electronics

MDPI

Article

5.7 ps Resolution Time-to-Digital Converter Implementation Using Routing Path Delays

Roza Teklehaimanot Siecha [1,*], Getachew Alemu [2], Jeffrey Prinzie [3] and Paul Leroux [3]

[1] Department of Electrical and Computer Engineering, Addis Ababa Science and Technology University, Addis Ababa P.O. Box 16417, Ethiopia
[2] Department of Electrical and Computer Engineering, Addis Ababa Institute of Technology, Addis Ababa University, Addis Ababa 1178, Ethiopia; getachew.alemu@aait.edu.et
[3] Department of Electrical Engineering, Faculty of Engineering Technology, Katholieke Universiteit Leuven, 3000 Leuven, Belgium; jeffrey.prinzie@kuleuven.be (J.P.); paul.leroux@kuleuven.be (P.L.)
* Correspondence: rozateklehaimanot.siecha@student.kuleuven.be; Tel.: +251-947626583

Abstract: A tapped delay line (TDL)-based time-to-digital converter (TDC) implemented on an FPGA (Field Programmable Gate Array) is sensitive to nonlinearities because of significant variations in the delay of the delay elements. Most of the nonlinearity of FPGA-based TDCs comes from the routing of the design. It is promising to realize TDCs using internal routing resources available in FPGAs, as these devices contain a lot of routing resources and are resistant to voltage and temperature changes. This work implements and tests a TDC based on a series of counters driven by a variable delay line that exploits the internal routing resources available in the FPGA as delay elements. A manual placement and routing technique that results in greater resolution and linearity is proposed. The time-interleaving concept is used to improve the resolution of the TDC. A measurement matrix with 512 and 1024 parallel counters is implemented on a Zynq Evaluation and Development (ZED) board. The result of the 1024-unit TDC showed that a dynamic range of 93.6 ns can be measured using a 4-bit coarse gray code counter running at a reference frequency of 171 MHz, and a resolution of 5.7 ps is achieved. The implemented TDC is low-cost, has a fast time to market, and it benefits from the abundant routing resources in the FPGA.

Keywords: tapped delay line (TDL); time to digital converter (TDC); field programmable gate arrays (FPGAs); delay elements

Citation: Siecha, R.T.; Alemu, G.; Prinzie, J.; Leroux, P. 5.7 ps Resolution Time-to-Digital Converter Implementation Using Routing Path Delays. *Electronics* **2023**, *12*, 3478. https://doi.org/10.3390/electronics12163478

Academic Editor: Raffaele Giordano

Received: 25 July 2023
Revised: 12 August 2023
Accepted: 14 August 2023
Published: 17 August 2023

1. Introduction

A time-to-digital converter (TDC) converts an input, which is a time interval between a start and a stop event, to a digital number [1–4]. TDCs can be realized in an analog or a digital form. A complementary metal–oxide semiconductor (CMOS) process has been predominant over analog architecture implementations because it is easy to fabricate and can easily be integrated with Very Large-Scale Integration (VLSI) designs. All digital TDCs can be realized as fully customizable Application-Specific-Integrated-Circuit (ASIC)-based circuits or as reconfigurable Field-Programmable-Gate-Array (FPGA)-based circuits. ASIC-based TDCs [5–7] can be designed to achieve better resolution because they are fully customizable, but they take a long time to develop and are costly compared to FPGA-based TDCs. Moreover, it is hard to reconfigure an ASIC-based TDC once it is designed. FPGA-based TDCs are low-cost, configurable, and fast to develop [2,8–10]. The intrinsic propagation delay of FPGA-based TDCs, which is set and cannot be modified by the designer, restricts the resolution and is one of the issues they face.

A comprehensive literature review on different topologies of time-to-digital converters (TDCs) is presented in [11]. The simplest technique to realize a digital TDC is to use the counter-based approach. Other TDC topologies use the time interpolation technique. These TDCs include delay line-based TDCs, also known as Tapped Delay Line (TDL) TDCs

or Flash TDCs [8,12–14], Vernier delay line TDCs utilized by designers to improve the resolution beyond the cell delay [10,15–17], Delay Locked Loop (DLL) TDC [7,18] or wave union TDC [19–21].

The delay lines are commonly based on carry chains [4,8,12,19], logic gates, buffers or inverters [22,23] available in the FPGA. Due to their comparatively high linearity, carry chain delay lines are frequently used by designers. The carry logic is a dedicated logic element in the FPGA, with its routing path aimed at performing fast arithmetic and logic operations.

Studies [2,24] show that most of the delay non-linearity comes from the routing path delay in the FPGA. The automatic placer and router provide unpredictable routing paths in the implementation [25]. These routing path delays found in the FPGA have been used as delay elements in some research [2,24,26]. Wires, switch boxes, and Programmable Interconnect Points (PIPs), which make up more than 80% of the space of an FPGA, provide plentiful routing resources. Additionally, since routing paths are insensitive to variations in voltage and temperature, realizing TDCs utilizing internal routing resources as delay elements improves measurement stability.

The general architecture of Xilinx FPGAs consists of two-dimensional arrays of CLBs (Configurable Logic Blocks) with programmable routing wires placed in vertical and horizontal directions between CLBs [25,27,28]. This is depicted in Figure 1. The routing resources in the FPGA's routing architecture include switch boxes (SBs), connection boxes (CBs), and wires in vertical and horizontal tracks. CBs connect the horizontal and vertical wires with the CLB input and output pins. Switch boxes allow wires switching from horizontal routing channels to vertical ones and vice versa. The placement of logic gates in the FPGA is identified by X and Y coordinates. Slices, SBs, CBs, and wires have their corresponding XY coordinates.

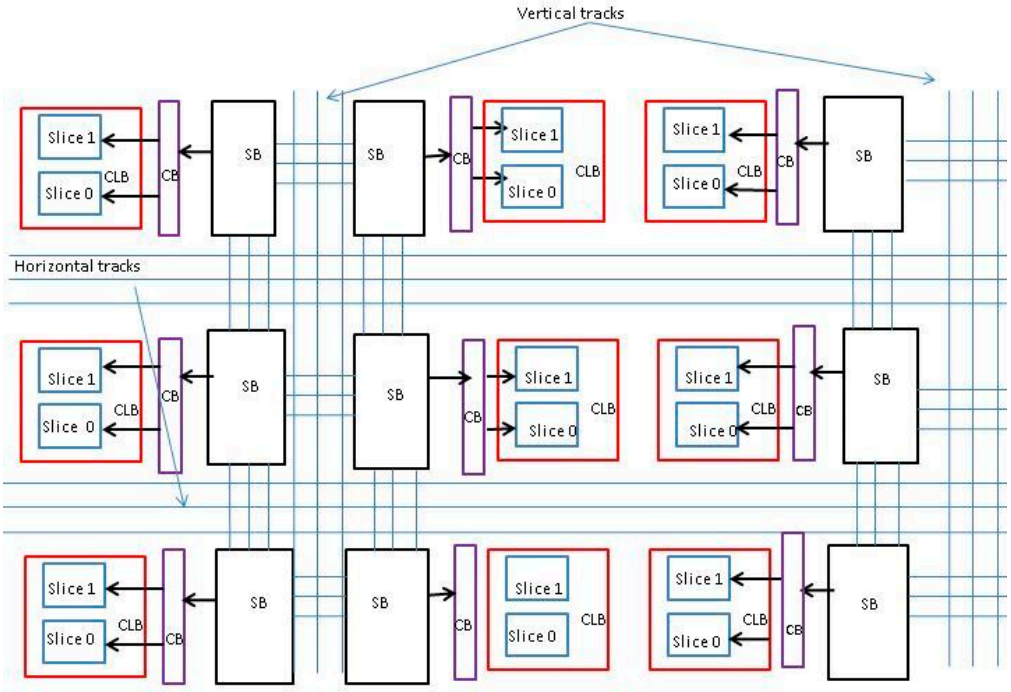


Figure 1. Field Programmable Gate Array routing architecture.

By forgoing the special delay elements inherent in FPGAs and employing internal routing resources as delay elements, a 9 ps resolution Vernier delay line TDC is built in [24]. In [2], a new TDC design based on a counter matrix is described. A resolution of 7.4 ps is attained by arranging the counters in a rectangle matrix and the driver outside the TDC matrix.

Although the architecture of the TDC in our work is based on the methods described in [2], our study shows that without knowing how the FPGA's automatic placer and router

work, it takes a lot of trial and error to identify the best driver location that offers good path delay linearity and resolution. This work examines the ZedBoard FPGA's [29] routing architecture, and a manual placement and routing method is proposed to achieve high-resolution TDC. Our research, which, as far as we know, has not been conducted on a ZedBoard FPGA, enhances the TDC's resolution when compared to earlier studies [2,24] that have used comparable techniques. The findings of this investigation also demonstrate the tradeoffs between the path delay linearity, resolution, power consumption, and resource utilization of TDC.

This paper is organized into four sections. Section 2 presents the proposed architecture of the TDC, the type of FPGA used, and its routing architecture; Section 3 explains the experimental setup and simulation results, and compares the obtained results to earlier investigations. This paper is concluded in Section 4 with a summary of the findings and suggestions for further research.

2. Methodology

2.1. Principle and Architecture of the Implemented TDC

Using a counter is the most straightforward method of TDC implementation on FPGA. A counter counts with a clock period equal to the reference clock's cycle time to convert a time interval from a START to a STOP event to a digital output [2,4,26]. Figure 2 illustrates the operation of a counter-based TDC.

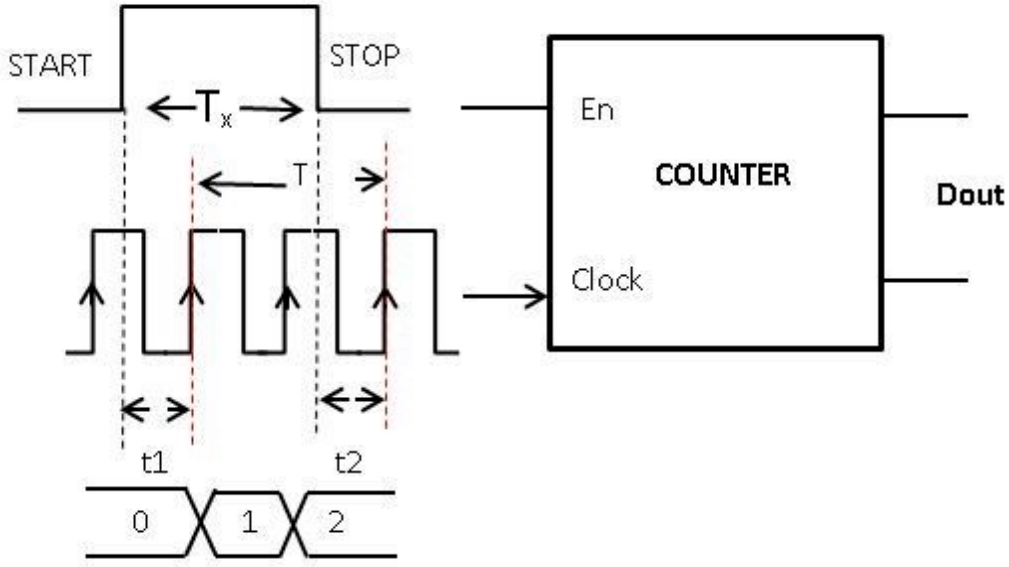


Figure 2. Timing diagram for counter method implementation of time-to-digital converter.

The time interval we want to measure in Figure 2 is T_x and an enable signal is generated according to T_x. The counter begins to run when the START signal rises, therefore it counts using the reference clock's cycle time. The counter is disabled at the edge of the STOP signal, and it ceases to count. Given that the counter counts to N_{ref} with a reference clock frequency of f_{ref}, the coarse time measured by the counter (T) is mathematically defined in Equation (1). The time measurement error (ΔT) is calculated as in Equation (2).

$$T = \frac{N_{ref}}{f_{ref}} \tag{1}$$

$$\Delta T = T - T_x = t_2 - t_1 \tag{2}$$

A counter-based TDC has the benefit of being easy to deploy and having a wide measuring range. The measuring range or dynamic range of the TDC is increased with an overhead of area by raising the counter's bit count. However, resolution is constrained by

the counter's clock frequency. The counter should be derived using a high-speed reference clock to obtain a fine resolution. As a result, in multistage converters, counter-based TDC implementation is typically realized as a coarse TDC.

In paper [2], a novel method for enhancing the resolution of a TDC on an FPGA is proposed that utilizes parallel counters in a delay line. Figure 3 shows the structure of the TDC implementation using a matrix of counters.

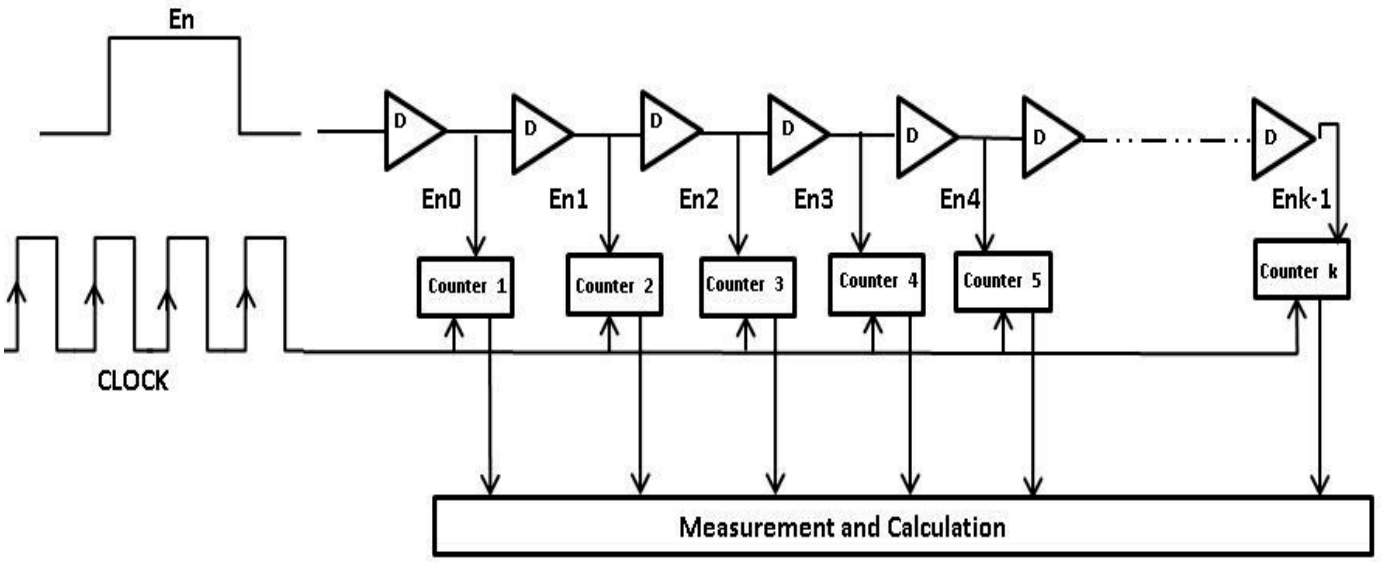


Figure 3. Design of a time-to-digital converter using a counter-based delay line.

A delay line composed of a series of delay elements, each having a delay value equal to D, is compelled to pass an enable signal, En. The subsequent counters with a D time difference are enabled by this enable signal. The parallel counters are driven by the same clock signal. At the falling edge of the enable signal, the output of the counters is sampled. The time measured (T_x) and the measurement error (ΔT_x) for K parallel counters can be calculated using Equations (3) and (4) respectively.

$$T_x = \frac{\frac{1}{K}\sum_{i=0}^{K-1} N_{ref,i}}{f_{ref}} \tag{3}$$

$$\Delta T_x = \frac{1}{K}\sum_{i=0}^{K-1} \Delta T_{x,i} = \frac{1}{K}\sum_{i=0}^{K-1} (t_{2,i} - t_{1,i}) \tag{4}$$

where $N_{ref,i}$ is the output of the ith counter, $\Delta T_{x,i}$ is measurement error associated with the ith counter, $t_{2,i}$ and $t_{1,i}$ are the fine times t_2 and t_1 of the ith counter.

The timing diagram of the counter based on delay line TDC is presented in Figure 4. In the figure, the initial enable signal and its delayed versions after being processed by the delay elements are displayed. The counter output can be K, $K + 1$, or $K - 1$. The phase difference between N parallel counters and the reference counter is -1, 0 or +1. These scenarios can be represented using only two bits. Therefore, to reduce resource usage as indicated in [2], only one coarse n-bit counter and N $-$ 1 two-bit counters are used.

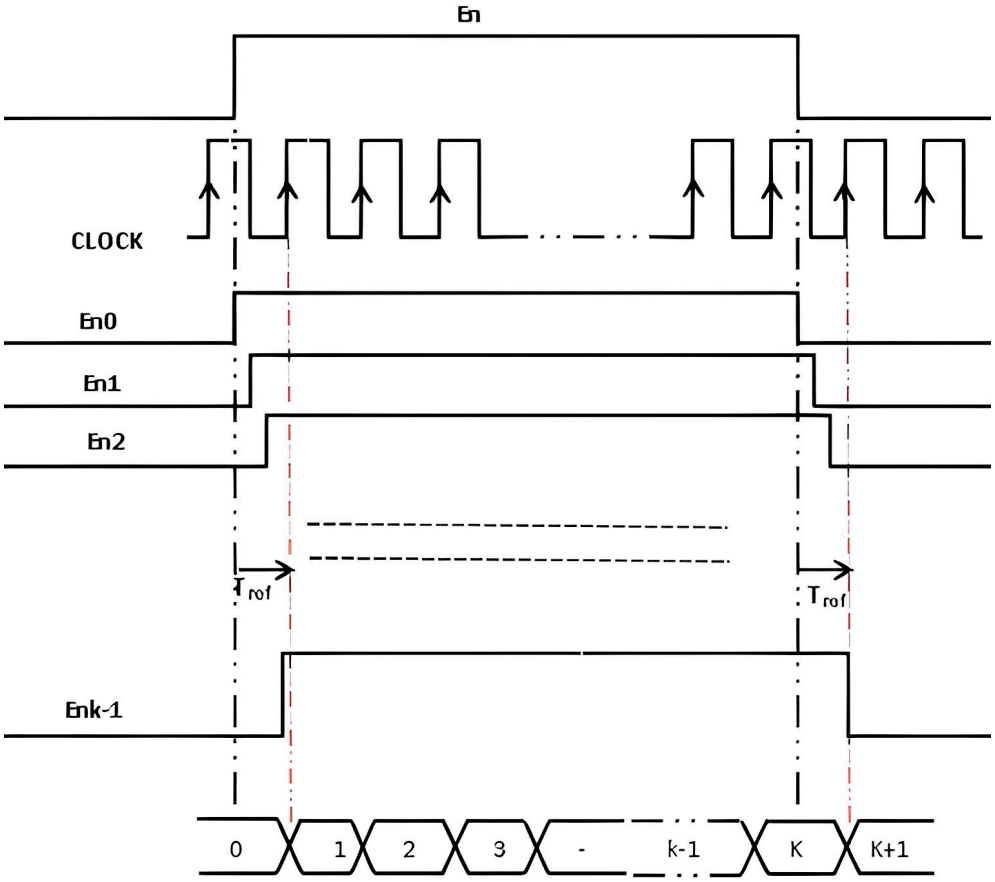


Figure 4. Timing diagram of counter based on delay line time-to-digital converter.

The counter based on the delay line TDC for linear measurement can only be constructed if all of the delay elements have the same delay D as represented in Equation (5). Another difficulty is that the full delay line's overall delay must equal one reference clock period.

$$D = \frac{1}{K * f_{ref}} \tag{5}$$

2.2. Enable Signal Generation

ZedBoad FPGA [29] is used for the experimental setup and testing, as is explained in detail in Section 3. In the proposed TDC architecture, an SR latch Programmable logic is constructed using the Xilinx Vivado design suite to produce the EN signal that enables the parallel counters. Two clock signal sources with slightly different frequencies were generated by the FPGA PLL to drive the S and R inputs of the SR latch. A clock source with a 100 MHz clock frequency drives the R input while a clock source with a 100.19608 MHz clock frequency drives the S input. These two slipping clock inputs generate a gate signal with different widths. The behavior of the EN signal generation is simulated using Matlab 2020. The simulation result is depicted in Figure 5. In the diagram, Clock 1 in the blue line plot drives the SR latch's S input, while Clock 2 in the red line plot drives the SR latch's R input. The frequencies of Clocks 1 and 2 are slightly different. The output of the SR latch is, therefore, Set, Reset, or Latched based on the phase difference between the two clocks as indicated by the blue line. In this manner, a different width enable signal is generated, as seen in the black line plot. This EN signal input feeds each parallel counter. The counters begin counting at the rising edge of the EN signal and sample their output at the EN signal's falling edge.

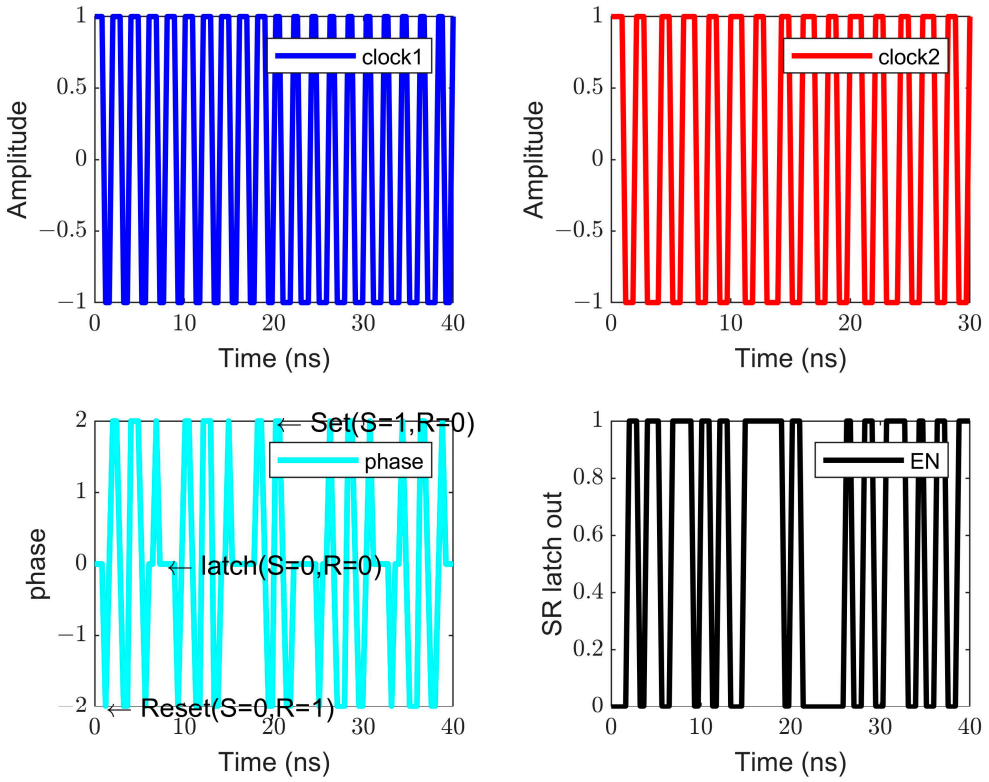


Figure 5. Gate signal generation.

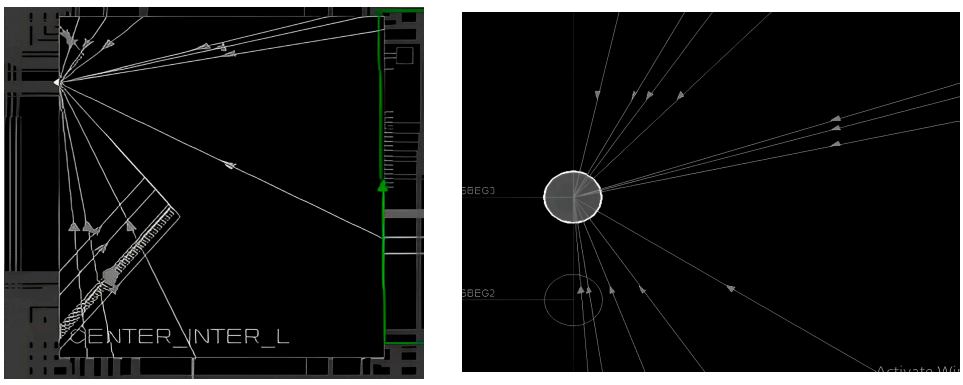
2.3. FPGA's Routing Architecture (ZedBoard)

FPGAs are structured comprising these three blocks:

1. Programmable logic blocks that implement logic functions;
2. Programmable routing architecture that connects these logic blocks;
3. Input/Output blocks connected to the logic blocks through the routing architecture to make off chip connections.

Most of the FPGA area is devoted to its programmable routing resources. FPGAs are made flexible and programmable by this [25,30].

Wires can move from a horizontal routing channel to a vertical routing channel using switch boxes. Figure 6a illustrates the topology of the switch boxes that are available on the ZedBoard, which was extracted from the Xilinx Vivado 2020.1 design suite software. The SB has a variety of PIPs that allow users the selection from various routing wires that are available in the FPGA architecture. This is shown in Figure 6b.



(**a**) (**b**)

Figure 6. (**a**) Switch box in ZedBoard; (**b**) Programmable Interconnect Point available in switchbox.

Routing wires might be short, long or of medium length. The amount of CLBs that the wires traverse determines their length. According to the number of CLBs, they span and the direction in which they route the CLBs, routing wires in the ZedBoard FPGA are given different classifications. The amount of CLBS they cover can be one, two, four, or six. North, Northeast, Northwest, South, Southwest, Southeast, East, or West are route directions for the routing wires.

Only one CLB is covered by short or single-length wires. They are designed to be used for brief connections between CLBs. Medium wires or double-length wires are wires that span two CLBs, and they are intended for moderately long connections with lower routing delays. While long wires that span four and six CLBs are intended for long connections between CLBs, they have lower routing delays and skew compared to the short wires.

3. Experimental Results and Discussion

3.1. Implementation and Simulation Results

An FPGA-based TDC with 512 parallel counters based on delay lines is implemented, and the sorted path delay characteristics with different scenarios are presented in Figure 7. In the figure, the blue line represents the sorted path delay plot for the design without constraining the placement of counters or the net routing order. The red line is the plot of the sorted path delay after routing the EN net first. The yellow line represents the sorted path delay after constraining the placement of the counters in a 4 × 128 rectangular matrix. The purple line is the plot of the sorted path delay after placing the design in the 4 × 128 matrix, routing the EN net first, and constraining the driver location outside the matrix. The plot shows that placing the driver outside the matrix (the purple line) provides a delay line with a dynamic range of 2.5 ns and a resolution of (2.5 ns/512 = 4.9 ps). Although the linearity compared to other scenarios presented in the figure seems to be better in the middle of the delay line, it is not good at the start and end of the plot.

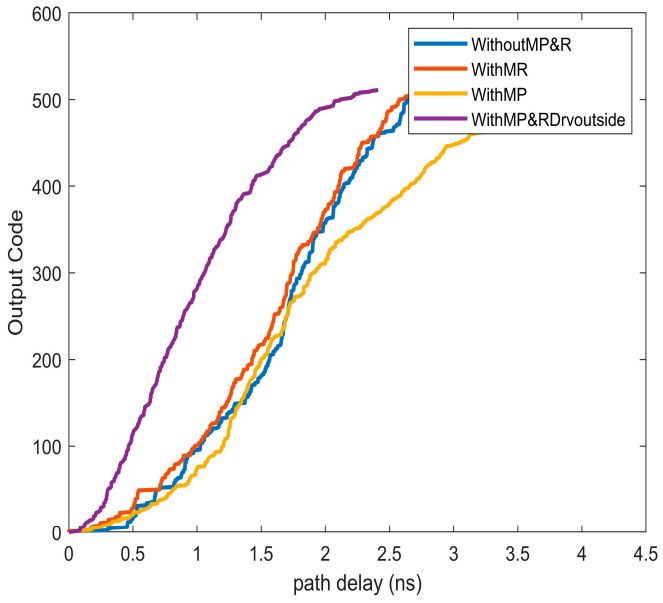


Figure 7. Sorted path delay characteristics for 512 different paths.

As can be seen in Figure 7, even after constraining the placement of the design in a rectangular matrix and placing the driver outside the matrix, obtaining a good linearity delay line is challenging and it involves a lot of trial and error to find the right driver location. After many trials and errors of placing the driver outside the matrix, constraining the router to route the EN net first and constraining the router's minimum path delay time, a good linearity delay line is found. This is presented in Figure 8 below. In the figure, the sorted path delay for distinct locations of the driver is plotted. The yellow line where the

driver is located at X103Y47 and the minimum routing delay is set to 2 ns gives a linear delay line with a resolution of 20 ns/512 = 39 ps. The red line where the driver is located outside the matrix at X113Y47 and the minimum routing delay is set to 4 ns and the blue line with the driver located outside the matrix at X63Y47 and the minimum routing delay set to 4 ns has a better resolution (4 ns/512 = 7.8 ps), but the linearity at the start and end of the path is not as good as those of the yellow line.

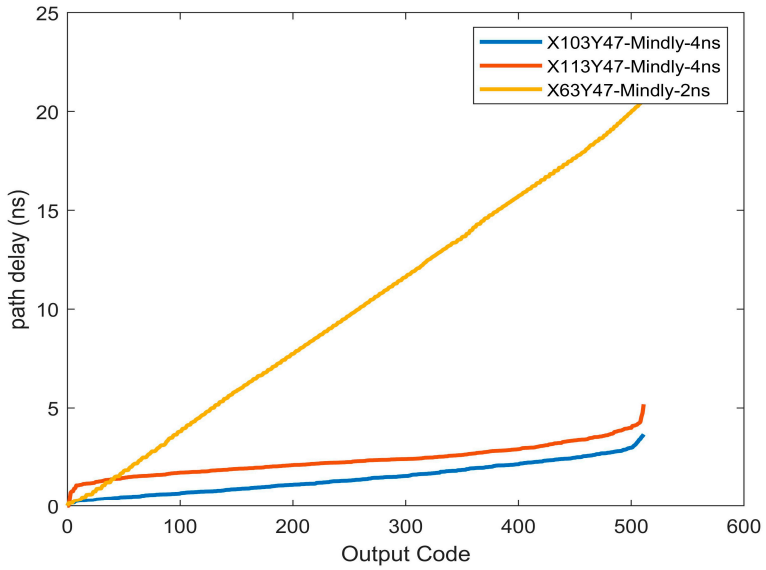


Figure 8. Sorted path delay for 512 paths for different driver locations outside the time-to-digital converter matrix.

Finding the right driver location for linear path delay output is challenging due to the FPGA router's automatic routing and cost-versus-delay tradeoff. Studying the FPGA's routing architecture helps achieve optimal design points with good resolution and linearity.

3.2. Parallel Delay Lines with Fixed Manual Routing of the EN Net

To avoid the trial-and-error approach, manual placement of the counters and fixed routing of the EN net is proposed in this paper. This overcomes the routing unpredictability. The FPGA's CLBs and routing resources are placed in a rectangular fashion in XY plane coordinates. By placing the counters in a rectangular matrix, parallel delay lines with interleaving delay times can be formed. Vertical and horizontal routing lines on the ZedBoard FPGA are available as short wires that only span one CLB, medium wires that span two or four CLBs and long wires that span six or twelve CLBs. These wires have different routing delays associated with them. As reported by Vivado, short wires are 74 ps and long wires are 120 ps. The delay associated with the wires increases when the wires are loaded.

The first scenario of manual fixed routing is performed for 512 counters that are placed in 4 $\times$ 128 rectangular matrixes. The driver is located outside of the matrix at the bottom right. A linear delay line can be achieved by maintaining the uniformity of routing through the matrix. To achieve this, wires are routed via each parallel delay line in the same manner. This is depicted in Figure 9. As can be seen in the figure, four parallel delay lines are formed, each with 128 counters. The counters in a single delay line are routed using vertical short wires that only span a single CLB. The short wires are depicted as the red-directed wires in the figure. The driver is directed to the four parallel delay lines by the green wires. The dynamic range of the TDC is the total sum of the path delays of the short wires in a single delay line. The resolution of the TDC is improved by time interleaving between the parallel delay lines created by the delay difference between the green wires (T1, T2, T3, and T4).

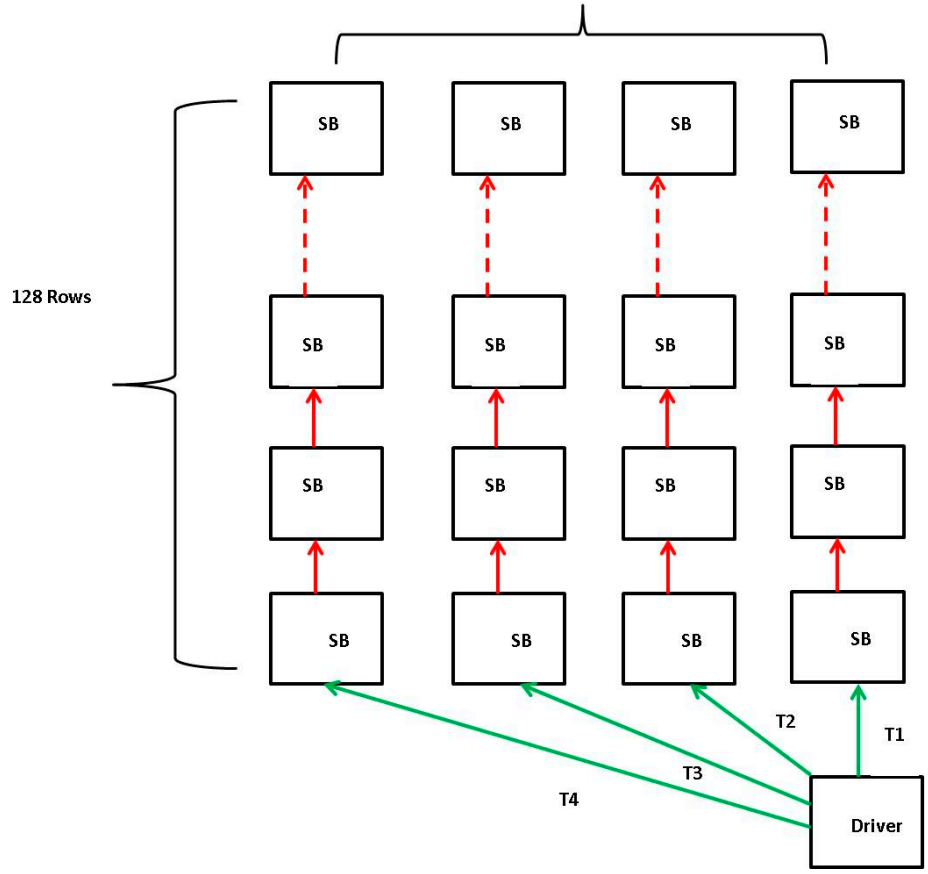


Figure 9. Scenario 1: Manual routing with only short wires.

The simulation result after implementing manual placement and manual routing for Scenario 1 is presented in Figure 10. The result shows that the output is linear; therefore, the positioning of the driver does not require any trial and error. The dynamic range for a 512-unit TDC is 20 ns, and the resolution is 20 ns/512 = 39 ps.

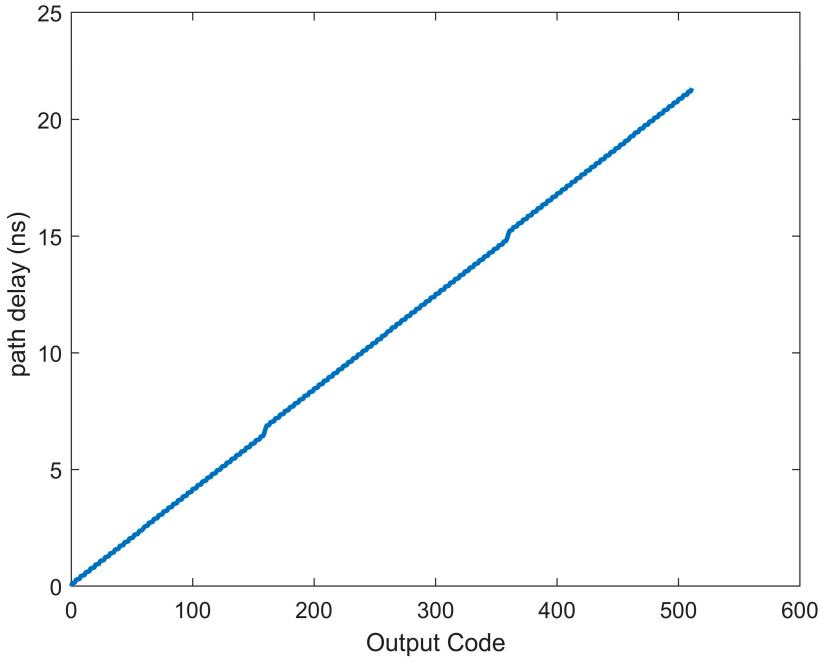


Figure 10. Scenario 1 sorted path delay for 512-unit time-to-digital converter.

The Scenario 2 approach of manual fixed routing is implemented using a combination of long and short wires to route the counter matrixes, as shown in Figure 11. The long wires are the yellow-directed wires that span six CLBs. They are intended for long connections with smaller delays because the switching delays associated with them are small. Short wires (red-directed wires) span only a single CLB. A block contains six counters, each routed with short wires. Blocks are routed to other blocks with long wires with smaller delays compared to the total path delay within a block. This creates more parallel delay lines with fine time interleaving, which in turn leads to better resolution. The time interleaving in Scenario 2 is a two-way interleaving, one caused by the green wires among the four parallel delay lines and the other caused by the yellow wires among the different blocks within a single delay line.

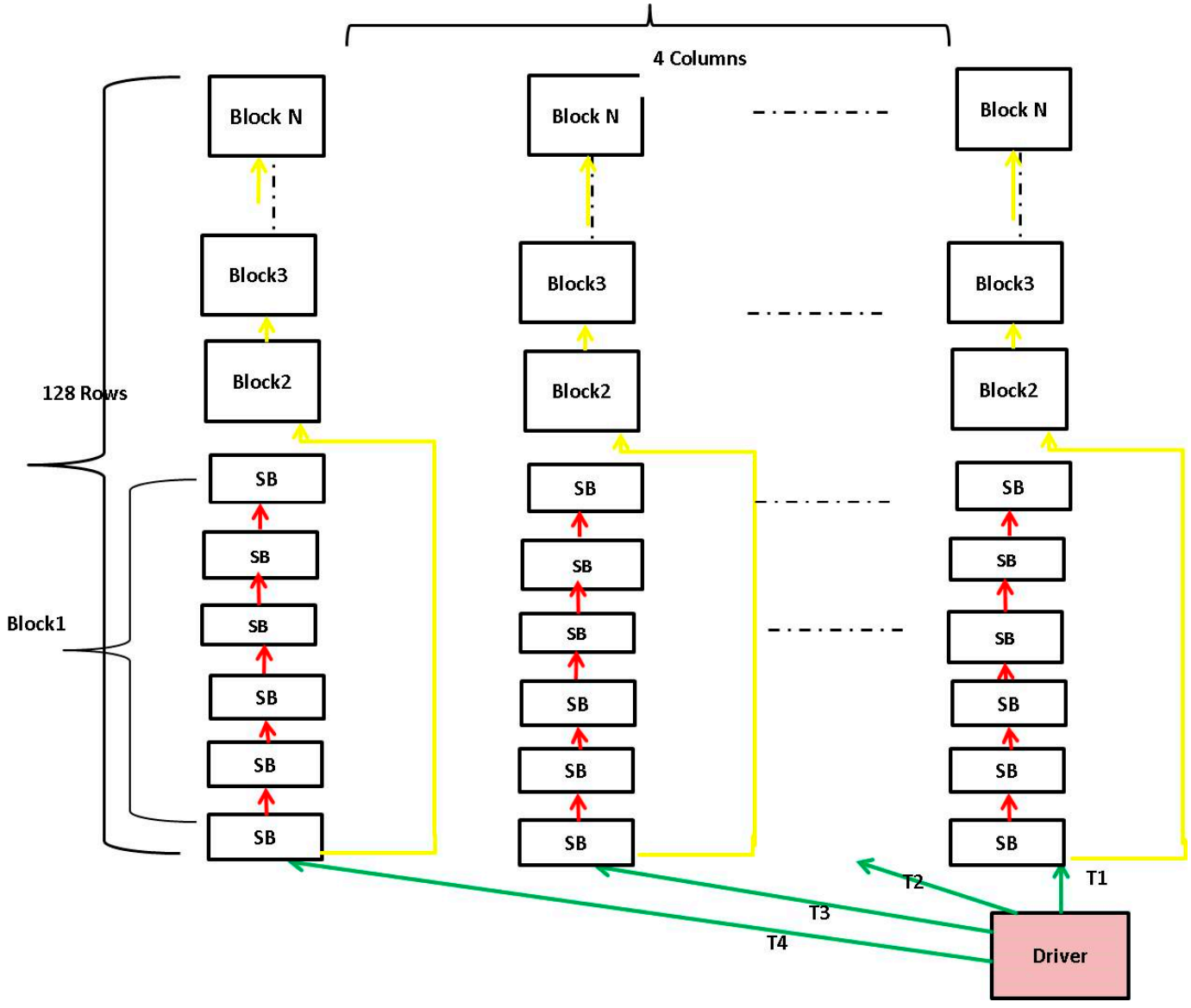


Figure 11. Scenario 2: Manual routing with short wires and long wires.

A comparison of the sorted path delays for 512-unit TDC between Scenario 1 and Scenario 2 implementations is presented in Figure 12. The dynamic range for Scenario 2 is 5 ns, and its resolution is 5 ns/512 = 9.7 ps. Routing the design with a combination of long and short wires increases the number of interleaving blocks and improves the resolution by fourfold when compared to Scenario 1.

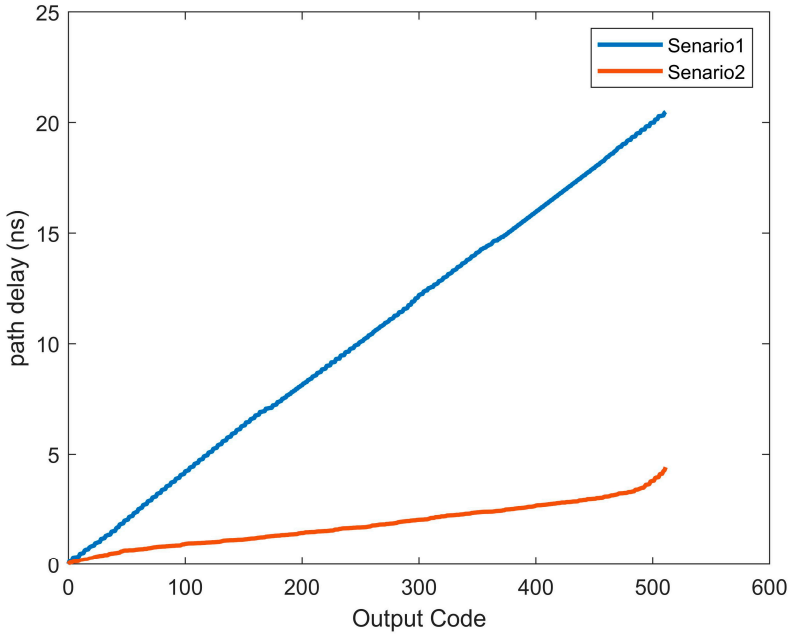


Figure 12. A comparison between manual routing Scenario 1 and Scenario 2 for 512-unit time-to-digital converter.

Increasing the number of counters to 1024 doubles the number of parallel delay lines to eight. A total of 1024 counters are placed in an 8 × 128 matrix, and Scenario 2 fixed manual routing of the EN net is implemented. The sorted path delay for the 1024-unit TDC is plotted in Figure 13. The dynamic range of the path is 5.831 ns, and the resolution is 5.831/1024 = 5.7 ps. Increasing the number of counters enhances the resolution. The reference clock that drives the parallel counters is set to a frequency of 1/5.831 ns = 171 MHz.

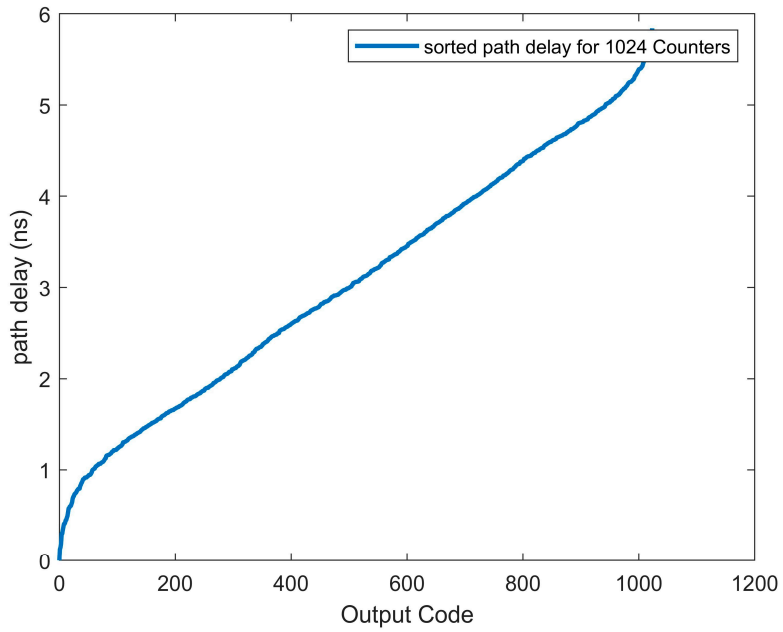


Figure 13. Manual routing Scenario 2: sorted path delay for 1024-unit time-to-digital converter.

3.3. Experiments and Results

For the experimental setup, a Zynq Evaluation and Development (ZED) Board is used. It has a Zynq 7000 ARM processor and an FPGA SOC (System-On-Chip). It is a low-cost

development board used for prototyping [29]. The Zed board used for the experiment is presented in Figure 14.

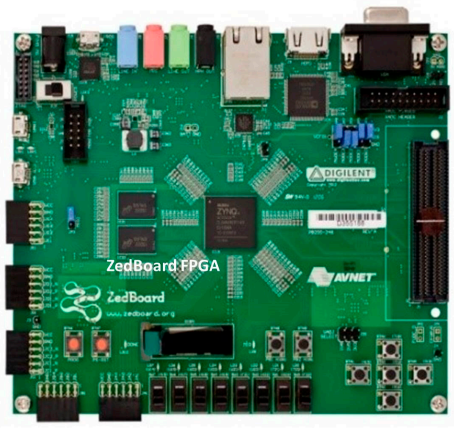


Figure 14. Zynq Evaluation and Development board used for evaluating TDC.

The steps followed for the experimental setup are presented in Figure 15. An IP integrator is invoked to create a block diagram for the implemented design. A top-level hardware description language (HDL) wrapper is generated. After successfully running constrained synthesis and implementation of the design, a bit stream is generated, and the hardware is exported to a PC. A software platform and application projects are created on Xilinx Vitis to test the hardware. The output of the counters is sampled at the occurrence of the STOP event and stored in memory. The processing of the data is performed on the Zynq processing system of the evaluation board. Vitis has a built-in serial terminal program that makes it possible to transfer the processed data through a universal asynchronous receiver and transmitter (UART) to a PC. The graphical plots are generated in Matlab 2020 [31].

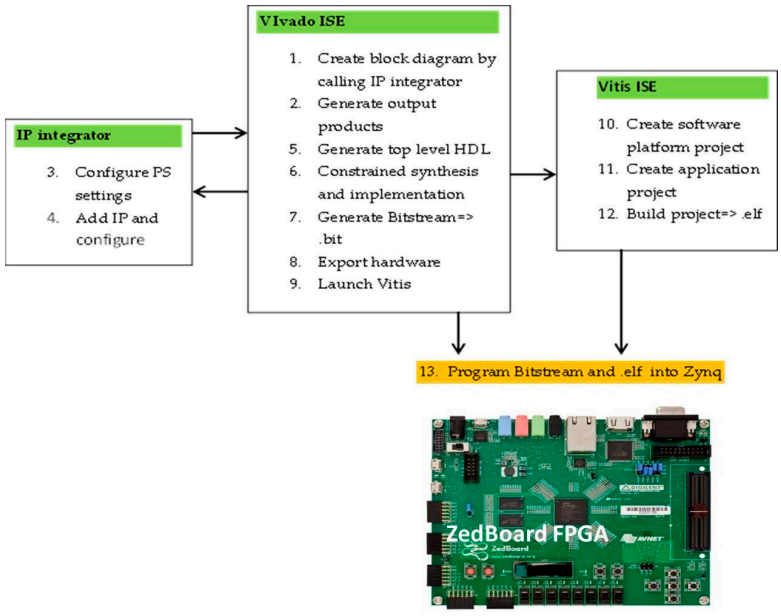


Figure 15. Embedded System Design using Vivado + Vitis.

The test setup for the design is depicted in Figure 16. It includes the Zynq processing system (PS), the TDC tester module, an SR latch that generates an Enable signal for the TDC module, the AXI interfaces and block RAMs. All the components are configured according to the design. The board is powered by a 12 V power supply. The data evaluation and calculation are performed on a personal computer.

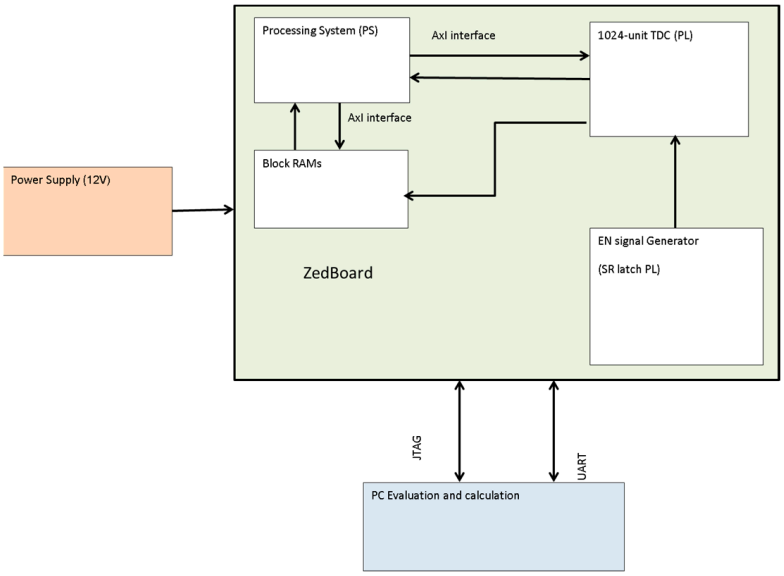
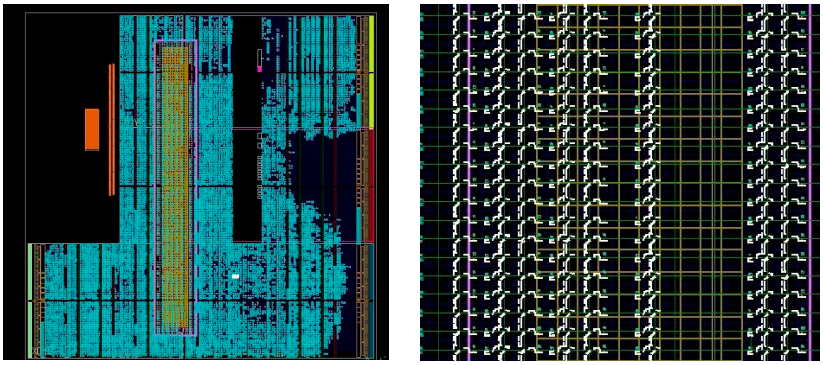


Figure 16. Experimental test system.

The design after placement and routing on the FPGA chip is presented in Figure 17. The TDC is placed in 8 × 128 matrixes. This is shown by the purple rectangular line in Figure 17a. The driver is placed outside the matrix on the bottom right. The routing of the design is also presented in white lines in Figure 17b. The routing is manually fixed so that it is uniform.



(**a**) (**b**)

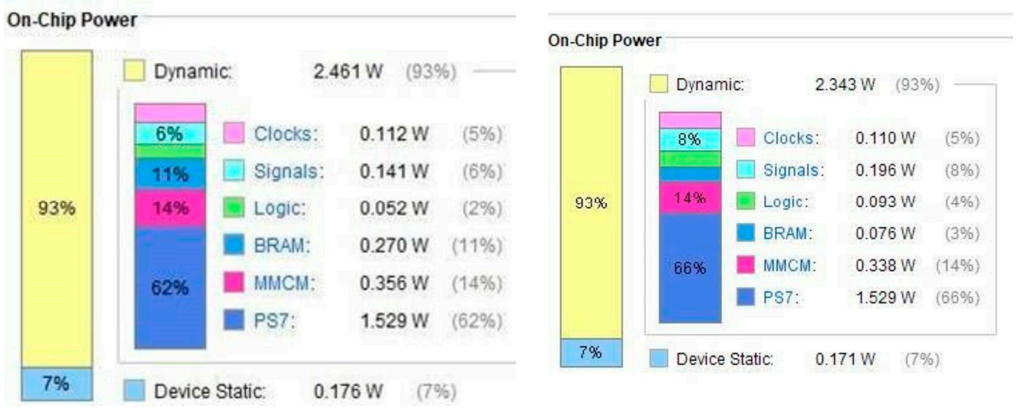
Figure 17. (**a**) The proposed time-to-digital converter after place and route; (**b**) Manual fixed routing.

A resource utilization summary for 512- and 1024-unit TDC implementations as reported by Viviado 2020.1 is presented in Table 1.

Table 1. Resource utilization report.

	512-Unit TDC			**1024-Unit TDC**	
	Available	**Used**	**Utilization (%)**	**Used**	**Utilization (%)**
Slice registers	106,400	25,290	23.77	41,175	38.70
Slice LUTs	53,200	16,864	31.70	25,819	48.53
LUTs as logic	53,200	11,288	21.22	16,875	31.72
LUTs as memory	17,400	5576	32.05	8944	51.40
Slices	13,300	6947	52.23	11,041	83.02
Block RAM	140	32.5	23.21	32.5	23.21
BUFGCTRL	32	7	21.88	8	25.00
MMCME2-ADV	4	3	75	3	75

The dynamic and static power consumptions for 512- and 1024-unit TDC implementations are presented in Figure 18, as reported by Vivado. Overall, 93 percent of the total on-chip power is dynamic power, and of this, more than 60 percent is consumed by the processing system (PS). For a 1024-unit TDC, the total chip power consumption is 2.514 watts.



(**a**) (**b**)

Figure 18. Dynamic and static power consumption: (**a**) 512-unit time-to-digital converter, (**b**) 1024-unit time-to-digital converter.

3.3.1. Histogram of the TDC

To build the histogram for the proposed TDC, the statistical code density test method is used. For a 4-bit gray code coarse counter and 1024 parallel counters, there are 16,384 bins. Many measurements were taken for randomly generated time interval inputs to the TDC. Over 6,000,000 samples were taken, and the number of hits in each bin was counted. The histogram is plotted in Figure 19. The parallel counters that make up the TDC's delay line for precise time measurement reveal the repeated pattern over 1024 bins in the histogram.

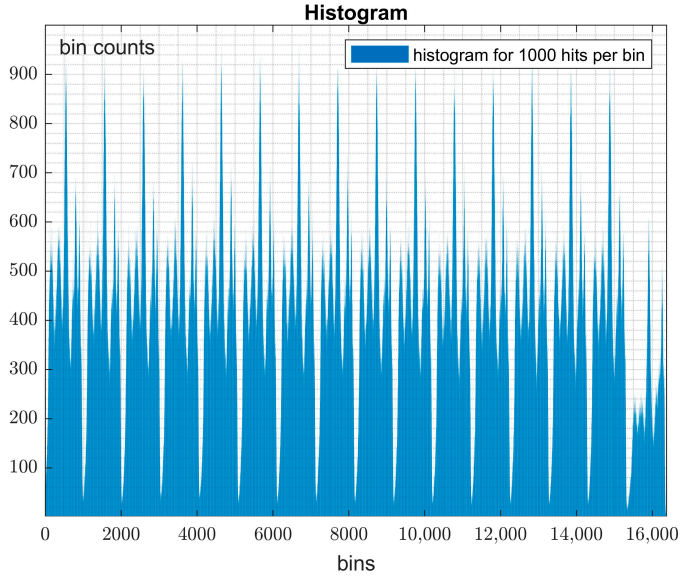


Figure 19. Histogram for 1024-unit time-to-digital converter.

3.3.2. Transfer Function of the TDC

The transfer function is derived from the histogram of the TDC. The normalized time interval is plotted for 16,384 output codes. The cumulative sum of the number of counts over the input time range divided by the total number of samples is the normalized input time range. The transfer function for the implemented TDC is presented in Figure 20.

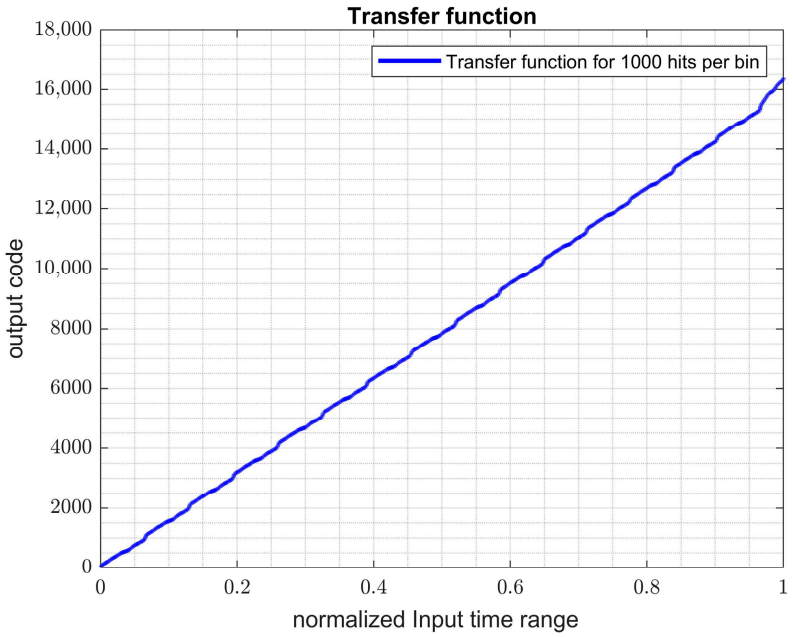


Figure 20. Transfer Function for 1024-unit time-to-digital converter.

3.3.3. Non-Linearity of the TDC

Using the statistical code density test method, the differential nonlinearity (*DNL*) is extracted from the histogram. The *DNL* is the deviation from the ideal bin width for a given input time range. Mathematically, the *DNL* for output code n can be expressed as in Equation (6),

$$DNL(n) = \frac{ActualHits}{TotalHits * p(i)} - 1 \quad (6)$$

where $p(n)$ is the probability of code n.

The *DNL* plot for 1024-unit TDC is plotted in Figure 21. The *DNL* is in the range of −0.9953 Least Significant Bit (LSB) (−5.67 ps) to +1.3651 LSB (7.75 ps). The Root Mean Square (RMS) *DNL* is 0.4312 LSB (2.4577 ps).

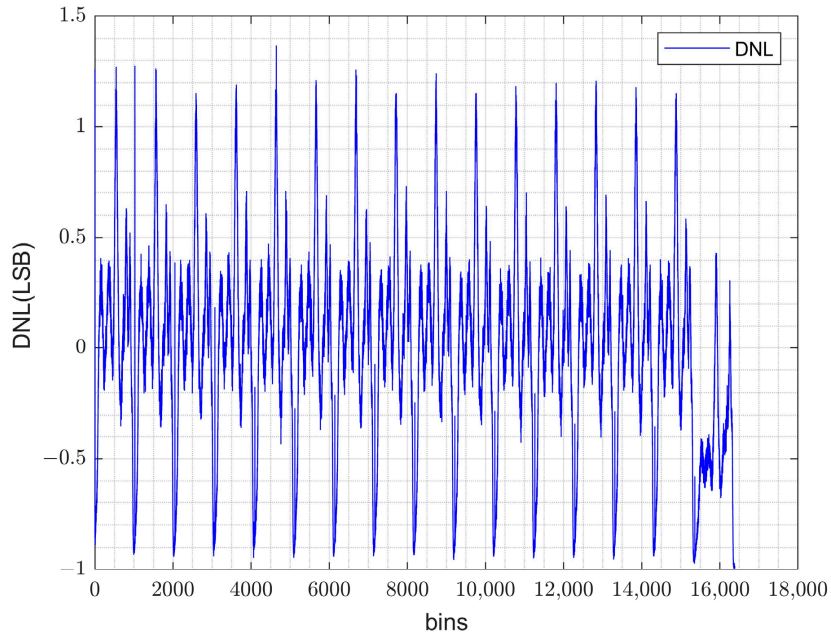


Figure 21. Differential nonlinearity for 16,384 bins.

Another non-linearity performance metric is the TDC's integral non-linearity (INL). It is the deviation from the ideal linear slope of the TDC. We use the best straight-line linear fitting technique, as plotted in Figure 22, to calculate the INL. The deviation of the slope of the actual transfer function from the slope of the ideal transfer function is calculated for each output bin. The INL plot for 16,384 bins is depicted in Figure 23. The INL is high at the start and end of the delay line. The INL is in the range of +21.8123 LSB (124.33 ps) at the end of the plot and −0.5666 LSB (−3.2294 ps). The RMS INL is 2.1545 LSB (12.2807 ps).

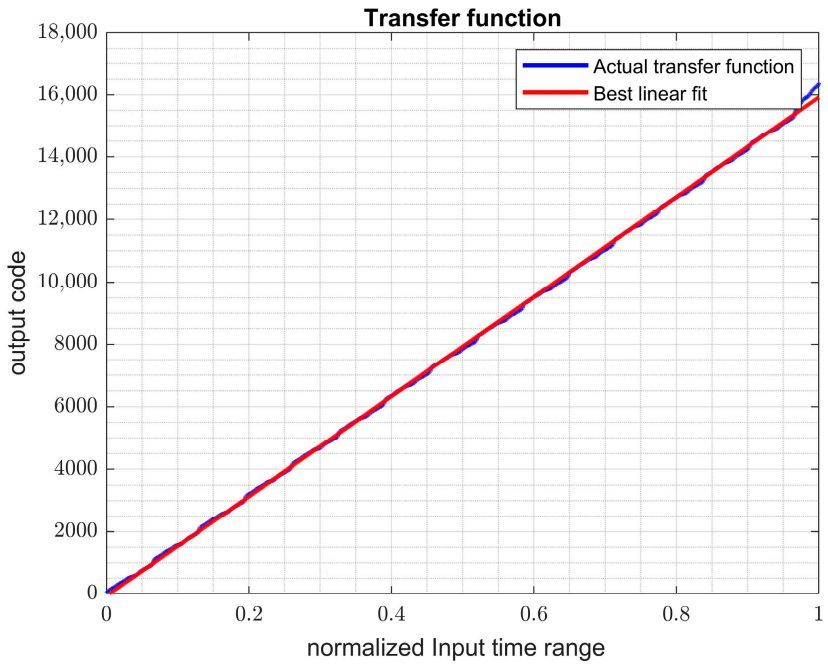


Figure 22. Best linear fit Transfer function.

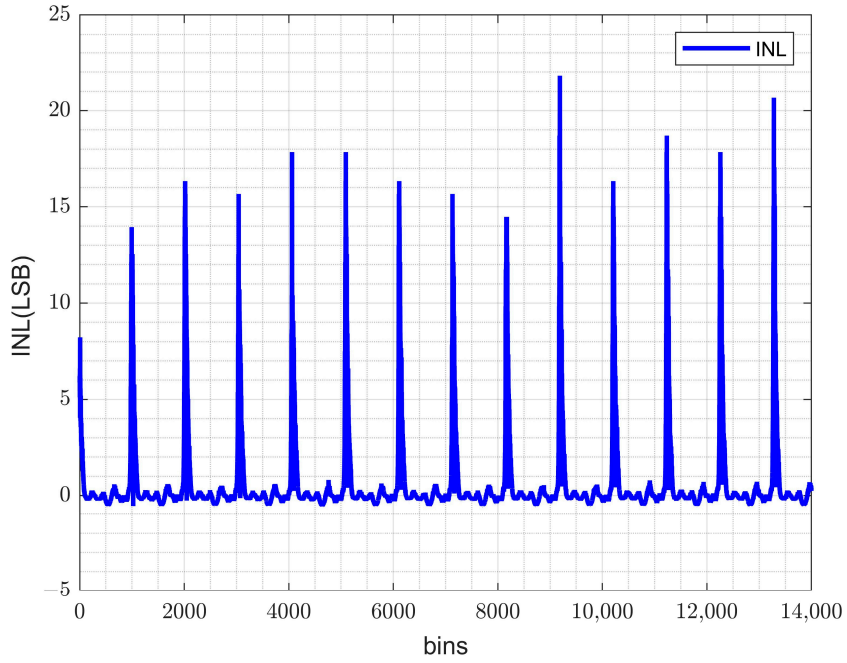


Figure 23. Integral non-linearity for 16,384 bins.

The repetitive pattern of the 1024 bins over 16,384 bins can also be seen in the INL plot. As can be seen from the figure, the INL is higher for some bins in the range of 1024 bins.

Rearranging the number of bins to 1024, the INL plot is depicted in Figure 24. The results show that the INL is not good at the start and end of the plot, but it is good in the middle of the delay line. The INL is in the range of 2.1514 LSB (12.263 ps) and −0.4813 LSB (2.7436 ps).

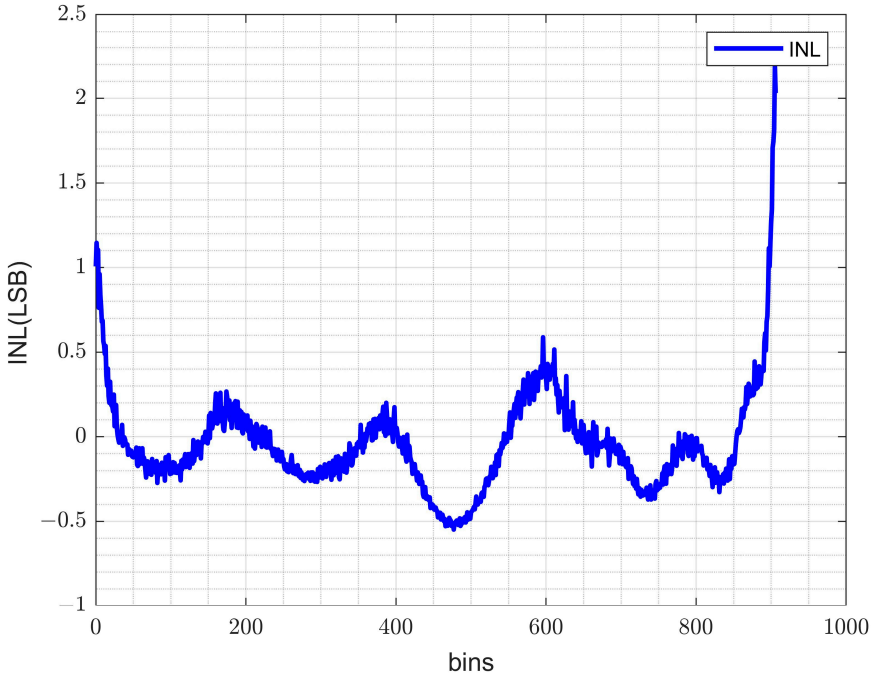


Figure 24. Integral non-linearity for 1024 bins.

3.3.4. Single-Shot Precision of the TDC

To evaluate the time-capturing capacity of the implemented TDC, a constant time interval of 1.25 ns and 2.5 ns is measured. Repetitive measurements for each constant time input are taken, and the distributions of the measurement results are obtained. The mean value (μ) is calculated from the distribution function as in Equation (7). The standard deviation (σ) is calculated using Equation (8), where n_i is the frequency of the ith bin, m_i is the ith bin and N is the total sample size.

$$\mu = \sum_{i=1}^{N} \frac{n_i m_i}{N} \tag{7}$$

$$\sigma = \sqrt{\sum_{i=1}^{N} \frac{n_i (m_i - \mu)^2}{N - 1}} \tag{8}$$

The two clock inputs for the SR latch are kept at a clock frequency of 100 MHz and the phase of the clock input to the R input of the SR latch is shifted by 45 degrees and 90 degrees to produce a constant time input of 1.25 ns and 2.5 ns, respectively. Sufficiently large numbers of repetitive measurements for each constant time input are taken, and the distributions of the measurement results are depicted in Figures 25 and 26, respectively. For a 1.25 ns constant time input, the calculated mean is 244.8 LSB = 1.39 ns, and the calculated standard deviation is 14.23 LSB (81.1 ps). For the 2.5 ns constant time measurement, the calculated mean value is 443.6 LSB (2.53 ns) and the standard deviation is 25.01 LSB (142.6 ps). Results show that the standard deviation increases for longer time inputs. This is due to the clock skew associated with the reference clock input. As the delay path increases, the effect of clock skew on the stability of measurement also increases.

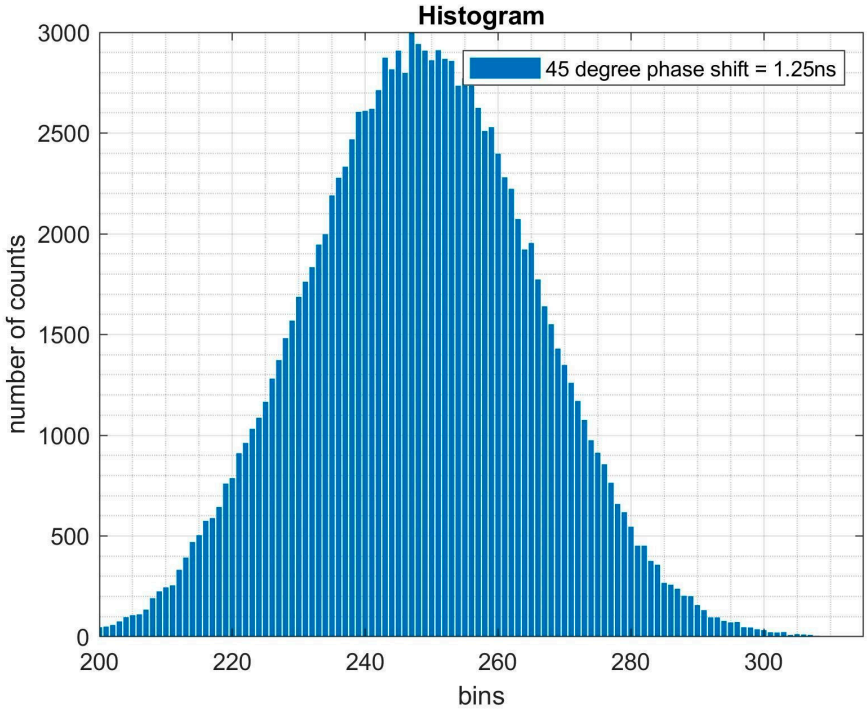


Figure 25. Distribution for 1.25 ns constant time interval input.

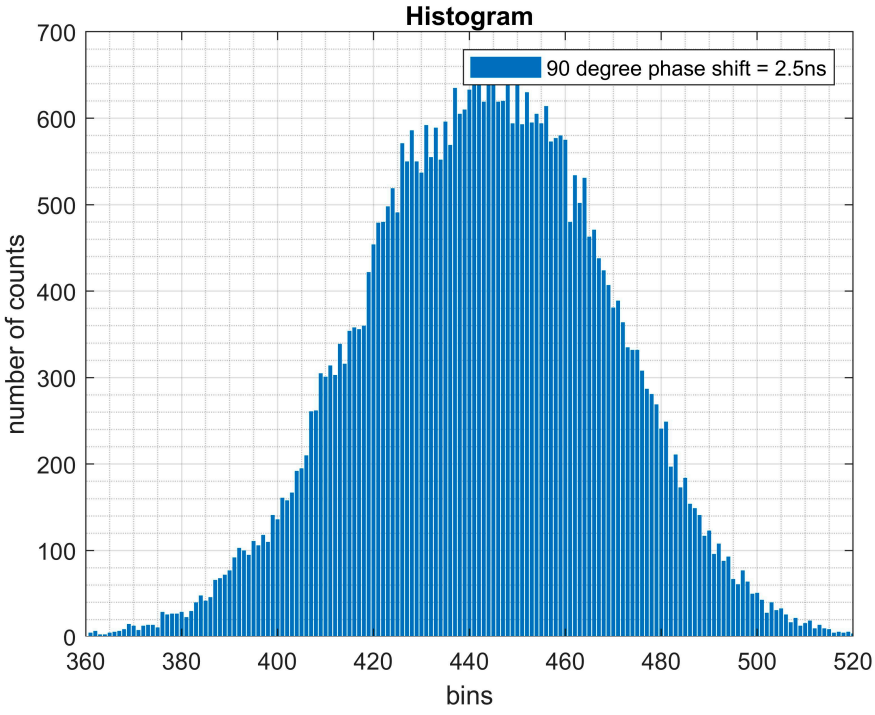


Figure 26. Distribution for 2.5 ns constant time interval input.

To reduce the effect of the clock skew, a dedicated global clock distribution network is used to feed the reference clock to the parallel counters. Using the global clock distribution network instead of the regional clocks reduces clock skew and increases measurement stability. This effect is tested in a 2.5 ns constant time measurement, and the improvement in the standard deviation is shown in Figure 27. The standard deviation is reduced to 7.42 LSB (42.3 ps), which is a 70% improvement in measurement stability when compared to the 2.5 ns constant time measurement using a regional clocking of the reference clock.

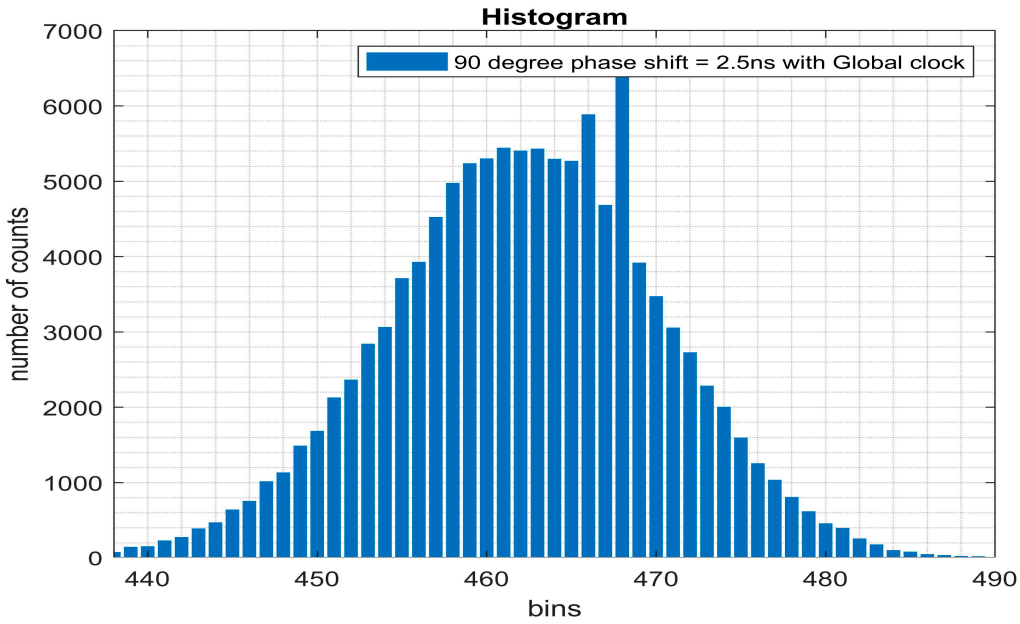


Figure 27. Distribution for 2.5 ns constant time interval input after global clock distribution network implementation.

3.4. Comparison with Other TDCs

Table 2 compares the performance of our TDC with other TDCs implemented on FPGAs. Most of the previous works presented in Table 2 use the carry chain delay line approach. Our work and the work in [2] use parallel counters based on delay lines to realize TDCs. Compared to the work in [2], our TDC has a higher resolution, which is 5.7 ps, and a DNL of 0.43 LSB (rms). The work in [9] uses Digital Signal Processing (DSP) units found in FPGAs as delay elements. Although it offers a higher performance in resolution (4.2 ps) compared to our work, the DNL of the implementation is 20 LSB which is not as good as our TDC. Work [19] describes better resolution (0.46 ps), but it uses a combination of Wave Union Launcher (WU) and dual sampling techniques to improve performance.

Table 2. Comparison with prior works.

Work	**This Work**	**[2] (2017)**	**[4] (2013)**	**[8] (2013)**	**[9] (2019)**	**[12] (2022)**	**[19] (2023)**	**[32] (2015)**
Method	Counters based on delay lines	Counters based on delay lines	Carry4 delay line	Carry4 delay line	DSP delay lines	Carry4 delay line	Carry4 delay line with WU +dual sampling	Carry4 delay line with calibration
FPGA process technology	28 nm	65 nm	90 nm	40 nm	28 nm	40 nm	16 nm	28 nm
Number of bins	16,384	1024	256	-	480	300	6000	80
Reference frequency	171 MHz	133.3 MHz	115 MHz	-	700 MHz	-	450 MHz	710 MHz
Resolution	5.7 ps	7.4 ps	48 ps	19. 6 ps	4.2 ps	20 ps	0.46 ps	15 ps (rms)
DNL	0.4312 LSB (rms)	0.74 LSB	-	1.5 LSB	20 LSB	0.3 LSB (rms)	6.84 LSB	1 LSB
INL	2.1545 LSB (rms)	1.57 LSB	-	1. 61 LSB	31.54 LSB	0.45 LSB (rms)	72. 55 LSB	0.8 LSB
Dynamic range	93.6 ns	-	6.150 ns	-	4 ns	3.2 ns	-	-

4. Conclusions

A TDC based on a matrix of 1024 parallel 4-bit counters is implemented on a 28 nm CMOS FPGA on a ZedBoard. The delay line is realized using the routing resources of the

FPGA. By manually fixing the routing of the delay line and applying user-constrained placement of the counters and the driver, the unpredictability of the routing is overcome. Finding a perfect design point with high-resolution TDC was the study's aim, and it was accomplished. In comparison to earlier studies, our method provides a greater resolution of 5.7 ps. Moreover, the tedious trial-and-error method used by earlier studies to discover the appropriate driver placement for good linearity and resolution is overcome. The implemented TDC has a DNL of 0.4312 LSB (rms) or 2.4577 ps, and an INL of 2.1545 LSB (rms) or 12.2807 ps.

The experimental results show that there is a tradeoff between the resolution, linearity, resource utilization, and power consumption of the design. Increasing the number of parallel delay lines increases the resolution, but with an overhead in resource utilization and power consumption. The measurement range can be increased by increasing the number of bits of the coarse counter.

Combining nonlinearity calibration techniques such as bin-by-bin calibration or average-bin-width calibration techniques or using the Wave union launcher with this implementation may lead to improved performances in the future. High-performance ASIC-based TDCs may also result from the realization of TDCs with internal routing resources acting as delay components.

Author Contributions: Conceptualization R.T.S. and J.P.; methodology, R.T.S.; software, R.T.S.; validation, R.T.S., J.P. and P.L.; formal analysis, R.T.S.; investigation, R.T.S.; resources, R.T.S., J.P. and P.L.; writing—original draft preparation, R.T.S.; writing—review and editing, R.T.S., G.A. and P.L.; supervision, G.A., J.P. and P.L.; project administration, G.A., J.P. and P.L. All authors have read and agreed to the published version of the manuscript.

Funding: This research was supported by the Home-Grown Ph.D. Program (HGPP) funded by the Ethiopian Ministry of Education.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Roberts, G.W.; Ali-Bakhshian, M. A brief introduction to time-to-digital and digital-to-time converters. *IEEE Trans. Circuits Syst. II Express Briefs* **2010**, *57*, 153–157. [CrossRef]
2. Zhang, M.; Wang, H.; Liu, Y. A 7.4 ps FPGA-based TDC with a 1024-unit measurement matrix. *Sensors* **2017**, *17*, 865. [CrossRef]
3. Van Bockel, B.; Leroux, P.; Prinzie, J. Tradeoffs in time-to-digital converter architectures for harsh radiation environments. *IEEE Trans. Instrum. Meas.* **2021**, *70*, 2005710. [CrossRef]
4. Lai, J.; Luo, Y.; Shao, Q.; Bao, L.; Liu, X. A high-resolution TDC implemented in a 90 nm process FPGA. In Proceedings of the 2013 IEEE 10th International Conference on ASIC, Shenzhen, China, 28–31 October 2013; pp. 1–3.
5. Tsai, T.-H.; Yuan, M.-S.; Chang, C.-H.; Liao, C.-C.; Li, C.-C.; Staszewski, R.B. 14.5 A 1.22 ps integrated-jitter 0.25-to-4 GHz fractional-N ADPLL in 16 nm FinFET CM0S. In Proceedings of the 2015 IEEE International Solid-State Circuits Conference-(ISSCC) Digest of Technical Papers, San Francisco, CA, USA, 22–26 February 2015; pp. 1–3.
6. Wu, W.; Staszewski, R.B.; Long, J.R. A 56.4-to-63.4 GHz multi-rate all-digital fractional-N PLL for FMCW radar applications in 65 nm CMOS. *IEEE J. Solid-State Circ.* **2014**, *49*, 1081–1096. [CrossRef]
7. Prinzie, J.; Steyaert, M.; Leroux, P. A single shot TDC with 4.8 ps resolution in 40 nm CMOS for high energy physics applications. *J. Instrum.* **2015**, *10*, C01031. [CrossRef]
8. Fishburn, M.; Menninga, L.H.; Favi, C.; Charbon, E. A 19.6 ps, FPGA-based TDC with multiple channels for open source applications. *IEEE Trans. Nucl. Sci.* **2013**, *60*, 2203–2208. [CrossRef]
9. Kwiatkowski, P.K. Employing FPGA DSP blocks for time-to-digital conversion. *Metrol. Meas. Syst.* **2019**, *26*, 631–643.
10. Cui, K.; Ren, Z.; Li, X.; Liu, Z.; Zhu, R. A high-linearity, ring-oscillator-based, Vernier time-to-digital converter utilizing carry chains in FPGAs. *IEEE Trans. Nucl. Sci.* **2016**, *64*, 697–704. [CrossRef]
11. Mattada, M.P.; Guhilot, H. Time-to-digital converters—A comprehensive review. *Int. J. Circuit Theory Appl.* **2021**, *49*, 778–800. [CrossRef]
12. Dikopoulos, E.; Birbas, M.; Birbas, A. An Adaptive Downsampling FPGA-Based TDC Implementation for Time Measurement Improvement. *Chips* **2022**, *1*, 175–190. [CrossRef]
13. Fan, H.-H.; Cao, P.; Liu, S.-B.; An, Q. TOT measurement implemented in FPGA TDC. *Chin. Phys. C* **2015**, *39*, 116101. [CrossRef]
14. Tontini, A.; Gasparini, L.; Pancheri, L.; Passerone, R. Design and characterization of a low-cost FPGA-based TDC. *IEEE Trans. Nucl. Sci.* **2018**, *65*, 680–690. [CrossRef]

15. Dudek, P.; Szczepanski, S.; Hatfield, J.V. A high-resolution CMOS time-to-digital converter utilizing a Vernier delay line. *IEEE J. Solid-State Circ.* **2000**, *35*, 240–247. [CrossRef]
16. Cui, K.; Li, X.; Liu, Z.; Zhu, R. Toward implementing multichannels, ring-oscillator-based, Vernier time-to-digital converter in FPGAs: Key design points and construction method. *IEEE Trans. Radiat. Plasma Med. Sci.* **2017**, *1*, 391–399. [CrossRef]
17. Zhang, J.; Zhou, D. An 8.5-ps two-stage Vernier delay-line loop shrinking time-to-digital converter in 130-nm flash FPGA. *IEEE Trans. Instrum. Meas.* **2017**, *67*, 406–414. [CrossRef]
18. Van Bockel, B.; Prinzie, J.; Cao, Y.; Leroux, P. A delay locked loop for time-to-digital converterswith quick recovery and low hysteresis. In Proceedings of the TWEPP 2018 Topical Workshop on Electronics for Particle Physics, Antwerpen, Belgium, 17–21 September 2019.
19. Wang, Y.; Xie, W.; Chen, H.; Li, D.D.-U. High-resolution time-to-digital converters (TDCs) with a bidirectional encoder. *Measurement* **2023**, *206*, 112258. [CrossRef]
20. Liu, C.; Wang, Y. A 128-channel, 710 M samples/second, and less than 10 ps RMS resolution time-to-digital converter implemented in a Kintex-7 FPGA. *IEEE Trans. Nucl. Sci.* **2015**, *62*, 773–783. [CrossRef]
21. Wang, Y.; Liu, C.; Cheng, X.; Li, D. Spartan-6 FPGA based 8-channel time-to-digital converters for TOF-PET systems. In Proceedings of the 2015 IEEE Nuclear Science Symposium and Medical Imaging Conference (NSS/MIC), San Diego, CA, USA, 3–5 November 2015; pp. 1–3.
22. Amiri, A.M.; Boukadoum, M.; Khouas, A. A multihit time-to-digital converter architecture on FPGA. *IEEE Trans. Instrum. Meas.* **2008**, *58*, 530–540. [CrossRef]
23. Daigneault, M.-A.; David, J.P. A high-resolution time-to-digital converter on FPGA using dynamic reconfiguration. *IEEE Trans. Instrum. Meas.* **2011**, *60*, 2070–2079. [CrossRef]
24. Wang, H.; Zhang, M.; Yao, Q. A new realization of time-to-digital converters based on FPGA internal routing resources. *IEEE Trans. Ultrason. Ferroelectr. Freq. Control* **2013**, *60*, 1787–1795. [CrossRef]
25. Prado, D.F.G. Tutorial on FPGA routing. *Electrón.-UNMSM* **2006**, *17*, 23–33.
26. Huang, H. A 0.1 ps Resolution Coarse-Fine Time-to-Digital Converter with 2.21 ps Single-Shot Precision. Ph.D. Thesis, The University of Texas at Dallas, Richardson, TX, USA, 2018.
27. Brown, S.; Rose, J. Architecture of FPGAs and CPLDs: A tutorial. *IEEE Des. Test Comput.* **1996**, *13*, 42–57. [CrossRef]
28. Farooq, U.; Marrakchi, Z.; Mehrez, H.; Farooq, U.; Marrakchi, Z.; Mehrez, H. FPGA architectures: An overview. In *Tree-Based Heterogeneous FPGA Architectures: Application Specific Exploration and Optimization*; Springer: Berlin/Heidelberg, Germany, 2012; pp. 7–48.
29. Avnet. *ZedBoard (Zynq Evaluation and Development) Hardware User's Guide, Version 2.2*; Avnet: Phoenix, AZ, USA, 2014.
30. Farooq, U.; Marrakchi, Z.; Mehrez, H. *Tree-Based Heterogeneous FPGA Architectures: Application Specific Exploration and Optimization*; Springer Science & Business Media: Berlin/Heidelberg, Germany, 2012.
31. Hanselman, D.C.; Littlefield, B.L. *Mastering Matlab*; Prentice Hall Press: Hoboken, NJ, USA, 2011.
32. Wang, Y.; Liu, C. A nonlinearity minimization-oriented resource-saving time-to-digital converter implemented in a 28 nm Xilinx FPGA. *IEEE Trans. Nucl. Sci.* **2015**, *62*, 2003–2009. [CrossRef]