Design Methodology of TDC on Low Cost FPGA Targets

Case Study: Implementation of a 42 ps Resolution TDC on a Cyclone IV FPGA Target.

Foudil Dadouche, Timothé Turko, Wilfried Uhring, Imane Malass, Jérémy Bartringer, Jean-Pierre Le Normand
ICube, UMR 7357, University of Strasbourg and CNRS
Strasbourg, France
Email: wilfried.uhring@unistra.fr

***Abstract*— This work aims to introduce a design methodology of Time-to-Digital Converters (TDCs) on low cost Field-Programmable Gate Array (FPGA) targets. First, the paper illustrates how to take advantage of the presence of carry chains in elementary logic elements of the FPGA in order to enhance the TDC resolution. Then, it describes how to use the Chip Planner tool to place the partitions composing the system in user specified physical regions. This allows the placement of TDC partitions so that the routing paths are constrained. As a result, the user controls the propagation delay effectively through the connection network. The paper ends by applying the presented methodology to a case study showing the design and implementation of high resolution TDC dedicated to fast imaging systems. The obtained resolution of 42 ps using a low cost FPGA target Cyclone family is very promising and suitable for a large amount of fast applications.**

Keywords- Time-to-Digital Converter; FPGA; Chip Planner; Carry Chain Logic.

I. INTRODUCTION

Nowadays, numerous applications require a precise measurement of time duration separating two or several physical events. 3D scanners or 3D console games represent typical application requiring precise time quantification of the interval time to reconstitute a three-dimensional scene. Such systems are generally based on Time Of Flight (TOF) measurement of the light emitted by a laser diode or a light-emitting diode (LED) and detected by a suitable light sensors after reflection by an object. The TOF of the light is proportional to the distance traveled by the latter. The measurement is made independently by several pixels allowing the reconstitution of the 3D scene [1][2].

To measure this duration, we use devices capable of converting extremely low time durations (some tens of ps) into digital values understandable for downstream processing and conditioning chain. These devices are commonly known as Time-to-Digital Converters (TDCs) [3].

In order to design such systems there are several techniques which are proposed in literature. Some of the techniques that can be readily identified are [3]-[5]: Tapped Delay Lines (TDL), Delay Locked Loop (DLL), Vernier Delay Line (VDL), Multilevel TDC, etc. All of these Time-to-Digital conversion techniques are usually designed as Application-Specific Integrated Circuits (ASICs). The latter have the advantage to have high performances but suffer from a higher cost, slow time to market and limited reconfiguration possibilities. It is also worth noting that the ASIC solutions are not suitable for integration into reconfigurable digital designs mostly described in Hardware Description Languages (HDL). As a result, numerous solutions for implementing TDCs on FPGA circuits have emerged [6]-[11]. However, the most significant limitation of these architectures is the difficulty to predict the placement and routing delays as well as the time delay of logic gates themselves. The consequence of this inevitable hardware restriction is a non-stable resolution of the designed TDC [8].

In this work, we aim to introduce a design methodology for high resolution TDC on low cost FPGA targets. This methodology enables the mastering of the network routing delays as well as the delays of the gates themselves. Therefore, it leads to an optimized TDC design with stable and accurate resolutions.

In order to give some background, the functional principle as well as the structure of the studied TDC in given in the second section. We also point out some associated difficulties encountered while using classical inverters as delay cells. The third section is dedicated to present our approach of implementing a TDC on an FPGA. Firstly, we show how to take advantage of the Carry Chain Logic to enhance and optimize TDC resolution. Secondly, we illustrate how to use the Chip Planner to define the exact physical layout location in the Chip. Therefore, we point out the importance of this operation. Section 4 provides a detailed case study consisting of implementing a 42 ps resolution TDC on Altera Cyclone IV low cost FPGA. The implemented TDC is associated to an FTDI (Future Technology Devices International) USB interface circuit operating in parallel mode with transfer rates reaching up to 40 Mbytes per second. Finally, we end our work by providing some final observations.

II. SETTING IN THE CONTEXT

A. Functional principle of the studied TDC

A TDC is an electronic system that measures the interval time between two occurring events of a given signal. Its main purpose is to convert temporal information to binary sequence understandable for a downstream processing chain.

For an accurate time duration measurement, generally, a TDC is composed of three blocks: two fine measurement blocks and a coarse one. The coarse one counts the number (N) of clock periods between enabling to disabling the

 ISBN: 978-1-61208-425-1

measured interval, and the fine blocks evaluate the uncertainties in both sides that cannot be counted since their duration is shorter than the clock period.

To understand the role of each one of the three blocks, we illustrate by the timing diagram of Figure 1 the functioning principle of a generic TDC.

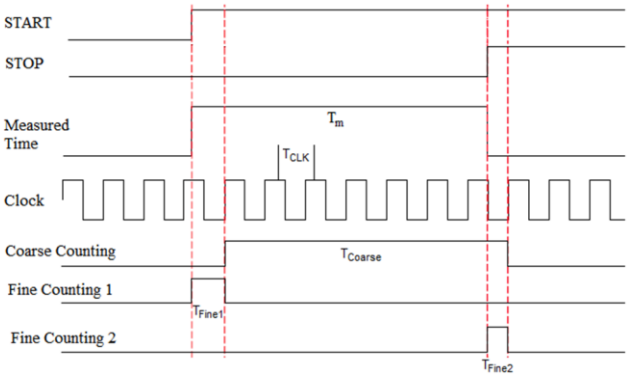

Figure 1. Functioning principle of a generic TDC

As we can see from this timing diagram, the time interval to be measured (T_m) is a combination of three individual durations: (1) T_{Coarse}, which represents the number of clock periods from enabling to disabling coarse measurement, (2) T_{Fine1}, representing the time between the measured signal active edge and the first following rising clock edge, and (3) T_{Fine2} which is the time between the falling edge of the measured interval and the following rising clock edge. Accordingly to this timing diagram, the measured time will be expressed as follows:

$$T_m = T_{Fine1} + T_{Coarse} - T_{Fine2} = T_{Fine1} + N \cdot T_{clk} - T_{Fine2} \quad (1)$$

In practice, TDCs are mostly used in fast imaging systems needing to know the delay separating a photon emission by a laser diode and the detection of that photon by a Single-Photon Avalanche Diode (SPAD). Therefore, the events can be represented by two signals: (1) a START signal, which can be synchronized with the coarse counter clock, and (2) a STOP signal that means that the SPAD has detected a photon. In that case, the whole TDC can be reduced to a coarse counter associated to a fine TDC measuring T_{Fine2}. Consequently, the measured interval time will be given by the expression hereafter:

$$T_m = N \cdot T_{clk} - T_{Fine2} \quad (2)$$

Since the coarse block is a simple counter incremented by the system clock, we will focus in the following section on the implementation of the fine TDC.

B. *Structure of the studied TDC*

As mentioned previously, there are different techniques of designing TDCs. In this work, we focus on the commonly used Tapped Delay Lines (TDL) architecture depicted in Figure 2.

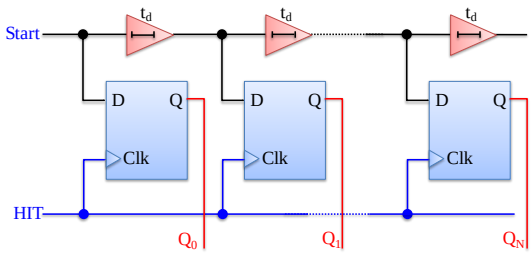

Figure 2. Tapped delay line TDC

A TDL TDC consists of N cascaded delay elements whose inputs are stored in D Flip Flops (DFFs). We would then have as many DFFs as there are delay elements. Therefore, each delay element can be regrouped with its associated DFF to form an elementary cell of the TDC.

The number (N) of these elementary cells depends of the common DFF clock frequency, as well as the propagation time of the delay element (t_d). This is given by the ratio of clock period to propagation time t_d. Since the value of the previous parameter is not provided, it is determined experimentally.

C. *Design and validation of the elementary cell*

In order to design the elementary cell of the studied TDC, we first used a simple invertor as a delay element associated with a DFF as illustrated by Figure 3.

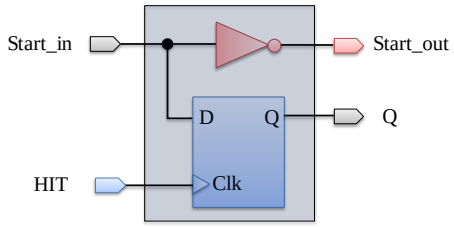

Figure 3. Simple TDC elementary cell

However, implementing a TDC chain on an FPGA by duplication of this cell leads to a simplified circuit entirely different from the desired function. Indeed, if the input signal Start_in and the output signal Start_out are equipotential, the used HDL software (Quartus II) will simplify the logical equation giving the output versus the input so that it saves place and time.

To illustrate this phenomenon, we represent in Figure 4 the RTL view resulting from the implementation of a simple TDC chain composed of four elementary cells. It can readily be seen that, in spite of the presence of inverters, the software has simplified the logical equations. Consequently, all the inverted signals are grouped independently of the non-inverted ones. It is thus evident that this method is not suitable for designing a TDC. Nevertheless, it is worth to notice that, if it is not possible to prevent Quartus II software to optimize data path, it is quite possible to create this path manually by operating directly on the logical resources of the FPGA. Indeed, the Quartus II Chip Planner tool allows physical access to logical resources available on the chip.

 ISBN: 978-1-61208-425-1

Using this tool, we can perform a customized configuration of the logic elements and impose the data path. However, the manual configuration of logic elements is tedious and time consuming in particular for systems with a certain complexity such as TDCs.

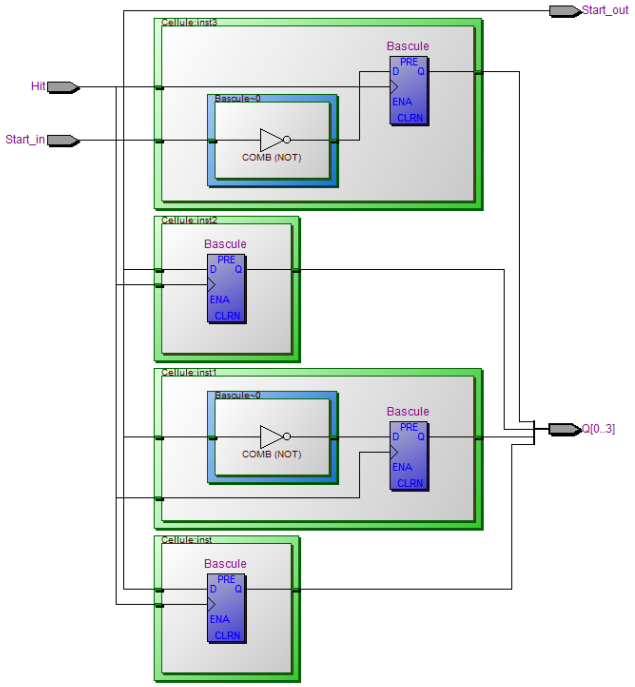

Figure 4. RTL view of the implementation of a simple 4-cells TDC

Even if we can use this technique to implement a TDC on an FPGA, given the large number of logic elements to be configured individually, it is still difficult to set up. Moreover, the TDC chain size can vary from an application to another; it will be therefore preferable to automate the configuration so that the solution will be generic and adaptable. Hence, we propose an appropriate design methodology in the following section.

III. DESIGN METHODOLOGY

In order to provide solutions to the above raised issues, in this section we suggest an alternative approach that can carry out a TDC structure fulfilling the following needs:

- Avoid the software data path simplification
- Increase TDC resolution by reducing the propagation time through delay elements
- Automate the elementary cells set-up process to optimize the design time and make possible the development of generic and adaptable structures
- Use a low cost FPGA target to implement the TDC.

This method is focused on two main areas:

- Using adders as delay elements and utilization of the Carry Chain Logic of the FPGA
- Using the Chip Planner tool.

A. *Using Adders and Carry Chain Logic*

The implementation of digital circuits on FPGA targets depends on the architecture of the logical resources of the target. In this work, we are aiming to use a low cost FPGA from Altera Cyclone family. The selected target is the Cyclone IV (EP4CE55F23C8) based on the logic element shown by Figure 5 [12].

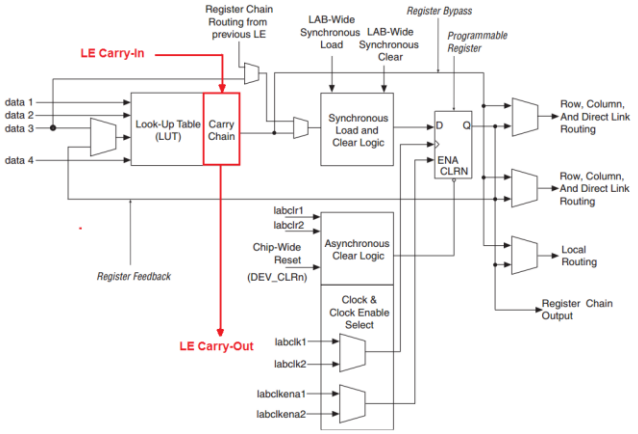

Figure 5. Cyclone IV logic element structure

The Cyclone IV logical element, given by Figure 5, provides a dedicated path for fast carry propagation. The role of this carry chain is to use specific fast paths for carry propagation instead of general-purpose routing network. By doing so, it makes it possible to drastically optimize the propagation time. This is ideal for the enhancement of the TDC resolution. Moreover, it allows to harmonize the delays of the TDC elementary cells.

The problem is that customized handling of carry chains is reserved to high performance FPGAs as such as the Stratix family from Altera whose cost is outstandingly high. However, it is possible to configure the Quartus II synthesis tool to optimize speed. In this case, the synthesis tool uses the carry chain logic automatically when synthetizing an HDL model involving adders.

It is therefore possible to use the carry chain logic to minimize and harmonize propagation delays for components involving adders. It is precisely the idea that is exploited here to design TDC elementary cells based on simple adders. This was done by developing a simple behavioral VHDL model for an adder with a customizable number of elementary cells. The number of the cells depends on the data width modeled by a generic parameter called DATA_WIDTH. The whole model is given by Figure 6.

The fine TDC using adders can be performed by: (1) applying the TDC input signal STOP to the carry input signal (*cin*) of the adder, and (2) choosing values for adder operand inputs (*a* and *b*) so that an output carry is generated (*cout*='1') if input carry is equal to '1'. The output carry is then an exact replication of the input carry delayed by a transmission time through the cell. To do so, all it takes is to set all the bits of the first operand to '1' and the bits of the second operand to '0'. For each bit (*i*) the arithmetic sum

 ISBN: 978-1-61208-425-1

a(i)+b(i) gives '1'. When the input carry is activated (*cin*='1') by the TDC input signal (*STOP*), the arithmetic sum *a+b+cin* gives '0' and the carry output moves to '1'.

```
entity signed_adder is
    generic ( DATA_WIDTH : natural := 127 );

    port (   a      : in signed  ((DATA_WIDTH-1) downto 0);
             b      : in signed  ((DATA_WIDTH-1) downto 0);
             cin    : in std_logic;
             cout   : out std_logic;
             result : out signed ((DATA_WIDTH-1) downto 0) );
end entity;

architecture rtl of signed_adder is

begin

PROCESS (cin, a, b)
    VARIABLE  s : signed ((DATA_WIDTH) downto 0);
    BEGIN
        s := ('0' & a) + ('0' & b) + ('0'&cin);
        result<=s((DATA_WIDTH-1) downto 0);
        cout <= s(DATA_WIDTH);
    END process;
end rtl;
```

Figure 6. Adder VHDL model

Figure 7 illustrates the implementation of one elementary cell of a TDC by a logic element of the Cyclone IV target. The adder cell is obtained by the look up table (LUT) and the DFF by the sequential configurable output register.

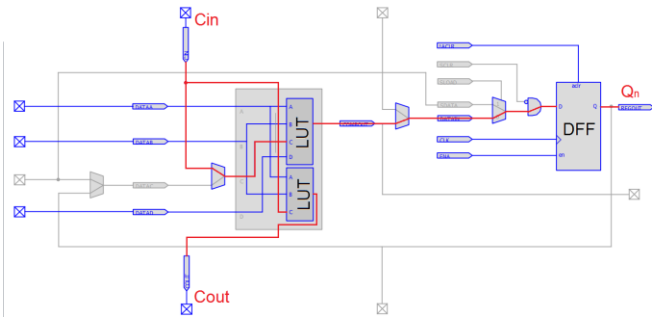

Figure 7. Implementation of a TDC elementary cell by a logic element

Theoretically, to obtain a TDC chain similar to the TDL structure shown by Figure 2, it is sufficient to duplicate the structure of Figure 7 as often as necessary to reach the number of desired cells. However, when implementing such a chain on the FPGA, some DFFs of the TDC elementary cells are dissociated of their corresponding 1-bit adder cells even if the data path is perfectly respected. This phenomenon occurs randomly and leads to the placing of the DFF and the delay element of the same TDC's elementary cell in different logic elements, as shown in Figure 8.

The direct consequence of component misplacing is that the delay time is no longer identical for all cells. This inevitably generates unpredictable artifacts. To ensure a reliable operation, it is necessary to overcome this problem by constraining the placement tool to bring together the components of the same cell in the same logic element. This is the purpose of the next section.

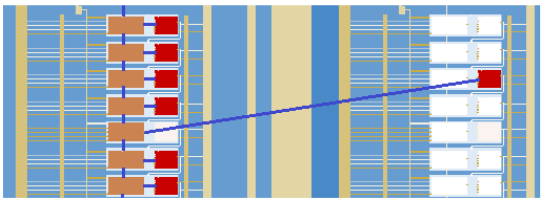

Figure 8. Random placing of DFFs on the chip

B. *Using Chip Plannar*

Using a TDC in fast imaging systems requires the measurement of very short time durations. It is therefore necessary to master all of the signal propagation delays through the cells as well as the routing network.

As we have seen in the previous section, unconstrained automatic implementation of a TDC on an FPGA usually leads to an inhomogeneous and irreproducible structure. Consequently, the measurement results are tainted by these uncertainties. Therefore, it is necessary to control the exact physical location of TDC cells on the chip.

This could be achieved by using the Chip Planner tool provided by Altera. The latter, according to the user's needs, allows the defining of specified implementation regions on the chip for blocks constituting the whole system. In addition, it supports incremental compilation to preserve the well-implemented parts and reduce the compilation time. This operation takes place in three distinct steps:

- Creating Design Partitions: the first step consists of dividing the design in individual partitions according to system complexity as well as user needs.
- Defining logic regions: after partitioning the design, it is necessary to define logical zones that will be associated to the partitions. This allows individual compiling and optimizing of each region. The tool used to perform this operation is LogicLock Region (LLR) within Chip Planner.
- Physical assignment of logic regions: in order to physically preserve the logic regions defined in the previous step, by means of the LLR tool, physical regions of the chip are assigned to implemented partitions.

The physical delimitation of regions permits to constrain the placing and root tool to put partitions in their specified regions defined by the user. Doing so, it allows not only avoiding the random placement of certain DFFs away from their associated delay elements, but also implementing the concerned partitions as close as possible to input signal pins (*HIT* and *STOP*). The purpose of the latter operation is to reduce the propagation delays of input signal before reaching the blocks to which they are intended to be applied.

For illustrative purposes, we represent on Figure 9 the assignments of physical allocations of the partitions defined using LLR and a close-up view of the layout of a 16-cells fine TDC implemented using the method presented above.

 ISBN: 978-1-61208-425-1

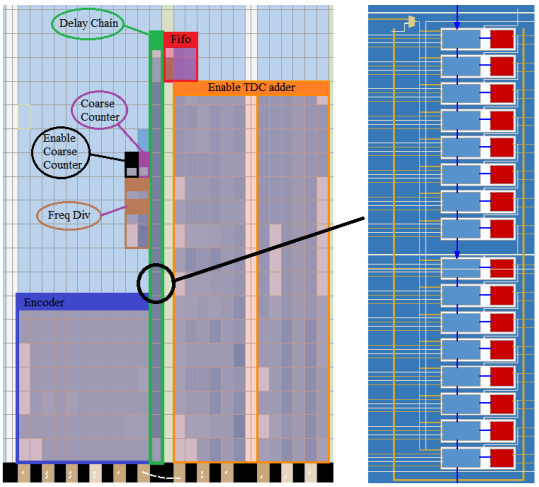

Figure 9. Layout of implemented partitions of a TDC

The TDC fits perfectly within the reserved region that would be assigned to it. Consequently, the DFF and the delay element of each TDC elementary cell are now implemented by the same logic element. The transmission delays are then identical for all cells.

IV. CASE STADY: IMPLENTATON OF A 42 PS TDC ON A CYCLONE IV FPGA

The proposed TDC design has been implemented within the Cyclone IV (EP4CE55F23C8) FPGA target. The coarse counter clock is 200 MHz, i.e., the clock period is 5 ns. The delay line for the fine TDC, based on carry chain adder architecture, comprises 127 cells in order to cover a dynamic of more than 5 ns. The signal that needs to be measured propagates through the delay chain, until the FPGA clock disables the DFFs to block their outputs and then memorizes their states. The value of these DFFs describes the time spent between the signal STOP and Clock. An encoder placed at the output of the TDC prepares the data before saving it into a FIFO (First In First Out) memory. The data is then transmitted to a USB port via an FTDI FT232H operating in parallel mode with transfer rates reaching up to 40 Mbyte per second. To acquire data measurements, we developed a specific application using LabVIEW software. Figure 10 shows the synoptic view of the system.

The TDC has been characterized on its whole dynamic, i.e., from 0 to 640 ns with a step of 5 ps. A Stanford research DG 645 digital delay generator has been used to generate the START and STOP signals. At this range of delay, the jitter of the delay generator is lower than 25 ps rms. Figure 11 shows the detail of the unconstrained and constrained fine TDC measurements between two reference clock edges, i.e., on a range of 5 ns. The unconstrained fine TDC response shows a large discrepancy of the LSB value indicating that some DFFs have been randomly placed. The resulting large steps make the unconstrained fine TDC unusable for sub nanosecond timing. Consequently, the use of the Chip Planner tool as described in section III is mandatory to obtain the behavior of the constrained fine TDC represented by the green curve. A linear fit is then used to assess the LSB value of the fine TDC which is given by the inverse of the linear fit slope, i.e., 41.5 ps in this study case.

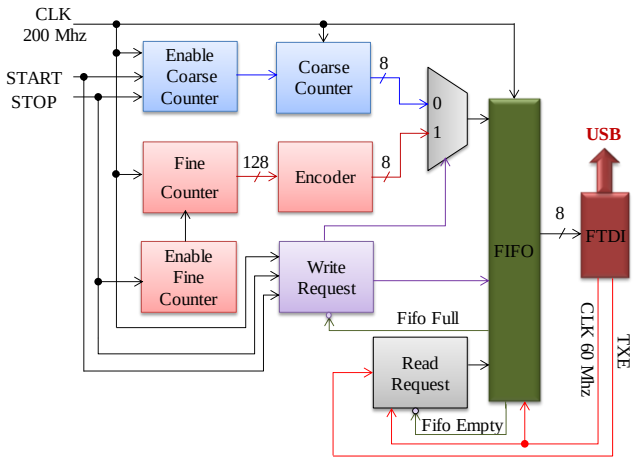

Figure 10. Synoptic view of the implemented TDC system

The noise visible on the fine TDC response is due to the jitter. The latter adds uncertainty on each measurement and it can be evaluated by computing the standard deviation of a set of measurements at a given fixed delay between the START and STOP signals.

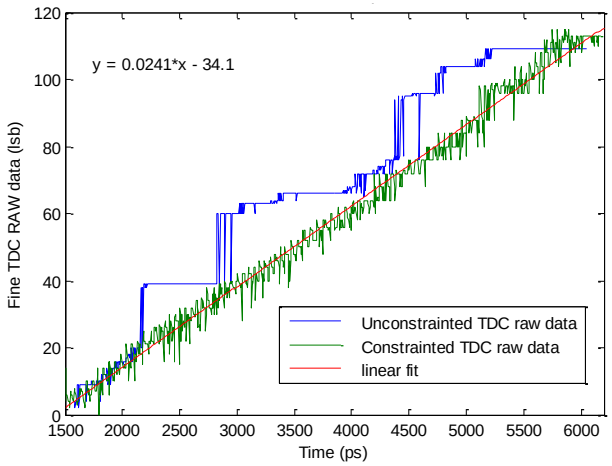

Figure 11. Responses of Fine unconstrained and constrained TDC

The jitter depicted in Figure 12 has been characterized for different delays corresponding to a given signal propagation along the fine TDC line. As each fine TDC elementary cell adds its own jitter [13], the global jitter will then increase as a square root of the number N cells as given by the following expression:

$$\sqrt{\alpha^2 + \beta^2 \cdot N} \quad (3)$$

Where: α is the initial jitter present at the input of the first cell and β the single cell jitter. A curve, following this law is fitted on the jitter profile to underline the jitter's variation relationship in the delay line. The extraction of this

parameters leads to an initial jitter α of 62 ps rms and a single cell jitter β of 5.8 ps rms. The jitter is mainly due to a noise presents on the 1.2 volt FPGA core power supply. The accumulated jitter across the fine TDC delay line leads to a mean jitter of 90 ps rms. Thus, the line length has to be kept as low as possible in order to obtain the best accuracy. This can be done by using the fastest achievable frequency for the coarse counter.

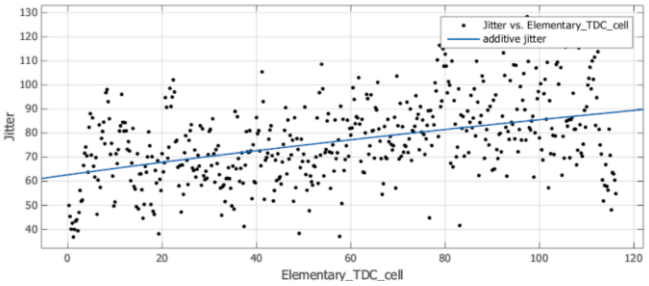

Figure 12. Jitter measurement according to the elementary level, the jitter increases as the signal propagates along the fine TDC cells

The integral non linearity error (INL) and the differential non linearity (DNL) have been measured over the entire range of the TDC. For illustrative purposes, the results from a delay of 0 to 160 ns are represented by the figure hereafter. From Figure 13, it can be seen that, the implemented system shows an INL of 132 ps rms and a DNL of 50 ps rms.

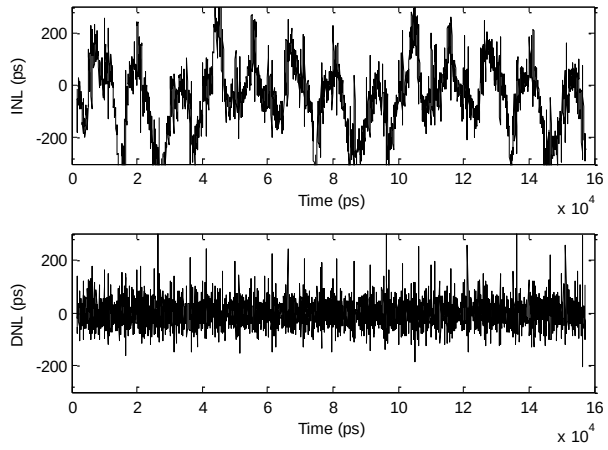

Figure 13. INL and DNL errors of the implemented TDC over a range of 160 ns

V. CONCLUSION

This paper has proposed a global methodology to design and implement Time-to-Digital Converters on low cost FPGA targets. It presents how to use different tools to enhance the TDC resolution by reducing propagation delays through the connection network as well as the logic gates themselves. First, the use of adders as delay elements, to benefit from a dedicated carry chain logic path, is presented. Then we detailed how to take advantage of the chip planner, to constrain the placing and root tool to put the partitions of the system in user specified physical regions. Doing so, it allowed the mastering of propagation delays and consequently improved the resolution and the stability of the TDC.

The work is ended by a case study that applied this methodology to design a TDC with a resolution of about 42 ps on a Cyclone IV FPGA. The implemented TDC presents a jitter of less than 90 ps rms, and the DNL and the INL has been measured respectively to 50 and 132 ps rms.

The obtained results are very promising, not only because they are suitable for domains requiring high performances, but also because they are achieved by using a low cost FPGA family which opens the door to a broader use in a great amount of fast application fields.

As a perspective in the near future, we plan to integrate the presented TDC in different applications such as image photon counting devices and microfluidic experimentations.

REFERENCES

[1] L. Li. "Time-of-flight camera – an introduction". Texas Instruments, SLOA190B – Technical White Paper, January 2014, revised May 2014.

[2] E . Charbon, M. Fishburn, R. Walker, R. K. Henderson, and C. Niclass, SPAD-based sensors TOF Range-Imaging Cameras, Springer-Verlag, F. Remondino and D. Stoppa, Eds. Berlin Heidelberg, 2013, pp. 11–38.

[3] S. Henzler, Time-to-Digital Converters, Springer Science+Business Media B.V., 2010, DOI: 10.1007/978-90-481-8628-0.

[4] J. Kalisz, "Review of methods for time interval measurements with picoseconds resolution," Metrologia, vol.41, 2004, pp. 17–32.

[5] C. S. Hwang , P. Chen, and H. W. Tsao, "A high-precision Time-to-Digital converter using a two-level conversion scheme," IEEE Tr. Nucl. Sci., vol. 51, 2004, pp. 1349–1352.

[6] J. Kalisz, R. Szplet, J. Pasierbinski, and A. Poniecki, "Field-programmable-gate-array-based time-to-digital converter with 200-ps resolution," IEEE Trans. Instrum. Meas., vol. 46, 1997, pp. 51–55.

[7] I. Vornicu, R. Carmona-Galán, and Á. Rodríguez-Vázquez, "Wide range 8ps incremental resolution time interval generator based on FPGA technology," 21st Int Conf. Electron. Circuits Syst (ICECS), 2014, pp. 395–398.

[8] M. Lin, G. Tsai, C. Liu, and S. Chu, "FPGA-Based high area efficient Time-to-Digital IP design," in TENCON 2006, 2006 IEEE Region 10 Conference, 2006, , pp. l–4.

[9] R. Narasimman, A. Prabhakar, and N. Chandrachoodan, "Implementation of a 30 ps resolution Time-to-Digital Converter in FPGA," Int Conf on EDCAV, Shillong, 2015, pp. 12–17.

[10] A. Aloisio, P. Branchini, R. Giordano, V. Izzo, and S. Loffredo, "High-precision Time-to-Digital converter in a fpga device," IEEE Nuclear Science Symposium Conference Record, vol. 13, 2009, pp. 290–294.

[11] S. S. Junnarkar, P. O'Connor, and R. Fontaine, "FPGA based self calibrating 40 picosecond resolution, wide range Time-to-Digital converter," IEEE Nuclear Science Symp. Conf. Rec. NSS '08, 2008, pp. 3434–3439.

[12] Altera Corporation, Cyclone IV Device Handbook. Volume 1, chapter 2, April 2014.

[13] M. Zlatanski, W. Uhring, J.-P. Le Normand, and D. Mathiot, "A Fully characterizable asynchronous multiphase delay generator," Nuclear Science, IEEE Transactions on, vol.58, no.2, 2011, pp.418–425.

 ISBN: 978-1-61208-425-1