Gray-Code TDC with Improved Linearity and Scalability for LiDAR applications

Rui Machado
Algoritmi Centre
University of Minho
Guimarães, Portugal
id6010@alunos.uminho.pt

Filipe Serra Alves
International Iberian
Nanotechnology Laboratory
Braga, Portugal
filipe.alves@inl.int

Jorge Cabral
Algoritmi Centre
University of Minho
Guimarães, Portugal
jcabral@dei.uminho.pt

***Abstract*-This paper presents a TDC architecture based on a gray code oscillator with improved linearity, for FPGA implementations. The proposed architecture introduces manual routing as a method to improve the TDC linearity and precision, by controlling the gray code oscillator Datapath, which also reduces the need for calibration mechanisms. Furthermore, the proposed manual routing procedure improves the performance homogeneity across multiple TDC channels, enabling the use of the same calibration module across multiple channels, if further improved precision is required. The proposed TDC channel uses only 16 FPGA logic resources (considering the Xilinx 7 series platform), making it suitable for applications where a large number of measurement channels are required. To validate the proposed architecture and routing procedure, two channels were integrated with a coarse counter, a FIFO memory and an AXI interface, to assemble the pulse measurement unit. A comparison between the default routing implementation and the proposed manual routing has been performed, shown an improvement of 27% on the overall TDC single-shot precision. The implemented TDC achieved a 380 ps RMS resolution, a maximum DNL of 0.38 LSB and a peak-to-peak INL of 0.69 LSB, corresponding to a 21.7% and 70.4% improvement, respectively, when compared to the default design approach.**

Keywords: Time-to-Digital Converters (TDC), Field-Programmable Gate Array (FPGA), Time Interval Measurement, Gray Code Oscillator.

I. Introduction

The research on Time-to-Digital Converters (TDC) implemented in Field-Programmable Gate Arrays (FPGA) has gained increased relevance over the last few years [1], [2]. FPGA-based TDCs have been highly utilized in physics experiments [3]–[5] and metrology equipment [6]. The recent FPGA technology developments allowed these systems to achieve high resolutions and precision, capable of competing with the ones reported in Application Specific Integrated Circuits (ASIC) implementations [7]–[9].

Modern applications concerning time-of-flight (ToF) measurement require not only high performance but also low resource and power consumption. Applications targeting mobile or wearable devices are examples where area and power concerns are prioritized over resolution. System's requiring multiple ToF measurement channels are another example benefiting from low resource and power consumption TDC implementations.

Light Detection And Ranging (LiDAR) sensors, to name one of such applications, have seen an increasingly research popularity, due to the industry efforts on achieving the autonomous vehicle, in which LiDAR is considered as a Key enabler [10]–[12]. Moreover, LiDAR sensors are still under exploratory research [11]. Therefore, prototyping platforms like FPGA are attractive for implementing the LiDAR's digital building modules, in which the ToF measurement unit is included. Although the requirements for LiDAR sensors are still not completely defined and under development, according to [13], the trend points to a minimum distance depth resolution of 20 cm, which corresponds to a time interval equal to 1.33 ns, and a maximum range of 200 meters (1.34 µs). Following the classification presented in [13] and [11] regarding the currently explored LiDAR technologies, one can have scanning or staring LiDAR sensors. Staring LiDAR (FLASH and fixed multi-beam) have no moving part and demand for multiple receivers. Scanning LiDAR based on MEMS micro-mirrors can also require multiple receivers, when a 1-D scanning approach is adopted [13]. For such scenarios, multiple ToF measurement channels must be implemented. Moreover, due to the LiDAR's time constraints for frame processing and presentation to allow extra timing slack to be used in the post-processing and point-cloud frame handling, multiple ToF measurement units per receiver can be implemented. This allows an arithmetic average to be performed based on multiple parallel measurements of the same receiver, to increase the precision and the entire system's reliability. Therefore, low resource consumption TDCs to measure the ToF on multi-beam LiDAR based sensors are a must.

Regarding FPGA-based ToF measurement systems, three main TDC architectures can be identified: Tapped-Delay Lines (TDL); Phased Clocks; and Differential TDCs.

TDLs are by far the most studied architecture, with numerous works reporting resolutions better than 20 ps [8], [14]–[16]. The main drawback of TDLs is its linearity, which forces the system to be implemented with a calibration circuit [15], increasing the system resource usage and power consumption. Phased clocks offer a simpler design but lower resolutions in the range of a few hundreds of picoseconds [17]–[20]. Differential TDCs, although achieving greater linearity, have the most complex design and higher conversion times, resulting in high system deadtime [21], [22].

Recently a novel TDC architecture based on a gray code oscillator was introduced by Wu and Xu [23]. The architecture design focused on low power consumption and low resource usage, while offering high scalability and portability. The main drawback of this architecture is the mean bin size ranging from 256 ps to 271 ps, which is worse than the typical resolution reported using TDLs (considering the same FPGA family). Nevertheless, this solution is a great candidate for the aforementioned LiDAR sensor applications.

In this paper, an improved version of the gray-code oscillator TDC is proposed, based on the control of the gray-code counter's Datapath routing, with some small changes on

978-1-7281-9581-0/20/$31.00 ©2020 IEEE

the TDC channel. The proposed approach largely improves the TDC linearity, as well as the architecture's scalability, since the routing can be replicated to different channels without modifications, achieving this way a homogenized performance across all implemented channels. These improvements minimize the need for calibration further reducing the area and power consumption of the TDC architecture.

The remaining of this paper is structured as follows: Section II presents the proposed gray-code oscillator TDC architecture and its operation principle. In Section III the proposed approach and implementation process, for linearity and performance homogenization improvement, are explained. Section IV presents the obtained experimental results. The obtained results are also compared to the ones presented in [23] to prove the relevance and impact of the presented approach. Finally, Section V presents the paper main conclusions and future work perspectives.

II. Gray Code Architecture

The reference gray code oscillator architecture was first proposed in [23], reporting a mean bin size of 256 ps and 271 ps, for the two TDC channels implemented, using 8-LUTs and 8 flip-flops per channel on a Xilinx Kintex-7. Typically, a counter is composed by a combinatorial stage, which calculates the next value in the counting schema, and by a sampling stage, responsible for latching the value of the counter at each clock cycle, assuring a stable value for the combinatorial stage, so that the next value can be correctly calculated and latched in the next clock cycle. This is of extreme importance for binary counters in which multiple bits can change from one counting value to the next, activating multiple combinatorial paths at the same time. However, this is not the case for the gray code sequence. In the original Gray-code, only one-bit changes from one state to the next one. Thus, the Gray-code can be configured in a loop, without the latching stage, since the risk of missing codes or random counting sequence is not an issue. This enables the implementation of a counter with a resolution that is no longer limited by the system clock used to sample the state of the counter.

According to Xilinx documentation [24], on the 7-series FPGAs there are available 6-input LUTs. These LUTs are the logic elements used to implement combinatorial functions on FPGAs. Therefore, in order to be capable of implementing a function in a single LUT, no more than 6 variables can be used, limiting the maximum number of bits that the gray code counter can have to 6. However, since an enable signal is required, the gray code counter gets limited to 5 bits, resulting in a maximum of 32 interpolation steps. The gray code counter presented in [23] used a 5-bit reflected binary code (RBC) schema. The same schema will be used in the work presented here.

The block diagram of the proposed system is depicted in Fig. 1. In order to increase the proposed TDC architecture dynamic range, a coarse counter was implemented using a 16-bit binary counter incremented at each system clock tick. The gray code counter is used to implement the TDC Channel (start and stop) and gives the system fine measurement (time interval between system clock ticks). In order to reduce power consumption, an input stage was implemented to guarantee that the gray counters are not enabled for more than one system clock cycle. Both start and stop TDC Channels are

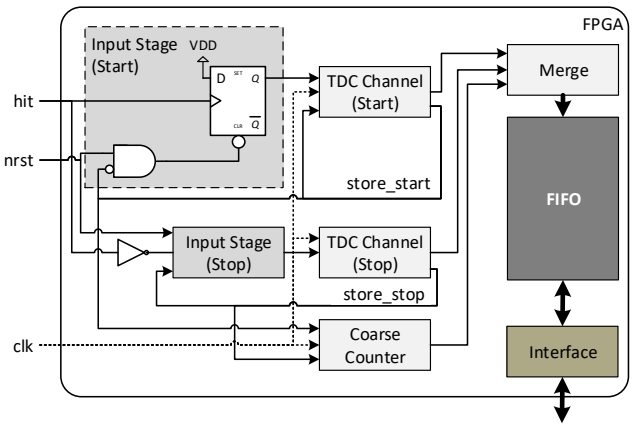

Fig. 1. TDC IP block diagram

identical. The Merge block concatenates the values from the coarse measurement and both TDC channels into a single 32-bit value and generates the control signals to store that data on a FIFO memory. Finally, the interface block implements an AXI-4 interconnection, for easy integration with Arm processors. The proposed schema allows for precise time interval measurement of a pulse (from its rise edge to its falling edge).

The main difference of the architecture proposed here when compared to the standard gray code oscillator is the method used to extend the fine measurement stage range. In [23] the TDC channel was extended using a 3-bit cycle counter, which counts the number of times the gray code counter reaches overflow. In order to reduce the fine measurement stage complexity, an implementation with only the 5-bit gray code is proposed in this work, being the dynamic range extended by an external binary counter. Although this decision forces the use of a faster clock, which can lead to higher power consumption, it also enables to save 3 LUTs, 3 flipflops and 1 carry block per TDC channel. The architecture of the proposed fine measurement stage is presented in Fig. 2.

The input time interval to be measured (denoted as hit) is filtered by the input stage and used to enable the gray counter of the fine measurement stage. The fine measurement stage is composed by the LUTs that generate the gray code sequencing, by a first set of registers, located in the same slice as the LUTs, and by a second set of registers, enabled only during one clock cycle after the arrival of a valid hit signal, that stores the fine measurement value for further processing. The store signal is used to enable the second set of registers, to reset the input stage and to sample the value of the coarse measurement stage. This allows to synchronize both counting stages, avoiding the risk of metastability.

The proposed system typical operation, during a measurement, is depicted in the waveform diagram of Fig. 3. The system's state machines are presented in Fig. 4. When a rise edge on the hit signal is detected, the start input stage generates the enable signal to the start channel, which begins to sequence through the gray code until the arrival of the next system clock positive edge. When the *start_sampled* signal changes to a value different from zero, an enable signal is generated internally to the *start_store* registers and to the registers responsible for sampling the coarse counter. Analogously, when the fall edge of the hit signal is detected the stop input stage generates the enable signal for the stop channel and after being sampled for the system clock, an

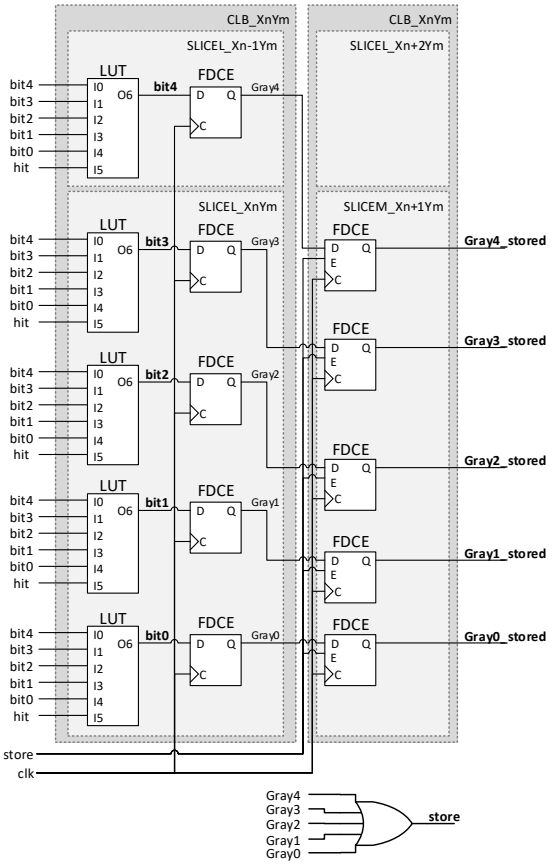

Fig. 2. Gray code oscillator TDC channel schematic

enable signal for the *stop_store* registers and for the coarse counter is generated. The final measurement value is composed by a coarse value (obtained by subtracting the two sampled coarse values at stop and start events) and by the two fine measurement values sampled from the gray code TDCs. A count reset signal is generated at the end of each conversion (after writing to the FIFO memory) to signal that the TDC is ready for a new measurement.

III. Design Improvement

From the analysis of the results reported in [23] two major conclusions can be drawn: first, the linearity of the TDC channel is highly dependent on the routing of the gray codes counter Datapath. According to Xilinx documentation [24], the propagation delay of a LUT is independent of the truth table that it is implementing. Thus, since differences as large as of 400 ps on the TDC channel steps' delay have been reported in [23], the routing resources must be the cause for such non-linearities. Moreover, contrarily to Carry chains, LUTs do not have a dedicated routing path, largely increasing the influence of the TDC routing scheme; Secondly, the positioning of the TDC has direct influence on the channel performance and characteristics, as reported in [23], where two similar TDC channels present delay differences larger than 200 ps on the same step.

The work by Chaberski et al. [25] proposed a method to improve the linearity of a TDL implemented using Carry chains, by regulating the wire load at each step. A different approach has been reported by Zhang et al. [26], demonstrating that the routing resources on a FPGA can be used as delay elements. Thus, one can state that the performance of the architecture presented in [23] can be improved if the routing is carefully planned. This Section will analyze and discuss the implementation process of the proposed gray code architecture in order to achieve better linearity performance and superior homogeneity between different TDC channels.

A. Gray code Oscillator

The 5-bit gray counter TDC channel has a maximum of 32 interpolation steps. Considering an average step delay of 250 ps for a 7-Series Xilinx FPGA (based on the results reported in [23]), a maximum time interval of 8 ns can be covered by the TDC channel. Therefore, a system clock of at least 125 MHz must be used. To validate the step propagation delay, a preliminary implementation of one channel was made using a 200 MHz system clock and a code density test was performed. For the FPGA used in this work, a Xilinx ZYBO Z7010, an average step delay of 308 ps was obtained. Thus, in order to reduce power consumption, for the final system implementation, the system clock was adjusted to 125 MHz, since for the average step delay obtained, the 32 TDC channel steps can cover more than one clock period.

B. Gary's Counter oscillator Datapath Analysis

Even though the truth table being implemented by a LUT does not affect its propagation delay [24], it does not hold true when considering the LUT's fanout. Unfortunately, the propagation delays obtained from the simulation tool are based on the best- or worst-case scenarios, making it harder to predict the timings at the typical operation point. Although not totally accurate, in this work the authors assumed that by securing a uniform routing delay on the worst-case scenario, at typical operation conditions, the delays' deviations would stay similar. Moreover, as the TDC channel can be confined to a single Configurable Logic Blocks (CLB), the voltage and temperature conditions should be similar in all the 5 LUTs used to build the gray code oscillator. Therefore, exploring the routing possibilities during the layout of the gray code TDC may lead to an implementation with higher linearity with no extra hardware cost.

Further analysis to the pattern on the 5-bit gray code lead to the conclusion that only 8 out of the 24 Datapath connections affect the size of the steps, namely the path from the output of LUT0 to the input of all the other LUTs (4 connections) and the output of all the other LUTs to the inputs of LUT0 (another 4 connections) (see Table I and Fig. 2). Only the first 17 steps are depicted in Table I, since the same propagation delay pattern was registered after the 16th step. This greatly reduces the implementation effort of the linearity correction through routing.

C. Manual Routing to Control Datapath's delay

The manual routing process can be performed in Vivado design tool, using the implemented design graphical interface, that will create a set of physical constraints annotated in a XDC (Xilinx Design Constraints) file. This allows for different routing options to be explored in order to achieve the desired net delay. To be able to fix the manual routing done on a pure combinatorial path, first the LUTs need to be fixed with placement constraints and its inputs need to be locked. Otherwise the tool may change the order of the input ports from one run to the other, leading to scenarios where it is impossible to respect the defined routing constraints [27]. The lowest_net_delay option was used as the starting point for the manual routing implementation. The nets' delay timings were annotated and are presented in TABLE II under the Automatic Routing column. This will ultimately result in a higher

average step size, but the linearity is increased (as it is demonstrated in Section IV-A). Therefore, manual routing will always be a trade-off between resolution and linearity. TABLE II presents the timing differences before and after the application of manual routing to the gray's code oscillator Datapath on the stop channel (based on the worst-case timing models). Only the 8 previously identified paths were changed (marked in bold).

D. Gray Code Counter Timing Simulation

To test the proper behavior of the implemented architecture, a set of simulations were developed. The simulation consists of a set of hit signals generated in sequence at random instants. Since one of the core modules behavior is based on the propagation time of the FPGA logic, timing simulations must be performed.

Analyzing the timing simulation results of the gray code oscillator start channel in detail, it is possible to verify that the gray sequence is being correctly generated (see Fig. 5). As expected, the delay of each step varies and repeats itself in the previously identified pattern (in Table I). Thus, if the routing timings (extracted from the timing reports generated by Vivado after completing the implementation phase) is subtracted to the times obtained from the simulation, the worst-case scenario of the LUT's propagation delay can be calculated according to (1).

$$t_{LUTi}=t_{PD}-t_{ROUTEi} \quad (1)$$

where t_{LUTi} is the LUT propagation delay, t_{ROUTEi} is the propagation delay of the LUT's output wire to the LUT which will change state for the next count, and t_{PD} is the total propagation delay, extracted from the simulation.

From the simulated results, it is possible to verify that the routing resources appear as the main contribution for the delay of the counter steps, thus validating that controlling the routing will contribute to improve the overall linearity of the architecture.

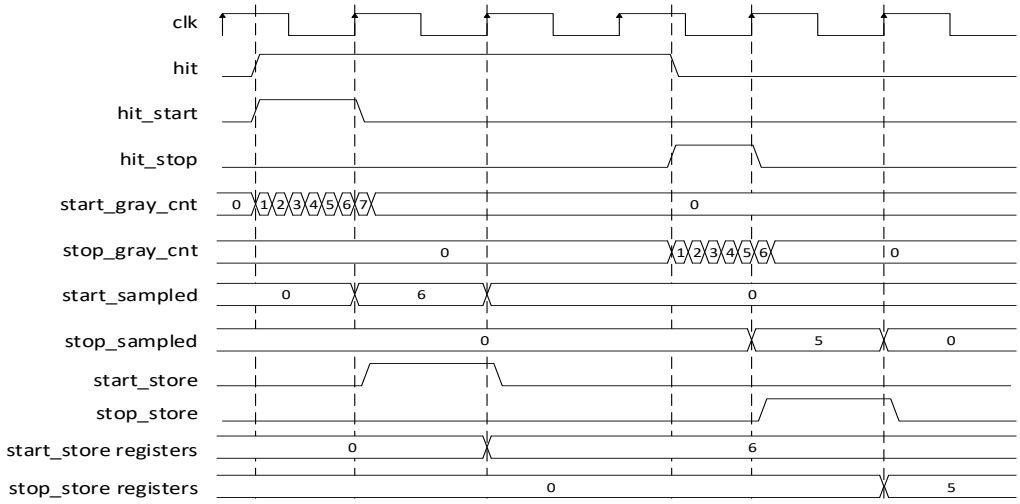

Fig. 3. Gray code oscillator TDC operation principle waveform

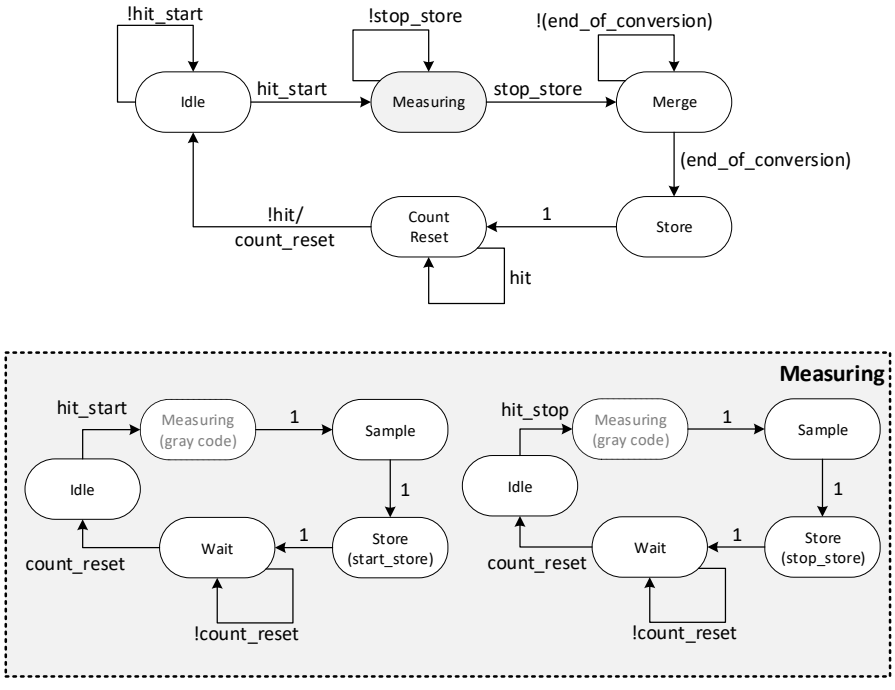

Fig. 4. TDC IP state machine (Top) and start and stop gray code oscillator channels state machines (bottom)

TABLE I. GRAY CODE DATAPATH DELAY ANALYSIS

Current Gray Value					Next Gray Value					Propagation delay
0	0	0	0	0	0	0	0	0	1	Tphit + tpLUT0
0	0	0	0	1	0	0	0	1	1	Tpbit0 + tpLUT1
0	0	0	1	1	0	0	0	1	0	Tpbit1 + tpLUT0
0	0	0	1	0	0	0	1	1	0	Tpbit0 + tpLUT2
0	0	1	1	0	0	0	1	1	1	Tpbit2 + tpLUT0
0	0	1	1	1	0	0	1	0	1	Tpbit0 + tpLUT1
0	0	1	0	1	0	0	1	0	0	Tpbit1 + tpLUT0
0	0	1	0	0	0	1	1	0	0	Tpbit0 + tpLUT3
0	1	1	0	0	0	1	1	0	1	Tpbit3 + tpLUT0
0	1	1	0	1	0	1	1	1	1	Tpbit0 + tpLUT1
0	1	1	1	1	0	1	1	1	0	Tpbit1 + tpLUT0
0	1	1	1	0	0	1	0	1	0	Tpbit0 + tpLUT2
0	1	0	1	0	0	1	0	1	1	Tpbit2 + tpLUT0
0	1	0	1	1	0	1	0	0	1	Tpbit0 + tpLUT1
0	1	0	0	1	0	1	0	0	0	Tpbit1 + tpLUT0
0	1	0	0	0	1	1	0	0	0	Tpbit0 + tpLUT4
1	1	0	0	0	1	1	0	0	1	Tpbit1 + tpLUT0
1	1	0	0	1	1	1	0	1	1	Tpbit0 + tpLUT1

tpLUTx = propagation time inside LUTx from the moment its input change until its output gets updated at the input of the sampling registers; Tpbitx = propagation time from the output pin of the LUTx to the input pin of the next LUT to change. (x = [0-4])

TABLE II. ROUTING DELAYS FOR THE GRAY CODE OSCILLATOR TDC DATAPATH

Stop Channel	Automatic Routing (default)					Manual Routing				
	LUT4	LUT3	LUT2	LUT1	LUT0	LUT4	LUT3	LUT2	LUT1	LUT0
bit0	**664**	**701**	**696**	**295**	**-**	**665**	**691**	**686**	**876**	**-**
bit1	700	198	193	475	**477**	700	198	193	475	**477**
bit2	909	730	735	1080	**165**	911	732	737	162	**580**
bit3	394	306	307	711	**709**	394	306	307	711	**709**
bit4	297	612	609	330	**514**	296	916	914	518	**513**

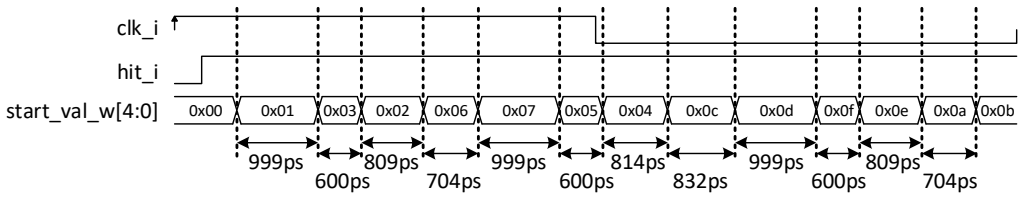

Fig. 5. Detailed view of the gray code sequence generation and step size

IV. EXPERIMENTAL RESULTS

To proper test the proposed solution, it is not enough to compare the results obtained by the architecture proposed in this work with the ones presented in [23], since in different platforms, the logic elements propagation delay may differ, not allowing for a fair comparison. The proposed architecture was deployed in Xilinx ZYBO Z7010 development board using two different design flows to study the effect of routing on the architecture performance and the key performance parameters were measured for each scenario. The first design flow adopted was similar to the one used in [23], where only manual placement has been used. In the second approach, manual routing was applied to the proposed architecture implementation, and the design re-implemented with the new routing constraints. To validate the scalability of the method, the routing constraints defined for the start channel were copied to the stop channel directly, without going through the manual routing process. The Tektronix AFG1022 arbitrary waveform generator was used to generate the time intervals to assess the developed TDC. According to the manufacturer datasheet [28], the used waveform generator has a typical jitter of less than 1 ns. This rather high jitter, although not having high influence in the results from the code density and linearity tests, has a negative impact on the system's measured single-shot precision.

The TDC IP was connected to the integrated Arm processor (used to read the values from the TDC and perform the interface to the host computer) using an AXI-4 Lite interface. A code density test was performed to extract the delay of each step of the fine interpolation stage. The results from this test were also used to calculate the non-linearities of the fine measurement module, namely, the Differential Non-Linearity (DNL) and Integral Non-Linearity (INL). A total of 100 thousand measurements were performed to reduce probabilistic errors. A single-shot precision test was also completed with 100 thousand samples. To understand the impact of a calibration mechanism in the TDCs' performance, a post-measurement software bin-by-bin calibration was applied to the 100 thousand samples collected during the single-shot precision test. The calibration tables were created based on the results from the code density test. Then, the single-shot precision was recalculated after applying the calibration. All the tests were performed with the development board at ambient temperature of 25°C.

A. Code Density Test

To perform the code density test, the waveform generator was configured to output a square wave signal at a frequency unrelated with the 125 MHz reference clock used, thus creating a sliding window effect on the sampling steps of the TDC. The selected frequency was 999133 Hz, which results

in a time interval of approximately 480.7 ns (verified using an oscilloscope).

The code density test results for both implementations are presented in Fig. 6. Since a 125 MHz reference clock was used, the average step delay, for the manual routing implementation, is 380.9 ps (according to (2), where τ_i is the gray code oscillator step delay, N_i is the number of times the i^{th} step was sampled, T is the sampling clock period and N_{total} is the total number of samples performed). This represents a 33.1 ps increase regarding the solution where only the placement is constrained (when comparing the results obtained from the deafult and manual routing implementation of the start channel TDC. When comparing to the stop channel TDC a 72.9 ps average step delay increase is verified). Another important factor to notice is the delay uniformity across channels. When manual routing is performed, the start and stop channels steps' delays are closely matched with a maximum difference of 95 ps in the same step (for the worst-case scenario). When analyzing the automatic routing implementation, a maximum difference of 295 ps is observable.

$$\tau_i = N_i * \frac{T}{N_{total}} \quad (2)$$

B. Linearity Tests

Based on the information of each step propagation delay the DNL can be calculated according to (3):

$$DNL_i = \tau_i - \tau \quad (3)$$

where DNL_i is the DNL of the i^{th} step and τ is the average step size. The INL is the accumulated non-linearity across the TDC measurement range [1]. Therefore, it can be calculated by adding all the DNL contributions according to (4)

$$INL_n = \sum_{i=0}^{n} DNL_i \quad (4)$$

The DNL and INL calculated from the code density test results are presented in Fig. 7. The results are presented normalized to one LSB, which for the case of the manual routing implementation is 380.9 ps and for the default routing implementation 348 ps.

As expected, the manual routing scenario shows superior linearity with a maximum DNL of 0.38 LSB (106.7 ps), while in the automatic routing approach, a maximum DNL of 0.6 LSB (208.8 ps) is observable, representing a 21.7% improvement. The INL is also improved with the manual routing approach reporting an INL in the range of 0.01 and 0.7 LSB for the start channel (worst case), representing a 71.4% improvement since the INL in the default routing implementation reached almost 3 LSB.

C. Single-Shot Precision

The single-shot precision tests were performed using the same time interval defined for the code density test. The results from the 100 thousand measurement made are presented in Fig. 8, for both calibrated and non-calibrated tests. When comparing both approaches, a 110 ps precision improvement can be observed when manual routing is used. However, the authors believe that the precision results are being influenced by the jitter of the waveform generator used. One indicative supporting this theory is the results obtained after calibration is applied. In the case of manual routing, applying calibration to the preformed measurements has no effect on the TDC's precision. Although manual routing enabled higher linearity to be achieved, it was expected that a bin-by-bin calibration would be able to improve the TDC's precision to at least 0.5 LSB. Thus, further tests must be done with a different waveform generator, in order to have a better characterization of the advantages of using manual routing on precision. Nevertheless, the achieved results are promising and able to demonstrate the advantages of the implementation of the gray code oscillator TDC using manual routing. The trade-off is purely in the design stage which require an additional manual step, increasing the TDC channel implementation complexity, and on the TDC average resolution step.

D. Discussion

When comparing the obtained result with the results presented in [23] (summarized in TABLE III) it is possible to conclude that the presented approach offers better linearity and superior homogeneity across multiple channels, while maintaining a low resource usage. As already explained, the worse single-shot precision reported was due to the influence of the waveform generator's jitter. When implementing multiple TDC channel in a complex system, the allocated routing resources on each TDC channel can change significantly due to the channels positioning and proximity to other system's blocks inside the FPGA. Since the constraints created to a single TDC channel can be applied on different channels directly, system scalability with channel homogeneity can be assured. Only two CLBs are required to implement a single TDC channel, thus the proposed architecture is suitable for multichannel applications, even on small, low-cost FPGAs. Furthermore, the same calibration mechanism, for instance a single bin-by-bin calibration table, can be deployed to calibrate multiple TDC channels due to its similar delays' distribution, enabling a considerable resource and power savings.

The presented architecture has been migrated to a Xilinx UltraScale+ FPGA platform. The preliminary test performed, without applying manual routing, showed an average step delay of 105-118 ps with a DNL in the range of [-0.4:0.87] LSB. The more complex routing resources schema on the UltraScale+ FPGAs makes the manual routing linearization process harder.

Even though the proposed TDC does not achieve state-of-art resolution, the proposed architecture's small number of resources utilization, high linearity and single-shot precision offers a very interesting trade-of, suitable for applications like mobile and wearables, where low area and power requirements are prioritized.

V. Conclusion

This work presents a novel TDC architecture focused on low resources and power consumption, suitable for applications where multiple channels are required. The architecture based on the gray code oscillator and inspired by the wire load model principle achieved a resolution of 380.9 ps with very low resources consumption (16 logic elements per channel), when implemented in a Xilinx ZYBO Z7010 FPGA. It also enables the implementation of multiple TDC channels with closely match characteristics, which opens the possibility for a single calibration circuit, shared among multiple channels, further increasing the hardware resources optimization and decreasing power consumption. The control of the routing resources used to define the Datapath of the gray counter steps enable an improvement of 21.7% on the TDC channel differential non-linearity and a 70.4% improvement on the TDC channel integral non-linearity

(considering the peak-to-peak values in picoseconds), reducing the need for calibration. A 27.5% single-shot precision improvement was also achieved without any calibration mechanism. These results point out to the conclusion that, regarding linearity, the routing is the main impact source and not the cell propagation delay differences due to process mismatch. Future work will focus on the study of the effect of temperature on the multiple channels implemented to check how it will affect the proposed architecture performance. The effect of manual routing on reducing the TDC Channel thermal dependency will also be targeted in future research. Finally, since the preliminary tests from the porting of the proposed architecture to Xilinx UltraScale+ FPGAs showed promising results, the routing resources are currently under study and the results of applying manual routing to the proposed architecture on UlttraScale+ FPGAs will be reported in future works.

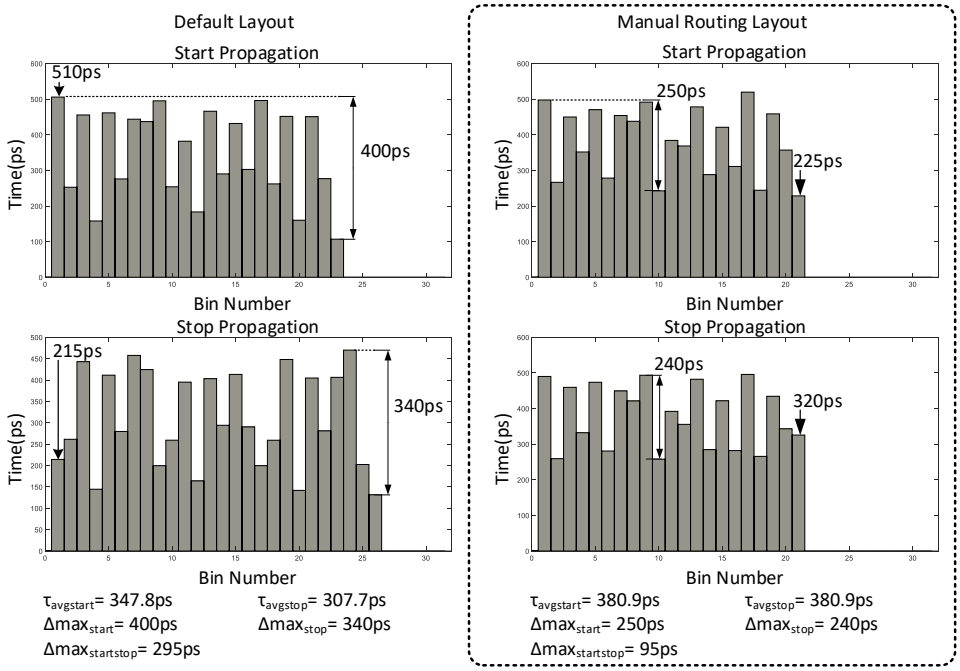

Fig. 6. Gray code TDC start (top) and stop (bottom) channels code density test

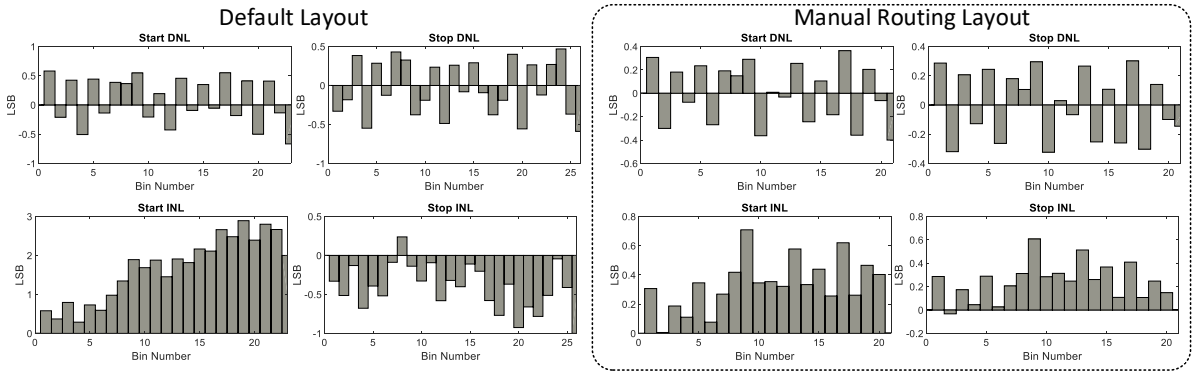

Fig. 7. Gray code TDC linearity results

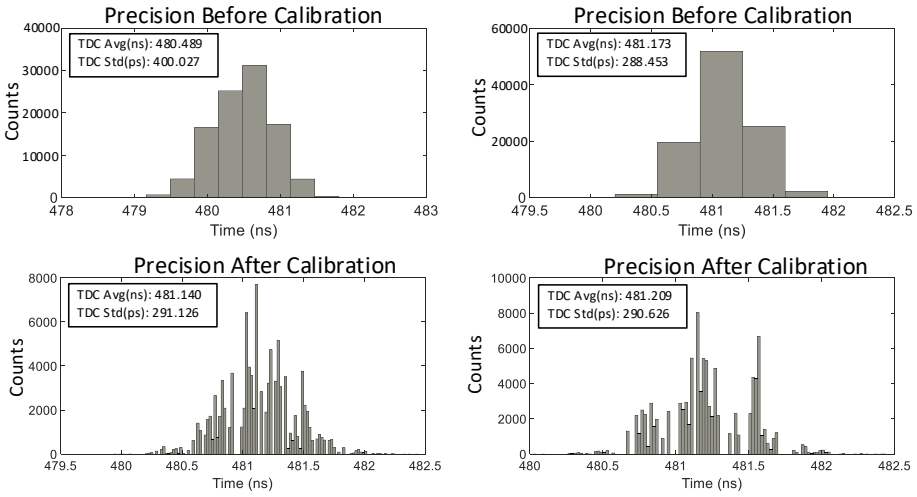

Fig. 8. Gray code TDC single-shot precision for default (left) and manual (right) routing

TABLE III. COMPARISON BETWEEN THE DIFFERENT GRAY CODE OSCILLATOR TDC IMPLEMENTATIONS

Architecture	Gray code Oscillator [23]	This Work (Xilinx ZYBO Z7010)	
		Default Routing	Manual Routing
LSB (ps)	256-271	308-348	380.9
DNL (LSB)	[-0.61:0.95]	[-0.65:0.55]	[-0.38: 0.38]
INL (LSB)	-	[0.01:2.9]	[0.01:0.7]
Single-Shot Precision (ps)	160 64**	400 290*	290 290*
Missing Codes	No	No	No

*After bin-by-bin Calibration; **After 4 measurement average Calibration.

ACKNOWLEDGMENTS

This work was supported by a Portuguese Scholarship from FCT - Fundação para a Ciência e Tecnologia and Bosch Car Multimedia, under the Advanced Engineering Systems for Industry (AESI) doctoral program. (Scholarship ID: PDE/BDE/114562/2016) and the European Structural and Investment Funds in the FEDER component, through the Operational Competitiveness and Internationalization Programme (COMPETE 2020) [Project nº 037902; Funding Reference: POCI-01-0247-FEDER-037902].

REFERENCES

[1] Y. Wang, J. Kuang, C. Liu, and Q. Cao, "A 3.9-ps RMS Precision Time-to-Digital Converter Using Ones-Counter Encoding Scheme in a Kintex-7 FPGA," *IEEE Trans. Nucl. Sci.*, vol. 64, no. 10, pp. 2713–2718, Oct. 2017.

[2] Y. Wang and C. Liu, "A 3.9 ps Time-Interval RMS Precision Time-to-Digital Converter Using a Dual-Sampling Method in an UltraScale FPGA," *IEEE Trans. Nucl. Sci.*, vol. 63, no. 5, pp. 2617–2621, 2016.

[3] J. Torres *et al.*, "Time-to-Digital Converter Based on FPGA With Multiple Channel Capability," *Nucl. Sci. IEEE Trans.*, vol. 61, no. 1, pp. 107–114, 2014.

[4] A. T. Eshghi, S. Lee, M. K. Sadoughi, C. Hu, Y.-C. Kim, and J.-H. Seo, "Generic high resolution PET detector block using 12×12 SiPM array," *Smart Mater. Struct.*, vol. 26, no. 10, p. 105037, Oct. 2017.

[5] A. Aguilar *et al.*, "Time of flight measurements based on FPGA and SiPMs for PET-MR," *Nucl. Instruments Methods Phys. Res. Sect. A Accel. Spectrometers, Detect. Assoc. Equip.*, vol. 734, no. PART B, pp. 127–131, 2014.

[6] R. Szplet, P. Kwiatkowski, K. Różyc, Z. Jachna, and T. Sondej, "Picosecond-precision multichannel autonomous time and frequency counter," *Rev. Sci. Instrum.*, vol. 88, no. 12, p. 125101, Dec. 2017.

[7] R. Machado, J. Cabral, and F. S. Alves, "Recent Developments and Challenges in FPGA-Based Time-to-Digital Converters," *IEEE Trans. Instrum. Meas.*, vol. 68, no. 11, pp. 4205–4221, Nov. 2019.

[8] R. Szplet, Z. Jachna, P. Kwiatkowski, and K. Rozyc, "A 2.9 ps equivalent resolution interpolating time counter based on multiple independent coding lines," *Meas. Sci. Technol.*, vol. 24, no. 3, p. 035904, Mar. 2013.

[9] R. Szplet, P. Kwiatkowski, Z. Jachna, and K. Rozyc, "An eight-channel 4.5-ps precision timestamps-based time interval counter in FPGA chip," *IEEE Trans. Instrum. Meas.*, vol. 65, no. 9, pp. 2088–2100, 2016.

[10] V. Hiligsmann, "How sensor technology will shape the cars of the future," 2017. [Online]. Available: https://www.melexis.com/en/insights/knowhow/how-sensor-technology-shape-cars-future. [Accessed: 22-Jul-2019].

[11] M. Khader and S. Cherian, "An Introduction to Automotive LIDAR." Texas Instruments, 2018.

[12] V. Milanović, A. Kasturi, J. Yang, and F. Hu, "Closed-loop control of gimbal-less MEMS mirrors for increased bandwidth in LiDAR applications," 2017, p. 101910N.

[13] N. Druml, I. Maksymova, T. Thurner, D. van Lierop, M. Hennecke, and A. Foroutan, "1D MEMS Micro-Scanning LiDAR," in *International Conference on Sensor Device Technologies and Applications (SENSOR-DEVICES)*, 2018.

[14] C. Liu, Y. Wang, P. Kuang, D. Li, and X. Cheng, "A 3.9 ps RMS resolution time-To-digital converter using dual-sampling method on Kintex UltraScale FPGA," *2016 IEEE-NPSS Real Time Conf. RT 2016*, pp. 1–3, 2016.

[15] Y. Wang and C. Liu, "A 4.2 ps Time-Interval RMS Resolution Time-to-Digital Converter Using a Bin Decimation Method in an UltraScale FPGA," *IEEE Trans. Nucl. Sci.*, vol. 63, no. 5, pp. 2632–2638, 2016.

[16] Q. Shen *et al.*, "A 1.7 ps equivalent bin size and 4.2 ps RMS FPGA TDC based on multichain measurements averaging method," *IEEE Trans. Nucl. Sci.*, vol. 62, no. 3, pp. 947–954, 2015.

[17] Y. Jia, C. Wang, and H. Shi, "Multi-channel high precision time digital converter system based on equivalent pulse counting," in *2018 Chinese Control And Decision Conference (CCDC)*, 2018.

[18] Z. Li *et al.*, "Development of an integrated four-channel fast avalanche-photodiode detector system with nanosecond time resolution," *Nucl. Instruments Methods Phys. Res. Sect. A Accel. Spectrometers, Detect. Assoc. Equip.*, vol. 870, no. November 2016, pp. 43–49, 2017.

[19] T. Xiang *et al.*, "A multi-phase clock time-to-digital convertor based on ISERDES architecture," *Proc. - 2014 IEEE 22nd Int. Symp. Field-Programmable Cust. Comput. Mach. FCCM 2014*, no. 1, p. 35, 2014.

[20] Y. Wang, P. Kuang, and C. Liu, "A 256-channel multi-phase clock sampling-based time-to-digital converter implemented in a Kintex-7 FPGA," in *Conference Record - IEEE Instrumentation and Measurement Technology Conference*, 2016, vol. 2016-July.

[21] H. Chen and D. D.-U. Li, "Multichannel, Low Nonlinearity Time-to-Digital Converters Based on 20 and 28 nm FPGAs," *IEEE Trans. Ind. Electron.*, vol. 66, no. 4, pp. 3265–3274, Apr. 2019.

[22] K. Cui, Z. Liu, R. Zhu, and X. Li, "FPGA-based high-performance time-to-digital converters by utilizing multi-channels looped carry chains," in *2017 International Conference on Field Programmable Technology (ICFPT)*, 2017, pp. 223–226.

[23] J. Wu and J. Xu, "A Novel TDC Scheme: Combinatorial Gray Code Oscillator Based TDC for Low Power and Low Resource Usage Applications," in *2019 5th International Conference on Event-Based Control, Communication, and Signal Processing (EBCCSP)*, 2019, pp. 1–7.

[24] Xilinx, "7 Series FPGAs Configurable Logic Block: User Guide." Xilinx, 2016.

[25] D. Chaberski, R. Frankowski, M. Zieliński, and Ł. Zaworski, "Multiple-tapped-delay-line hardware-linearisation technique based on wire load regulation," *Measurement*, vol. 92, pp. 103–113, Oct. 2016.

[26] M. Zhang, H. Wang, and Y. Liu, "A 7.4 ps FPGA-based TDC with a 1024-unit measurement matrix," *Sensors (Switzerland)*, vol. 17, no. 4, 2017.

[27] XIlinx, "Vivado Design Suite User Guide: Using Constraints," 2018.

[28] Tektronix, "Arbitrary/Function Generator AFG1000 Series Datasheet." Tektronix, 2016.