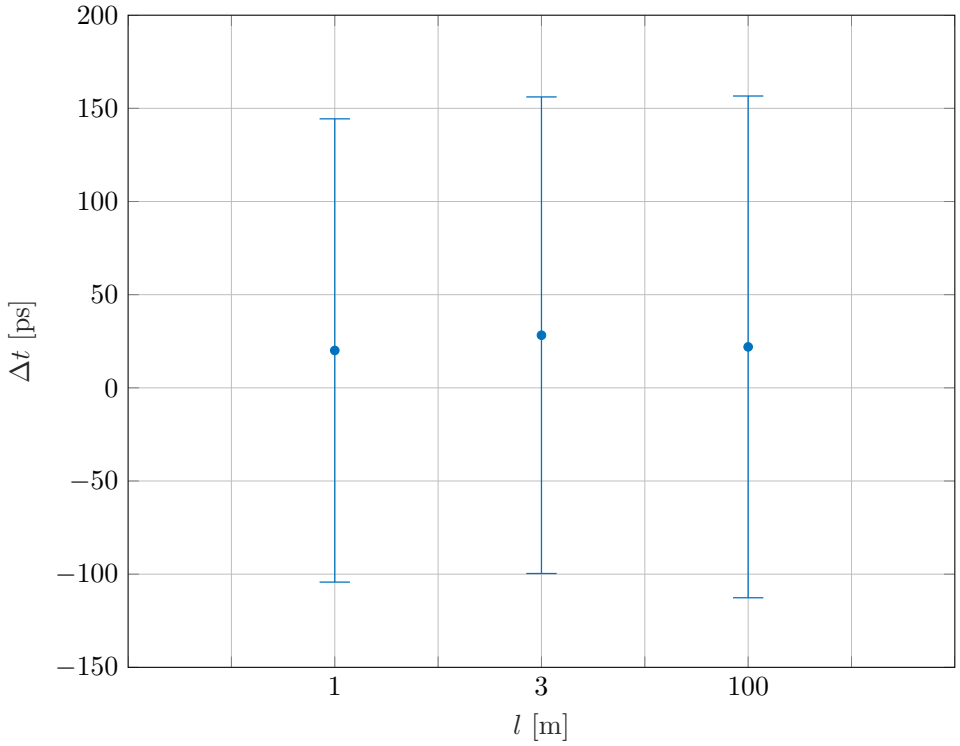


Figure 5.7: Validation of the TDC for three different lengths of optical fiber (after calibration).

Tests of the PPS signal transportation through an optical fiber show us an issue with signal reconstruction at the slave device. The PRBS modulation of the signal failed for particular clock signal phase shifts between master and slave device. The Initial master-slave clock internal synchronization can probably solve the issue. This task is not the easy one. The second available optical channel is used for master clock distribution as a temporary solution. It ensures the sufficient synchronization of the scrambler and descrambler on both sides. The periodical synchronization of the master and slave clock would be the appropriate solution of the described issue in the future.

Chapter 6
Conclusion

The goal of this master thesis is to implement the Time-to-Digital Converter inside an FPGA, design the hardware for two time scales comparison, implement the firmware for SFP modules, and create the user application. The starting point of the implementation was a theoretical analysis of the FPGA implemented TDCs. Investigation resulted in a decision to implement a converter using a combination of the delay line and counter. We were facing multiple issues during the implementation due to development software optimization and hardware limitations. Despite the troubles, we deployed the Time-to-Digital Converter on an FPGA with a resolution of tens picoseconds. The board Terasic DE10-Standard was used for the deployment itself. Unfortunately, its periphery interface does not fit our next task with the time scales comparison. Therefore, we had to design a new extension board that provides a suitable interface for SFP modules, clock signals, PC connection, and other minor functionalities. The processor Nios is also used in FPGA design. It ensures communication with PC, handling SFP module configuration and controlling the complete FPGA design. The user can control the whole system for time scales comparison from the PC. The application provides access to SFP module configuration registers. It also ensures visualization of measured time differences. The measured data can be exported to the CSV file.

Unfortunately, the final test of the complete system function was not performed. We managed the evaluation of the implemented delay line. Testing of the constructed TDC indicated excellent performance, but more extensive testing is needed for a complete view. The transfer of the PPS signal through the optical fiber was also tested. It works without any issues, but it is also necessary to transmit the clock signal of the master device via another optical line at this moment. It should be mentioned that based on the particular subsystem testing, there is no apparent reason why the final system should not work as we expect.

More advanced tests of the TDC itself would be surely needed in the future. The whole system should be subjected to tests as well. The PPS signal transfer is another area for future improvement. It would be advantageous to remove the slave side dependency on the master clock signal. The clock signal recovery can probably solve the problem during the initialization at the slave side. A new hardware design dedicated only to time scales comparison could be beneficial in the future.

Bibliography

[1] J. Savory, J. Sherman, and S. Romisch, "White Rabbit-Based Time Distribution at NIST," *IFCS 2018 - IEEE International Frequency Control Symposium*, pp. 1–5, 2018.

[2] M. Maamoun, S. Arami, R. Beguenane, A. Benbelkacem, and A. Meraghni, "A 3ps resolution time-to-digital converter in low-cost FPGA for laser rangefinder," *Lecture Notes in Engineering and Computer Science*, vol. 2229, no. figure 2, pp. 259–263, 2017.

[3] H. Wang, M. Zhang, and Y. Liu, "High-resolution digital-to-time converter implemented in an FPGA chip," *Applied Sciences (Switzerland)*, vol. 7, no. 1, pp. 1–11, 2017.

[4] R. Jiang, C. Li, M. Yang, H. Kobayashi, Y. Ozawa, N. Tsukiji, M. Hirano, R. Shiota, and K. Hatayama, "Successive approximation time-to-digital converter with vernier-level resolution," *2016 IEEE 21st International Mixed-Signal Testing Workshop, IMSTW 2016*, pp. 1–5, 2016.

[5] "Terasic DE10-Standard Development Kit." [Online]. Available: https://rocketboards.org/foswiki/Documentation/DE10Standard

[6] "Cyclone® V 5CSXC6 FPGA - Product Specifications." [Online]. Available: https://www.intel.com/content/www/us/en/products/sku/210462/cyclone-v-5csxc6-fpga/specifications.html

[7] P. P. Chu, *Embedded SoPC Design with Nios II Processor and VHDL Examples*, 1st ed. US: Wiley, 2011.

[8] S. Tancock, E. Arabul, and N. Dahnoun, "A Review of New Time-To-Digital Conversion Techniques," *IEEE Transactions on Instrumentation and Measurement*, vol. 68, no. 10, pp. 3406–3417, 2019.

[9] W. Lewandowski and C. Thomas, "GPS Time Transfer," *Proceedings of the IEEE*, vol. 79, no. 7, pp. 991–1000, 1991.

[10] A. A. Söderqvist, "White Rabbit Master Thesis : A Timing System Application using White Rabbit," no. January, 2014.

[11] J. Serrano, T. Fleck, P. Alvarez, R. Bär, M. Kreider, J. Lewis, P. Loschmidt, S. Rauch, E. G. Cota, T. Wlostowski, P. Moreira, G. Gaderer, C. Prados, J. Dedic, and M. Cattin, "The white rabbit project," 2009.

[12] J. Wu and Z. Shi, "The 10-ps wave union TDC: Improving FPGA TDC resolution beyond its cell delay," *IEEE Nuclear Science Symposium Conference Record*, pp. 3440–3446, 2008.

[13] M. Bousonville and J. Rausch, "Velocity of signal delay changes in fibre optic cables," *DIPAC 2009 - 9th European Workshop on Beam Diagnostics and Instrumentation for Particle Accelerators*, pp. 248–250, 2009.

[14] *PL133-37, Low-Power, 1.62V to 3.63V, 1 MHz to 150 MHz, Inverting 1:3 Fanout Buffer IC*, Microchip Technology Inc., 2020. [Online]. Available: https://www.microchip.com/wwwproducts/en/PL133-37

[15] *DSLVDS1001, 400-Mbps, Single-Channel LVDS Driver*, Texas Instruments, 2018, rev. A. [Online]. Available: https://www.ti.com/lit/ds/symlink/dslvds1001.pdf?ts=1621550189849&ref_url=https%253A%252F%252Fwww.google.com%252F

[16] *5PB1102PGGK8, 1.8V to 3.3V High-Performance LVCMOS Clock Buffer Family*, Renesas Electronics Corporation, 2021. [Online]. Available: https://cz.mouser.com/datasheet/2/698/REN_5PB11xx_Family_DS_DST_20210331-1711082.pdf

[17] E. Dierikx, and Y. Xie, "White Rabbit Good Practice Guide," May 2019.

[18] *INF-8074i, Specification for SFP (Small Formfactor Pluggable) Transceiver*, 2001, rev 1.0. [Online]. Available: https://members.snia.org/document/dl/26184

[19] *FT230X, USB TO BASIC UART IC*, Future Technology Devices International, 2016, version 1.4. [Online]. Available: http://www.ftdichip.com/Support/Documents/DataSheets/ICs/DS_FT230X.pdf

[20] *ADUM226N0WBRIZ, 5.0 kV RMS Dual Channel Digital Isolators*, Analog Devices, 2019. [Online]. Available: https://www.analog.com/media/en/technical-documentation/data-sheets/ADuM220N-221N-225N-226N.pdf

[21] Volnei A. Pedroni, *Digital electronics and design with VHDL*, 2008.

[22] FPGA Design Software - Intel Quartus Prime. [Online]. Available: https://www.intel.com/content/www/us/en/software/programmable/quartus-prime/overview.html

[23] *Cyclone V Device Handbook, Volume 1: Device Interfaces and Integration*, Altera Corporation, 2020. [Online]. Available: https://www.intel.com/content/dam/www/programmable/us/en/pdfs/literature/hb/cyclone-v/cv_5v2.pdf

[24] S. Šajić, N. Maletić, B. M. Todorović, and M. Šunjevarić, "Random binary sequences in telecommunications," *Journal of Electrical Engineering*, vol. 64, no. 4, pp. 230–237, 2013.

[25] F. Aznar, S. Celma, and B. Calvo, *CMOS Receiver Front-ends for Gigabit Short-Range Optical Communications*, 2013.

[26] E. Casas, "PN Sequences and Scramblers," *ELEX 3525: Data Communications*, p. 2, 2014. [Online]. Available: http://www.ece.ubc.ca/~edc/3525.jan2014/lectures/lec13.pdf

[27] Qt | Cross-platform software development for embedded & desktop. [Online]. Available: https://www.qt.io

[28] *SFF-8472, Specification for Management Interface for SFP+*, SNIA, 2021, rev. 12.4. [Online]. Available: https://members.snia.org/document/dl/25916

[29] *33522A, 30 MHz Function/Arbitrary Waveform Generators*, Keysight Technologies, 2019. [Online]. Available: https://www.keysight.com/zz/en/assets/7018-02567/data-sheets/5990-5914.pdf

[30] *SR620, Universal Time Interval Counter*, Stanford Research Systems. [Online]. Available: https://www.thinksrs.com/downloads/pdfs/manuals/SR620m.pdf

[31] *FS725, Benchtop rubidium frequency standard*, Renesas Electronics Corporation. [Online]. Available: https://www.thinksrs.com/downloads/pdfs/catalog/FS725c.pdf

Appendix A

Meaning of the used shortcuts

shortcut	meaning
TDC	Time-to-Digital Converter
PPS	Pulse Per Second
PTP	Precision Time Protocol
WR	White Rabbit
SAR	Successive-Approximation
ADC	Analog-to-Digital Converter
DAC	Digital-to-Analog Converter
TTL	Transistor-Transistor Logic
LVDS	Low-Voltage Differential Signaling
FPGA	Field-Programmable Gate Array
HPS	Hard Processor System
SFP	Small Form-factor Pluggable
PCB	Printed Circuit Board
PRBS	Pseudo-Random Binary Sequence
MLS	Maximum-Length Sequence
UART	Universal Asynchronous Receiver-Transmitter
CSV	Comma-Separated Values
HDL	Hardware Description Language
VHDL	VHSIC Hardware Description Language
VHSIC	Very High-Speed Integrated Circuits Program
RISC	Reduced Instruction Set Computer

Table A.1: Shortcuts overview.