*electronics*

*Article*

# Optimal Implementation of Tapped Delay Line Time-to-Digital Converters in 20 nm Xilinx UltraScale FPGAs

**Mattia Morabito, Nicola Lusardi *,†, Fabio Garzetti †, Gabriele Fiumicelli, Gabriele Bonanno, Enrico Ronconi, Andrea Costa and Angelo Geraci**

DEIB (Dipartimento di Elettronica, Informazione e Bioingegneria), Politecnico di Milano, Via Golgi 40, 20133 Milano, Italy; mattia.morabito@mail.polimi.it (M.M.); fabio.garzetti@polimi.it (F.G.); gabriele.fiumicelli@mail.polimi.it (G.F.); gabriele.bonanno@polimi.it (G.B.); enrico.ronconi@polimi.it (E.R.); andrea1.costa@polimi.it (A.C.); angelo.geraci@polimi.it (A.G.)

* Correspondence: nicola.lusardi@polimi.it

† These authors contributed equally to this work.

**Abstract:** This study investigated implementation strategies to optimize the precision of Tapped Delay Line (TDL) Time-to-Digital Converters (TDCs) designed for Xilinx 20 nm UltraScale Field-Programmable Gate Arrays (FPGAs). This optimization process aims to bridge the performance gap between FPGA-based TDCs, which are more flexible and suitable for fast prototyping, and the better-performing Application-Specific Integrated Circuit (ASIC) solutions, making FPGA-based TDCs viable for cutting-edge applications. Our key areas of focus included the optimal design of the decoder, the degree of sub-interpolation, and the placement of TDLs, with particular emphasis on the clocking distribution scheme within the Configurable Logic Block (CLB) to minimize the effects of Bubble Errors (BEs) and quantization error. The research led to the development and comparison of multiple TDL TDC solutions implemented on a Kintex UltraScale device (i.e., XCKU040-2FFVA1156E) housed on a KCU105 general-purpose Evaluation Board (EVB). From these, two main solutions emerged: one with high precision and one with low area. The first one was characterized by a Single-Shot Precision (SSP) of 2.64 ps r.m.s., and by Differential and Integral Non-Linearity (DNL/INL) Errors of 0.523 ps and 16.939 ps, respectively, occupying 883 CLBs and 126 kb of Block RAM (BRAM). The second one had an SSP of 3.75 ps r.m.s., a DNL of 0.599 ps, and an INL of 7.151 ps, and it occupies only 259 CLBs and 72 kb of BRAM.

**Keywords:** bubble errors; decoding; Tapped Delay Line (TDL); Time-to-Digital Converter (TDC); Field-Programmable Gate Array (FPGA)



**Citation:** Morabito, M.; Lusardi, N.; Garzetti, F.; Fiumicelli, G.; Bonanno, G.; Ronconi, E.; Costa, A.; Geraci, A. Optimal Implementation of Tapped Delay Line Time-to-Digital Converters in 20 nm Xilinx UltraScale FPGAs. *Electronics* **2024**, *13*, 4888. https://doi.org/10.3390/electronics13244888

Academic Editors: Stefano Ricci and Alexander Barkalov

Received: 11 September 2024
Revised: 8 November 2024
Accepted: 7 December 2024
Published: 11 December 2024



## 1. Introduction

Within the field of electronics engineering, Time-to-Digital Converters (TDCs) are integral parts of time-resolved systems and hold a prominent place [1]. Their functions encompass multiple fields, including quantum technologies [2,3], life sciences [4,5], and nuclear physics [6,7].

In digital electronics, including TDC circuits, Field-Programmable Gate Arrays (FPGAs) provide unmatched customization, adaptability, and off-the-shelf fast prototyping solutions compared to Application-Specific Integrated Circuit (ASIC) solutions, thanks to their rapid prototyping capabilities, reconfigurability, shorter time-to-market, and lower Non-Recurring Engineering (NRE) costs [8]. In this way, especially in R&D and academia, the use of FPGA-based TDCs has had a profound impact on the development of multiple novel TDC architectures [9,10]. The difficulty of developing competitive FPGA-based TDC implementations that excel in both performance (i.e., number of channels in a single device, resolution (i.e., LSB), Single-Shot Precision (SSP), linearity, dead time, and acquisition rate [5,11–16]) and efficiency (i.e., area occupied and power consumed by a single channel [5]) is addressed in the literature.

### *1.1. FPGA-Based Tapped Delay Line TDC*

The FPGA-based TDC architecture that best balances the performance and efficiency trade-off is the Tapped Delay Line (TDL) [17], also known as TDL TDCs [9,10]. In FPGAs, which are subdivided into predefined Configurable Logic Blocks (CLBs), each containing a limited set of resources, TDLs are usually implemented by using carry chains available either in Digital Signal Processing (DSP) blocks [18] or in the FPGA fabric as carry logic primitives (i.e., CARRY4 for Xilinx 28 nm 7-Series and CARRY8 for Xilinx 16/20 nm UltraScale FPGAs) [12,13]. According to the scientific literature, at the same technological node, DSP-based TDLs are less linear compared to carry-based TDLs, due to the presence of ultra-bins (i.e., taps with a propagation time much slower than the average) placed between two consecutive DSP blocks [18,19]. On the other hand, the average propagation delay offered by the DSP-based TDL is faster compared to the carry-based one; this forces the use of more DSP blocks in series, resulting in a high count of ultra-bins. To further improve linearity at the expense of resolution, the scientific literature also presents net-based TDLs implemented using routing logic (i.e., net) [20]. In addition to offering a lower average propagation delay per tap compared to carry-based and DSP-based TDLs at the same technological node, these structures are more complex, in terms of placement.

### *1.2. Tapped Delay Line TDC Working Principle*

As Figure 1 shows, the TDL TDC uses a digital low-to-high step signal (referred to as START) that passes through a series of buffers, called taps or bins, characterized by a specific propagation delay. The output of each tap of the TDL is connected to the D input of D-type Flip-Flops (DFFs). A digital low-to-high step signal (called STOP) is used as the clock for the DFFs. In this way, the propagation of the START signal through the TDL, with a quantization error proportional to the propagation delays, is captured at the Q outputs. This method yields a sequence of consecutive high-logic values known as a thermometer code, which is proportional to the duration of the time interval delimited by the START and STOP signals. For easier management of the time interval (i.e., numerical representation of the elapsed time between the START and STOP events) captured by the DFFs, the thermometer code is compressed into pure binary, using a thermometer-to-binary converter, referred to as a decoder or encoder in the literature. The two main architectures used are the "sum1s" approach [21,22], which counts the number of 1s in the thermometer code, and the "Log2" method [23], also known as "one-hot" [24], which detects the transition from 1 to 0 (the "one-hot" decoder presented in [24] searches for the most significant 0–1 transition, effectively performing the base-2 logarithm "Log2" of the thermometer code as presented in [23]).

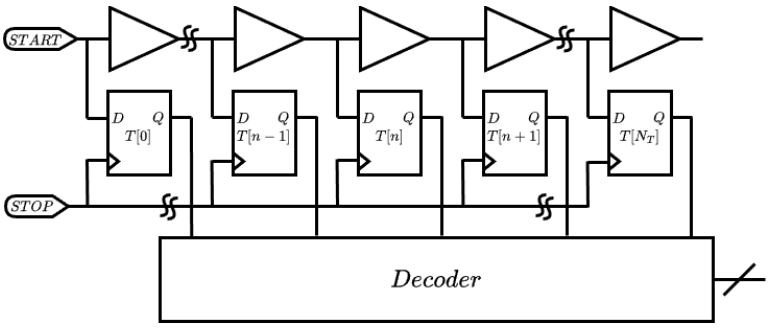



**Figure 1.** Schematic view of TDL TDC implementation, where each buffer output is sampled by a DFF followed by a decoder.

To be more precise, if all the $N_T$ taps that compose the TDL have the same propagation delay $\overline{\tau_p}$ then the time interval $\Delta T$ can be calculated simply by using (1), where $n$ is the output of the thermometric-to-binary converter:

$$\Delta T = T_{STOP} - T_{START} = n \times \overline{\tau_p} \quad (1)$$

Referring to Equation (1), it is evident that the resolution (LSB) of the TDL TDC is $\overline{\tau_p}$, the quantization noise $\overline{\tau_p}/\sqrt{12}$ (i.e., $\varepsilon_Q^2 = LSB^2/12$) [25,26] (the digitization process introduces a quantization error ranging between $-\overline{\tau_p}/2$ and $+\overline{\tau_p}/2$, with a uniform distribution characterized by a variance of $\overline{\tau_p}^2/12$, thereby defining the measurement precision), and the Full-Scale Range (FSR) $N_T \times \overline{\tau_p}$. Therefore, in the presence of jitter (i.e., $\sigma_j$) between the START and STOP signals, due to electronic system noise, the measurement precision (i.e., $\sigma_{\Delta T}$) of the measured time interval (i.e., $\Delta T$) results [27]:

$$\sigma_{\Delta T}^2 = \varepsilon_Q^2 + \sigma_j^2 = \frac{\overline{\tau_p}}{12} + \sigma_j^2 \tag{2}$$

### *1.3. Tapped Delay Line Calibration*

However, due to strong Process, Voltage, and Temperature (PVT) fluctuations, this linear approach does not properly fit FPGA technologies. In fact, each tap has a dispersed propagation delay (i.e., $\tau_p[k]$ with $k \in [0; N_T - 1]$) that strongly differs from the average $\overline{\tau_p}$. PVT fluctuations, if high resolution, precision, and linearity are required, necessitate calibrated operation. The most efficient algorithm is known in the scientific literature as "bin-by-bin" calibration, in which, thanks to a Code Density Test (CDT), the propagation delay of each tap is estimated with a negligible error and stored in a so-called Calibration Table (CT) [14,21,28], i.e., $CT[k] \backsimeq \tau_p[k]$ with $k \in [0; N]$. Thus, with proper integration of the CT, the Characteristic Curve (CC) is generated, which assigns a timestamp correcting PVT fluctuations to each decoded thermometric code $n \in [0; N_T - 1]$:

$$\Delta T = T_{STOP} - T_{START} = CC[n] \tag{3}$$

Under this condition, the resolution (i.e., LSB) is better-represented by the distribution of the CT and, roughly, by the average propagation delay $\overline{\tau_p}$ (i.e., $\overline{\tau_p} = \sum_k \tau_p[k]/N_T \backsimeq \sum_k CT[k]/N_T$), while the precision is represented by the so-called Equivalent LSB ($LSB_{EQ}$) [26], whose mathematical expression is expressed in Equation (4). Thus, due to the propagation delays distortion, the quantization error is proportional to the $LSB_{EQ}$ (i.e., $\varepsilon_Q^2 = LSB_{EQ}^2/12$) rather than the resolution (i.e., $\varepsilon_Q^2 \neq \overline{\tau_p}^2/12$):

$$LSB_{EQ}^2 = \frac{\sum_k \tau_p^3[k]}{\sum_k \tau_p[k]} \backsimeq \frac{\sum_k CT^3[k]}{\sum_k CT[k]} \tag{4}$$

### *1.4. Tapped Delay Line Decoding*

Considering the fact that PVT fluctuatuons can be compensated by calibration, the main criticality in TDL TDC is the Bubble Error (BE) [29–32], which is a switching effect in the thermometer code's uniform pattern. Instead of the output being a continuous string of high levels followed by zeros, which is typical of a proper thermometer code, the DFFs and their connections have non-idealities and mismatches that could cause irregularities (i.e., one or more zeros that show up as bubbles in the continuity of ones). For example, in an 8-tap-long TDL, the output might be "11111010" rather than "11111110". BEs result from deterministic non-linearities in TDL propagation (e.g., skews) [33] or from stochastic processes (e.g., sampling mistakes due to setup and hold time violations) [34]. The presence of BEs strongly impacts the output of the thermometer-to-binary converter, reducing the resolution, precision, and linearity. Moreover, the behavior of the pure binary output is also influenced by the architecture of the decoder. If the "sum1s" approach is used, the BEs are compressed (e.g., "11111010" is interpreted as "11111100"), while the "Log2" method neglects the presence of bubbles (e.g., "11111010" is interpreted as "11111110"). Due to these opposite behaviors, "sum1s" and "Log2" are also known as "bubble compression" or "ones-counter" [21] and "one-hot" algorithms [24], respectively.

### *1.5. Tapped Delay Line Sub-Interpolation*

In order to achieve TDL TDCs characterized by better resolution compared to that offered by the average propagation delay of the FPGA technology node $\overline{\tau_p}$ (Table 1), sub-interpolation techniques [20,35,36] and other exotic TDL-based structures [37] have been introduced in the scientific literature. These techniques involve either repeating the measurement on multiple TDLs (i.e., $N_{TDL}$) placed in parallel [21] (i.e., spatial sub-interpolation), using only one TDL with each tap sampled twice (i.e., $N_{TDL} = 2$) by doubling the DFFs (i.e., dual sampling) [38], or cycling through multiple times (i.e., $N_{TDL}$) via feedback on the same TDL (i.e., temporal sub-interpolation) [39], in order to obtain a Virtual TDL (VTDL) characterized by propagation delays of $N_{TDL}$ being faster, thus providing an improvement in resolution. Mathematically, a VTDL of $N_{TDL}$ sub-interpolation order provides an improvement in $LSB_{EQ}$ by a factor between $1/\sqrt{N_{TDL}}$ and $1/N_{TDL}$, but it also results in an increase in jitter (i.e., $\sigma_j$) compared to the non-sub-interpolated one [36]. This increase in jitter is low in spatial sub-interpolation and dramatically high in temporal sub-interpolation, especially in scaled FPGAs (i.e., 28 nm or less). In this scenario, a trade-off arises between resolution (i.e., sub-interpolation order, the number of TDLs in parallel in spatial sub-interpolation for modern FPGAs), precision (i.e., the increase in jitter), and efficiency (i.e., area and power used by the sub-interpolation). One of the most common techniques of temporal sub-interpolation is Wave Union A (WUA), while among the temporal techniques Wave Union B (WUB) stands out [40]. In WUA, initially two [40] and later even more [41,42] edges are propagated on the same TDL for each event for which a timestamp is desired. Meanwhile, in WUB, the TDL is closed in feedback [40] or connected to a ripple generator [39,43], constructing a sort of multivibrator that allows the event for which a timestamp is desired to be recirculated multiple times in the TDL; the number of re-circulations thus corresponds to the order of sub-interpolation. In this sense, with WUA and WUB using a single TDL an order of sub-interpolation equal to the number of propagated edges (WUA) or the number of re-circulations (WUB) is obtained at the cost of greater decoding complexity, which increases with the number of edges/re-circulations. Moreover, even if we focus solely on precision, as the order of sub-interpolation increases, the jitter associated with each edge/re-circulation also increases, making these techniques less effective at high orders of sub-interpolation.

Research has thus been directed towards spatial sub-interpolation, first by aligning multiple TDLs in parallel [44,45], each propagating a measurement edge (a.k.a. merged TDL) [46,47], and then moving towards the Super Wave Union (SuperWU), where a WUA is performed on each TDL to double the order of sub-interpolation [36]. Among these two techniques, due to a good trade-off between resolution, precision (i.e., reduction of quantization error), and implementation complexity (i.e., complexity of the decoding mechanism for the various edges), the merged TDL has become increasingly popular.

In the scientific literature, on the other hand, it is possible to find other and more efficient TDL-based architectures for increasing resolution, if viewed solely from the perspective of quantization error; among the main ones are the multisampling TDL [48], the Pseudo-Segmented Delay Line (PSDL) [37], and Multiple Time Coding Lines (MTCL) [33]. In multi-sampling TDL [48], a WUA with multiple edges is present not only in the START signal (i.e., the one propagated in the TDL) but also in the STOP signal (i.e., clock-off DFFs), thus allowing the order of sub-interpolation to be amplified but also increasing the decoding complexity. While the PSDL [37] and MTCL [33] resemble multi-TDL techniques, as several TDLs operating in parallel are appropriately temporally offset to minimize quantization error, the dimensioning of the offset implies, compared to the classic multi-TDL, an increase in complexity from the point of view of the placing algorithm.

In the tuned delay line [27], instead, the goal is not only to reduce the quantization error but to create an extremely linear TDL by appropriately choosing the taps among the available carry logic outputs. Whereas, in [49] the Multi-Segment digital TDL is presented, where the same goal is achieved by arranging the taps from multiple TDLs placed in parallel, at the cost of significant computational effort, to identify the optimal combination.

Another matter is the best placement of TDLs within the FPGA to mitigate signal jitter proportional to device occupation density [50].

**Table 1.** Average tap delay per device family.

| FPGA Family | Node (nm) | $\overline{\tau_P}$ (ps) [1] | Tested Device | Ref. |
|---|---|---|---|---|
| Cyclone I | 130 | 70 | EP1C20F400C6N | [36] |
| Cyclone II | 90 | 45 | EP2C35F672C6N | [36] |
| Virtex 5 | 65 | 34 | XC5V50T | [36] |
| Spartan 6 | 45 | 25 | XC6SLX9 | [36] |
| Spartan 6 | 45 | 20 | XC6SLX75 | [26] |
| Kintex 7 | 28 | 11 | XC7K160T | [33] |
| Kintex 7 | 28 | 17 | XC7K325T | [36] |
| Artix 7 | 28 | 15 | XC7A200T | [36] |
| Kintex UltraScale | 20 | 4.6 | XCKU040 | [33] |
| Kintex UltraScale | 20 | 5 | XCKU040 | [12] |

[1] Differences values of $\overline{\tau_P}$ for the same FPGA family could depend on the speed grade (not specified in the papers) as well as on different experimental setup conditions (e.g., varying measurement temperatures) and the positioning of the TDL within the FPGA.

### *1.6. Nutt Interpolation*

As mentioned previously, while TDL TDCs are able to provide good precision and resolution, they suffer from a trade-off between the FSR and area occupancy [51]. This is quite limiting, since the tap propagation delay in modern nodes is in the few-picoseconds range. To break this trade-off, a common method is to employ Nutt interpolation [52]. This technique, as Figure 2 shows, for all channels (e.g., START and STOP) splits the measurement into two parts: the Coarse part measured by an $N_{CC}$-bit width Coarse Counter clocked at $T_{CLK}$ (i.e., $N_{cc1}$ for the START and $N_{cc2}$ for the STOP), and the Fine part performed by the TDL (i.e., $T_{fine1}$ for the START and $T_{fine2}$ for the STOP). Given that the START signal of the event may happen at any time and is not synchronous with the system clock, the measured time in this particular configuration will be

$$T_{meas} = T_{fine1} + (N_{cc2} - N_{cc1}) \times T_{CLK} - T_{fine2} \tag{5}$$

In this way, if the dynamic range of the TDL exceeds $T_{CLK}$, the FSR of the integral system is extended up to $2^{N_{CC}} \times T_{CLK}$:

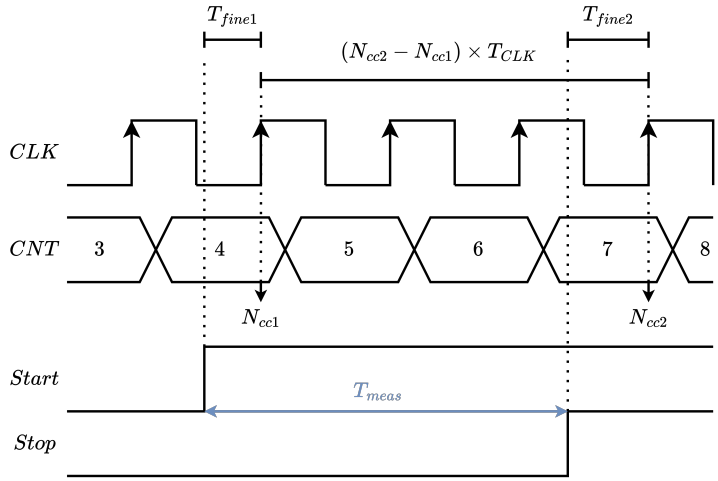



**Figure 2.** Timing diagram of system featuring the Nutt interpolation technique.

### *1.7. Purpose of This Contribution*

The purpose of these work was to optimize from an LSB ($LSB_{EQ}$)-and-precision point of view the structure reported in Figure 3, considering Xilinx 20 nm UltraScale

FPGA devices as targets. This optimization process aims to reduce the performance gap between FPGA-based and ASIC-based TDCs, with ASICs being better-performing and often preferred in cutting-edge applications (e.g., quantum technologies, life sciences, and nuclear physics) that require tens of channels within a square centimeter footprint and picosecond precision. The ultimate goal was to make FPGA-based TDCs usable in cutting-edge applications, as they are more suitable for R&D and rapid prototyping. They offer greater flexibility, lower costs for small production volumes, and a shorter time-to-market compared to ASIC-based solutions.

The research led to the development and comparison of multiple TDL TDC solutions implemented on a Kintex UltraScale device (i.e., XCKU040-2FFVA1156E) housed on a KCU105 general-purpose evaluation board (EVB).

We aimed to implement a TDC in a Xilinx 20 nm Kintex UltraScale FPGA with moderate area consumption capable of handling dozens of channels in a single device and equipped with a real-time decoding-and-calibration algorithm executed directly on the FPGA; thus, we selected the carry-based TDL topology with merged TDL. This choice optimized the trade-offs between linearity, average propagation delay, minimization of quantization error, area occupancy, and place-and-route complexity.

Figure 3 summarizes the architecture of a modern FPGA-based spatial sub-interpolated and Nutt-interpolated TDL TDC.

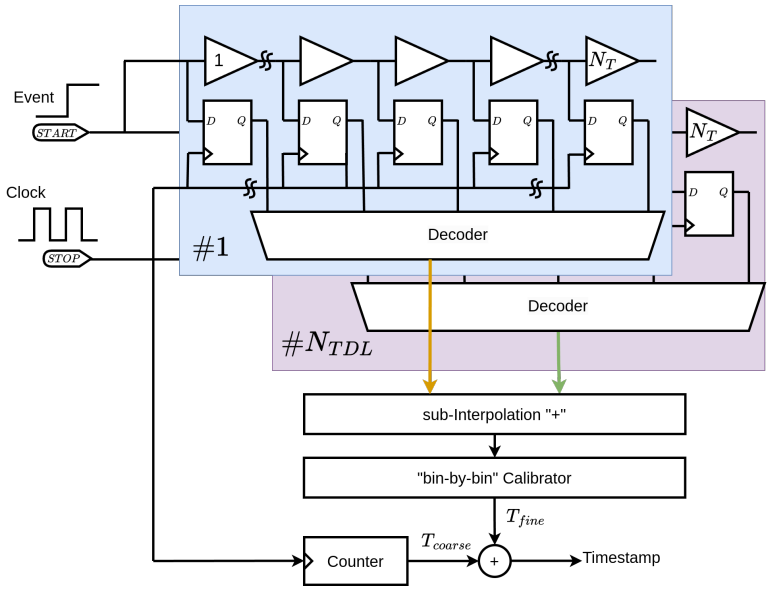



**Figure 3.** Modern FPGA-based spatial sub-interpolated and Nutt-interpolated TDL TDC.

From this optimization process, two main solutions emerged: one with high precision and another with low area utilization, both offering more than a 4% improvement in precision over the state of the art for this technology node. The first solution achieved an SSP of 2.64 ps r.m.s., with Differential and Integral Non-Linearity (DNL/INL) errors of 0.523 ps and 16.939 ps, respectively. This design occupied 883 CLBs and 126 kb of BRAM (i.e., 3.5 blocks), allowing up to 24 channels to be implemented on the selected 35 × 35 mm FPGA. The second solution, with an SSP of 3.75 ps r.m.s., a DNL of 0.599 ps, and an INL of 7.151 ps, occupied only 259 CLBs and 72 kb of BRAM (i.e., 2.5 blocks), which enabled up to 64 channels to be implemented on the target FPGA.

### *1.8. Paper Organization*

This paper is organized as follows: Section 2 discusses all the relevant UltraScale device features from a TDL TDC point of view. As shown in Section 3, multiple TDL TDC solutions were implemented and tested. A comparison with a state-of-the-art FPGA-based TDL TDC implemented in Xilinx 20 nm UltraScale FPGA is conducted in Section 4.

## 2. TDL TDC Architecture in 20 nm Xilinx UltraScale FPGA

In this Section, to discuss both the technological and architectural properties of the 20 nm Xilinx UltraScale FPGAs, we must take a bottom-up approach, starting from the CLB, the carry logic primitive (i.e., CARRY8), the clock distribution scheme in Section 2.2, and then addressing the CARRY8-based TDL itself in Section 2.3, as well as its specific placements inside the device. Lastly, the decoding policies are summarized in Section 2.4.

### *2.1. Configurable Logic Block Primitives*

In UltraScale, each CLB has a CARRY8 primitive with 8 outputs (numbered 0 to 7): for each output, two distinct signals are available, the CO and the O, which, for our purpose, were each the negation of the other. Each of these, now, 16 outputs is sampled by the corresponding lettered DFF (i.e., CO[0] to AFF, O[0] to AFF2). The CLB is split into two parts: the bottom part containing 8 DFFs (A to D and A1 to D1), and the top part containing 8 DFFs (E to H and E1 to H1).

The CARRY8 thus functions as an 8-tap TDL that propagates the input signal from the bottom to the top of the CLB (i.e., from CO[0] and O[0] to CO[7] and O[7]). Similar to previous Xilinx technology nodes (i.e., 28 nm, 40 nm, and 45 nm) [27], if taken independently then all CO and O outputs have a similar delay between each tap [12,53]; as such, to build a TDL we only have to sample one of them; however, in the same CARRY8, the first 4 taps and the last 4 taps may have different skews. This can be observed in UG574 [54] and in Figure 4. By running a Vivado Post-Implementation timing analysis, we can see how this independent clocking structure has an effect on the clock skew inside the CLB, as shown in Figure 5:

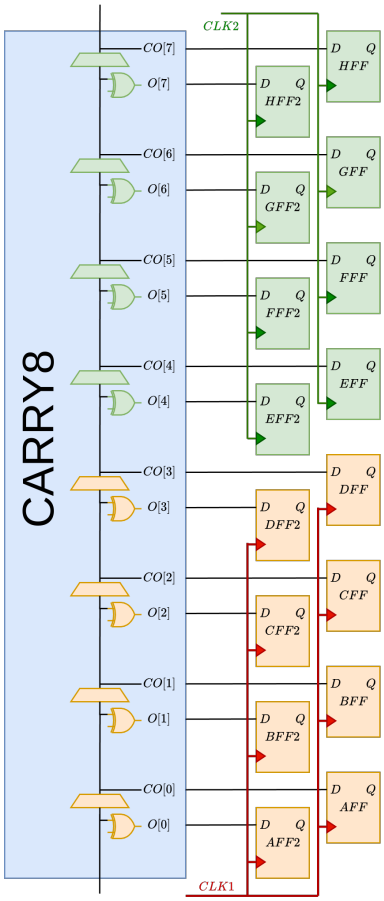



**Figure 4.** Independent clocking scheme of the two halves (i.e., red and green) of the CLB with DFFs and CARRY8 [54].

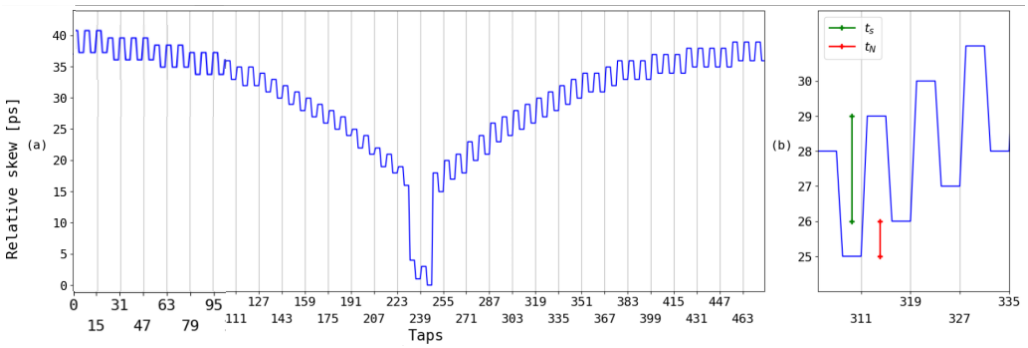



**Figure 5.** Clock skew pattern among taps, where the horizontal lines identify each CLB, with (**a**) increasing values starting from insertion point (roughly in correspondence to tap 247) and (**b**) different propagation times of the clock into the CLB. In (**b**), we can also see the local CLB skew ($t_s$) and the skew between the CLBs ($t_N$).

Observing Figure 5, we can see that the clock skew exhibits two main patterns:

- an inter-CLB trend, which is controlled by the clock distribution network, where we see a delay accumulation (i.e., $t_N$) CLB by CLB from the clock insertion point (roughly in correspondence to tap 240 in Figure 5);
- an intra-CLB trend, a local skew pattern (i.e., $t_s$), with the clock arriving at the second half of the primitive earlier than the first.

This allows us to model the clock distribution network of the CARRY8 primitive (Figure 6). To put it numerically, as also shown previously in Figure 5, the $t_s$ and $t_N$ values are below a few ps, e.g., $t_s \backsimeq 3$ ps and $t_N \backsimeq 1$ ps.

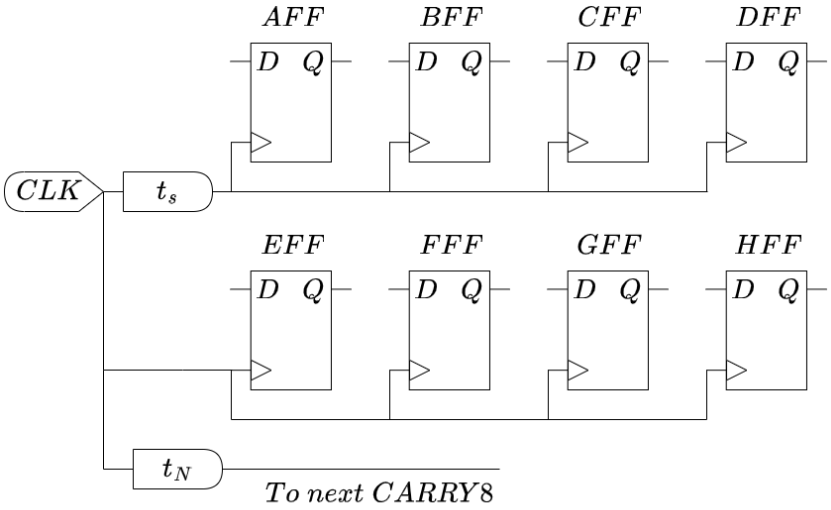



**Figure 6.** Model of the clock skew from the second half to the first one, represented by $t_s$, and from the second half to the second one, represented by $t_N$.

These figures must, however, be taken into account in light of the TDL, particularly with regard to the tap (C or O) propagation delay (i.e., $\tau_P[k]$ with $k \in [0;7]$). Indeed, in cases when the tap propagation delay falls short of $t_N$, we can run into skew-induced BEs. More logically, because there is a non-negligible amount of skew, the output code is compiled "progressively" rather than "simultaneously". In general, if the tap propagation delay is less than the skew between two sampling DFFs, it is possible that the values, which propagate asynchronously, will skip a few taps, as illustrated graphically in Figure 7, where $C[k]$ represents the CARRY8 outputs (i.e., the $D$ pins of the DFFs), $CK[n]$ represents the $C$ pins of the DFFs, and the tap propagation delays are represented by $\tau_p[k]$:

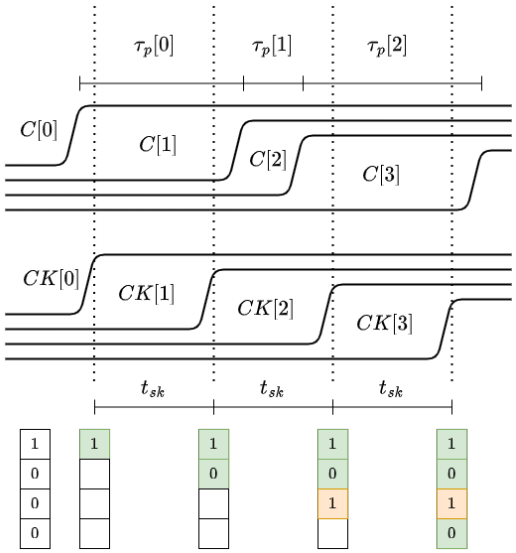



**Figure 7.** Generation mechanism of the BEs. In particular, showing the situation in which a single tap has a smaller propagation delay ($\tau_p$) than the clock skew between the sampling FFs ($t_{sk}$).

In our case, this is the situation where a signal is traveling in the second half of the CLB, and after being sampled in that area it keeps propagating in the next one, having a skew of $t_s + t_N$.

### *2.2. Clock Regions and the Clock Region Crossing Issue*

Xilinx FPGAs are organized in Clock Regions (CRs), that are groups of CLBs, characterize by internal negligible skews; CRs are spaced to adjacent ones by means of Clock Region Crossing (CRC), and they are connected by proper clocks lines characterized, across the CRC, with a clock skew that is more than an order-of-magnitude higher, with respect to $t_s$ and $t_N$. As illustrated in Figure 8, the clock signal is distributed among different CRs vertically via the clock distribution network from the backbone (green) and horizontally via horizontal lines (violet). Meanwhile, as described in Section 2.1, within the CR, the inter-clock region routing (red) is used. Further details are available in UG94 [55].

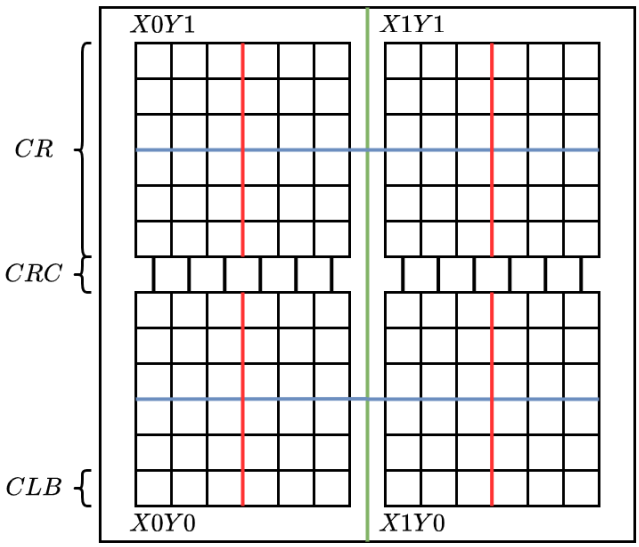



**Figure 8.** Schematic view of a generic Xilinx FPGA, subdivided into CRs (i.e., X0Y0, X0Y1, X1Y0, and X1Y1) spaced by CRC. The connection in between the clock region is shown, as well as the clock distribution network from the backbone (green) to the horizontal lines (violet) to the inter-clock region routing (red).

The skew diagram from Figure 5 has been expanded across the CR, and it is shown in Figure 9, to better represent this issue. This is especially critical, because at the CRC the clock skew (i.e., $t_{CRC}$) is ≈35 ps, which is guaranteed to result in BEs, if negative, and ultra-bin, if positive.

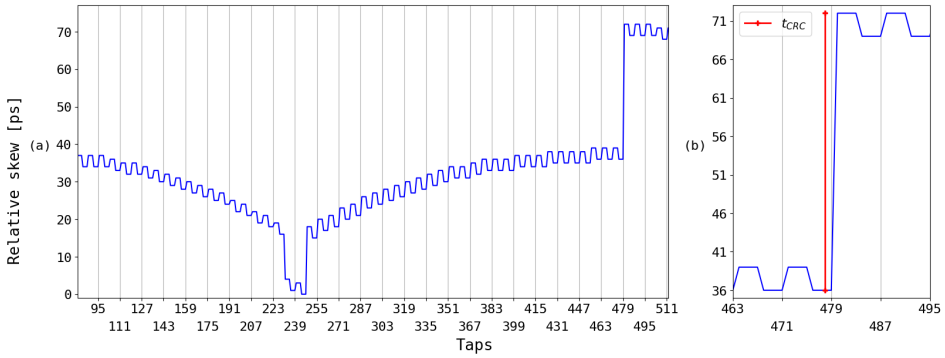
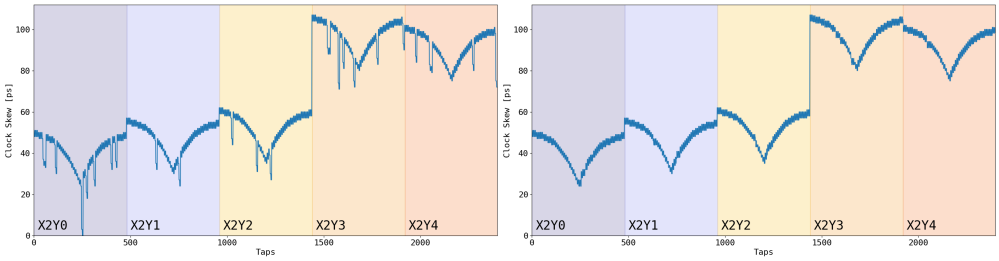



**Figure 9.** Clock skew pattern along two CRs (i.e., the CRC is between tap 479 and 480), with (**a**) showing the overall trend and (**b**) showing the sudden increase in clock skew ($t_{CRC}$).

Another noteworthy aspect to investigate was the absence of continuity in the clock skew diagram within the CLB (taps 223–255 in Figures 5 and 9) compared to the 28 nm 7-Series [33]. We repeated the tests in different CLBs (Figure 10), consistently observing the absence of continuity. Considering only data collected in post-implementation firmware, we hypothesize that this was due to the non-ideal nature of the FPGA fabric. This discontinuity contributes to the creation of an ultra-bin in the TDL. The discontinuities observed within the CLB, unlike what happened with the TDC in the 7-Series [33], were detected only when the FPGA was programmed with the TDC firmware. Moreover, such discontinuities, when varying the implementation of the TDC (i.e., order of sub-interpolation and type of decoder), remain almost identical in magnitude although they may manifest with slightly different numbers and positions. Another important point is that, from an architectural standpoint, not only the division of the CLBs underwent changes between the 28 nm 7-Series [56] and the 20 nm UltraScale [54] but also the clock distribution network [57]. Consequently, the Xilinx 20 nm technology is not simply a scaled-down version of the 28 nm but has its own architecture, both in terms of clock distribution and fabric organization.




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

**Figure 10.** Clock skew pattern along many CRs with TDC firmware (**a**) and without (**b**).

### *2.3. Tapped Delay Line*

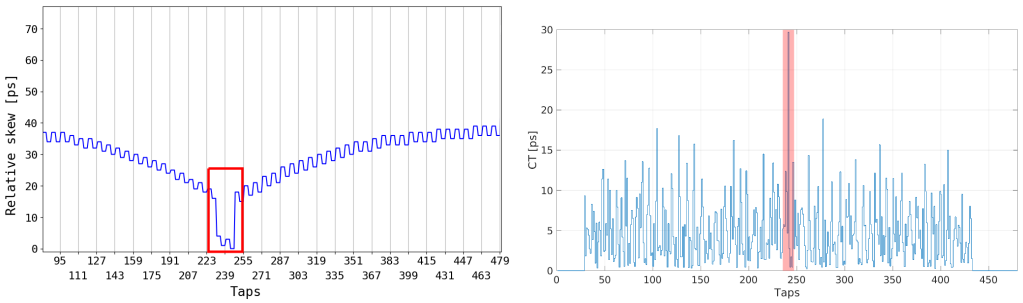
A complete TDL is created by concatenating the CARRY8 primitives. Therefore, the mean tap of delay $\overline{\tau_P}$ multiplied by the total number of taps provides a first-order estimate of the FSR of the TDL. Additionally, to overcame the CRC issue exposed in Section 2.2, it is necessary to fit the TDL inside the CR. In this context, the longest TDL segment that can fit inside a CR is a Figure of Merit (FoM), in relation to the FPGA family. However, since the CARRY8 primitive (i.e., that offers only $N_C = 8$ taps) can only be concatenated vertically (i.e., from the bottom to the top), the maximum length depends on the number of CLBs that are present vertically in a CR (i.e., $N_{CLB} = 60$) [54]. This restricts the span of the TDL

to up to 480 taps (i.e., $N_{CLB} \times N_C$) in Xilinx 20/16 nm UltraScale and UltaScale+ FPGA. In our system, considering that the FSR of the TDL is constrained by the Nutt interpolation, we can estimate the maximum clock period $T_{CLK}^{MAX}$, as shown in Equation (6), assuming, in the first approximation, that the tap propagation delay is uniform and equal to the average value $\overline{\tau}_p$ (i.e., 5 ps from Table 1):

$$T_{CLK}^{MAX} \leq N_{CLB} \times N_C \times \overline{\tau}_p = 60 \times 8 \times 5\,\text{ps} = 2.4\,\text{ns} \quad (6)$$

We require at least $T_{CLK}^{MAX} < 2.4\,\text{ns}$. Adding an engineering margin factor, we finally settle on $T_{CLK}^{MAX} = 2\,\text{ns}$ (i.e., 500 MHz, in terms of frequency) and a TDL composed of 512 (since the decoding block is modular to powers of 2, we approximate to the next-higher power) taps (of which the last 32 will never be reached), because it is well within the device's capability, which is defined between 630 MHz and 850 MHz, depending on the speed grade of the device [58]. The high clock frequency available made it possible to implement a TDL sampled from the same clock source without requiring the use of multi-clock techniques, thus avoiding architectural complications [26].

Once the vertical length and placing has been addressed, there may be some factors to take into account while determining its precise horizontal placement. The FPGA does, in fact, display a non-uniform structure, cycling between CLB, RAM, and DSP columns. In this regard, referring to the discontinuities at taps 223–255 in Figures 5 and 9, we can observe the presence of an ultra-bin in the same position (i.e., tap 241) of the CT (Figure 11) (for this reason, we attributed the skew to the interaction between the firmware and the non-uniform structure of the FPGA, and we hypothesize that the highlighted skew is the cause of the ultra-bin). Since such discontinuities are too temporally close, any TDL that excludes them would result in a $T_{CLK}^{MAX}$ that would be too small, and, therefore, physically could not be implemented.




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

**Figure 11.** CR's clock skew pattern with discontinuity at taps 223–255 highlighted (**a**) and the relative CT with an ultra-bin at tap 241 highlighted (**b**).

### *2.4. Decoder*

In light of the TDL TDC architecture that we previously discussed, the thermometric code spreads down the TDL before being sampled by the DFFs. According to this theory, if there are no BEs, the code's one-to-one ratio corresponds exactly to the event's timestamp. As a result, the thermometric code sampled by a $N = 2^{N_B}$-long TDL can be remapped into a $N_B$-long unsigned integer that indicates which has been hit. We will now compare the two main decoder architectures.

From the scientific literature, considering only one TDL, whether it be Log2 or sum1s, the decoders are always $N_B$-stage pipeline structures that proceed dichotomously. The input stage has a number of single-bit inputs $N$ equal to or bigger than the size of the TDL, which is 480. For this reason, in this paper, considering that 480 is not a power of two, the decoders are designed with 9 pipeline stages (i.e., $N_B = 9$) and 512 single-bit inputs (i.e., $N = 512$), where the last unused 32 single-bit inputs (i.e., $N - N_T = 512 - 480$) are connected to "0".

To perform the sub-interpolation over more TDLs (i.e., $N_{TDL}$), $N_{TDL}$ identical decoders (i.e., all sum1s or all Log2) are required. The $N_{TDL}$ outputs are summed together, using a tree adder with $\lceil log_2(N_{TDL}) \rceil$ pipeline stages (i.e., 1 for $N_{TDL} = 2$, 2 for $N_{TDL} \in \{3;4\}$, 3 for $N_{TDL} \in \{5;6;7;8\}$).

#### 2.4.1. The sum1s Decoder

This decoding scheme is the easiest and most direct, especially if there are no BEs, and it consists in counting the instances of "ones" inside the thermometric code. Nevertheless, if BEs are present, the sum1s approach applies bubble compression, meaning that several codes with the same number of "ones" may correlate to the same output. Figure 12 is an illustration of this:

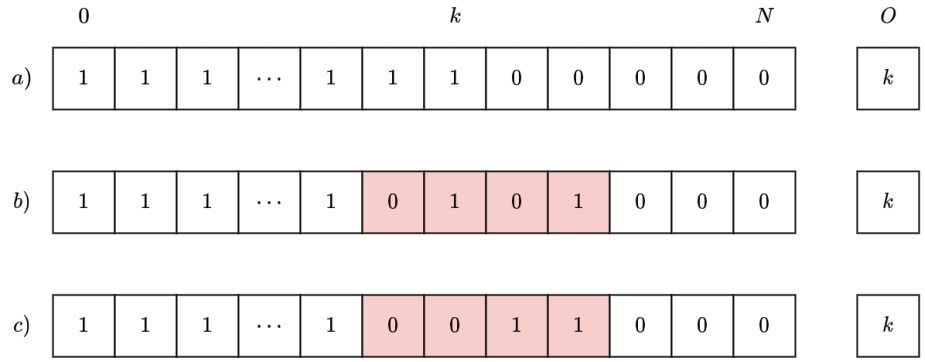



**Figure 12.** Example of bubble compression where codes a, b, and c return the same output (red highlights indicate BEs).

The sum1s decoder is usually organized as tree adders (unlike [21], in this paper this module is described in VHDL behavioral modeling style, and it is left to Vivado to assign the LUTs and DFFs appropriately), where the stage $i$-th (with $i \in [1;9]$) has $2^{N_B-(i-1)}$ inputs of $i$ bits and $2^{N_B-(i-1)}/2$ outputs of $i+1$ bits (i.e., $2^{9-(i-1)}$ inputs of $i$ bits and $2^{9-(i-1)}/2$ outputs of $i+1$ bits in this implementation). In Figure 13, an example is shown, with $N_B = 3$ (i.e., $N = 8$).

With an input thermometer code of $N$ bits where the LSB is always "1" the output code is an $N_B + 1$-bit number ranging from 1 to $N$. Therefore, typically, to save area, thermometer codes are remapped, such that their conversion into pure binary results in an $N_B$ bit number ranging from 0 to $N$. In the example shown in Figure 13, we thus obtain "100" (i.e, decimal 4) as the output binary instead of "0101" (e.g., decimal 5):

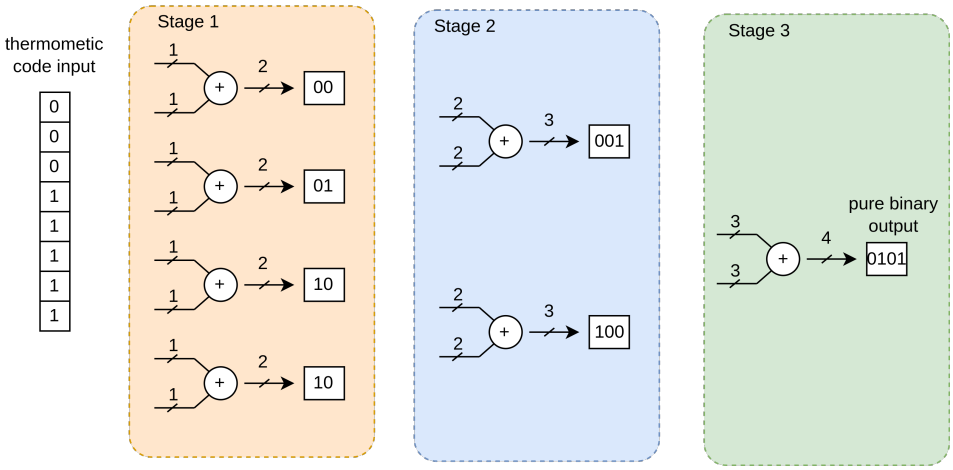



**Figure 13.** Example of a sum1s decoder with $N_B = 3$ (i.e., $N = 8$) and its corresponding data flow for decoding the thermometer code "0001111" into pure binary "0101" (i.e., decimal 5).

### 2.4.2. The Log2 Decoder

Building on the presumption shared with the sum1s decoder, it is clear that the population of the most significant bit (MSB) in the TDL is correlated with the total number of "ones" in the thermometric code. With this knowledge, a more effective binary search strategy can be used to identify the MSB, as opposed to thoroughly going over the full code sequence. In this way, BEs are ignored. Figure 14 shows how the Log2 Decoder operates:

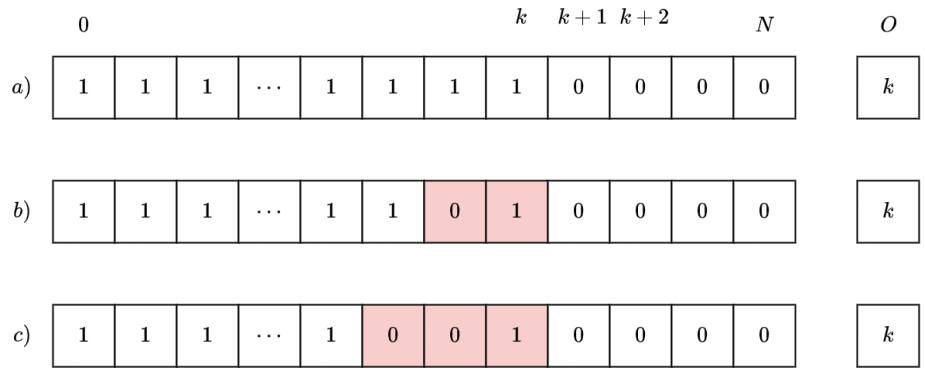



**Figure 14.** Example of insensitivity to BEs in Log2 decoder where a, b, and c will have the same output (red highlights indicate BEs).

The stages, in the case of Log2 decoders, are generally characterized by $2^{N_B-(i-1)}$ inputs and $2^{N_B-(i-1)}/2$ outputs (i.e., $2^{9-(i-1)}$ inputs and $2^{9-(i-1)}/2$ outputs in this implementation), which propagate the portion of the TDL where the most significant 1–0 transition occurs in the next stage. So, during the propagation the pure binary output is compiled from the most significant bit to the least significant bit, resembling a successive-approximation register. In Figure 15, an example is shown, with $N_B = 3$ (i.e., $N = 8$):

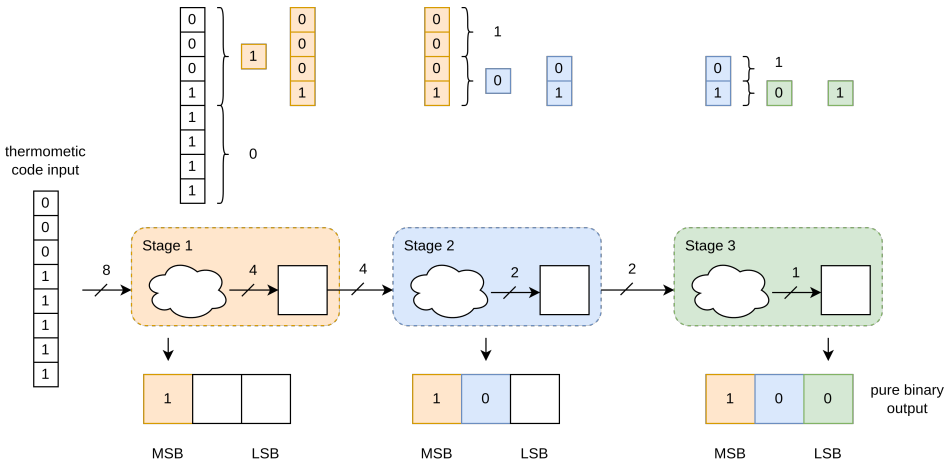



**Figure 15.** Example of a Log2 decoder with $N_B = 3$ (i.e., $N = 8$) and its corresponding data flow for decoding the thermometer code "0001111" into pure binary "100" (i.e., decimal 4).

### 2.4.3. A Brief Comparison

If we consider multi-channel systems, which are nowadays the most requested in cutting-edge applications, efficient resource utilization is a critical FoM for the design. In the FPGA context, resource usage is typically measured in terms of the number of CLBs consumed by the design. Each CLB contains a limited number of Look-Up Tables (LUTs) (eight, in this case), which are basic combinational elements capable of implementing any arbitrary binary function, DFFs (16 per CLB), and a CARRY8 element (1 per CLB). To provide a clear comparison, we evaluated the resource utilization of an $N_B = 8$ Log2 and sum1s decoders for 480-tap-long TDLs (i.e., $N = 512$). This comparison is summarized

in Table 2, which presents the maximum clock frequencies ensuring timing enclosure (i.e., Max. Freq.) [59], as well as the area utilization computed post-implementation by Vivado [60]. The area utilization is detailed not only by counting the number of occupied CLBs but also by reporting the number of individual LUTs and DFFs [61]. It is evident that if spatial sub-interpolation is required ($N_{TDL} > 1$) then the area occupancy scales linearly for both solutions. Although the Log2 decoder demanded fewer resources compared to the sum1s decoder, it performed less well, in terms of timing.

**Table 2.** Comparison of resources necessary for sum1s and Log2 decoders.

| $N_{TDL}$ | Type | LUTs | DFFs [1] | CARRY8 [2] | CLB | Max Freq. [MHz] [3] |
|---|---|---|---|---|---|---|
| 1 | sum1s | 808 | 1021 | 18 | 160 | 664 |
| | Log2 | 602 | 593 | 24 | 98 | 527 |
| 2 | sum1s | 1620 | 2042 | 38 | 288 | 704 |
| | Log2 | 1209 | 1185 | 50 | 183 | 500 |
| 4 | sum1s | 3247 | 4084 | 78 | 568 | 593 |
| | Log2 | 2423 | 2366 | 102 | 450 | 541 |
| 8 | sum1s | 6502 | 8168 | 158 | 979 | 560 |
| | Log2 | 4885 | 4719 | 206 | 839 | 500 |

[1] These DFFs do not take into account the 512 DFFs used to sample the 512-taps-long TDL. [2] These CARRY8 primitives do not count the 64 used to implement the 512-taps-long TDL (i.e., 64 × 8), but refer to the CARRY8 blocks used to perform addition operations within the decoder. [3] This parameter is calculated as the reciprocal of the slowest path.

## 3. Experimental Results

Several 2-channel (i.e., START vs. STOP) TDL TDCs characterized by different orders of sub-interpolation with different decoders (e.g., sum1s vs. Log2) were implemented on a Kintex UltraScale device (i.e., XCKU040-2FFVA1156E) hosted on a KCU105 general-purpose evaluation board, to identify the best-performing solutions, in terms of precision. The experimental setup used is described in Section 3.1, while the measurements to determine the optimal sub-interpolation order were carried out as described in Section 3.2. The precision and linearity measurements are reported in Sections 3.3, 3.4, and 3.5, respectively. Section 3.6 contains the final considerations.

### *3.1. Experimental Setup*

Figure 16 shows the experimental setup, where the KCU105 (the larger green EVB at the bottom-right) is depicted, connected via Low-Voltage Differential Signaling (LVDS) standard and SMA cables to START and STOP signals generated by an AWG-4014 Arbitrary Function Generator [62] from Active Technologies (black instrument in the center). Between the AWG-4041 and the KCU105 there was a board (the small green PCB in the center) composed of two HMC674LP3E comparators (with nominal jitter, i.e., <0.2 ps r.m.s.) tasked with converting the single-ended signal from the AWG-4041 into LVDS. As schematically depicted in Figure 16, the 2-channel (i.e., START vs. STOP) TDL TDCs could measure external signals coming from the AWG-4041 or internal signals generated by a simple logic hosted inside the FPGA, clocked by a clock (CLK1 in Figure 17) uncorrelated to that of the TDC (CLK0 in Figure 17).

Internal signals were used to evaluate quantization error and, thus, estimate the optimal order of sub-interpolation (Section 3.2), while external signals were used for the final validation (Sections 3.3–3.5), i.e., characterizing precision as a function of measured delay, single-shot channel precision, and linearity.

The timestamps used to perform the measurements were then acquired, and a histogram of the time difference between START and STOP was computed directly in real time inside the FPGA, using a proper hardware histogram module [63].

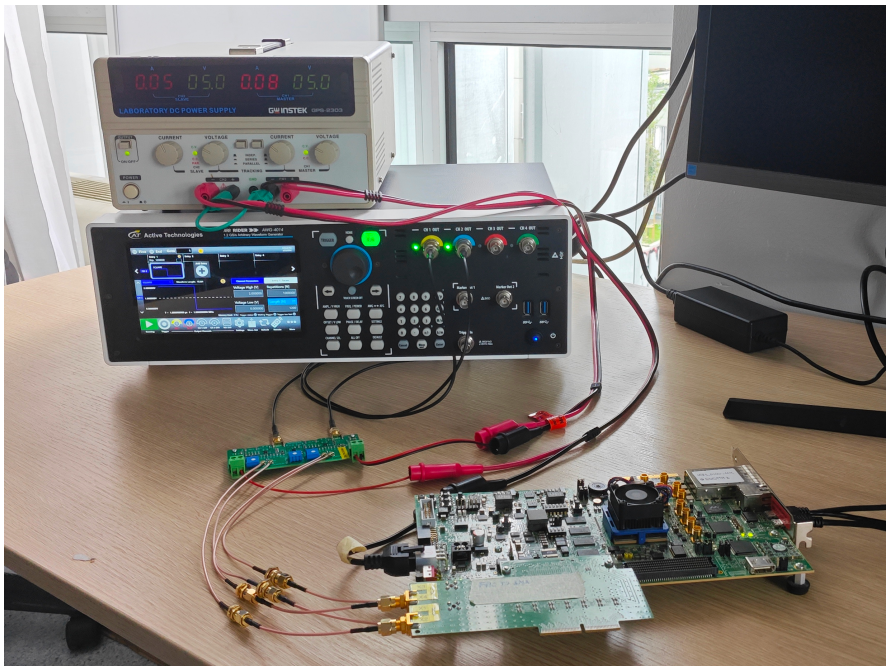
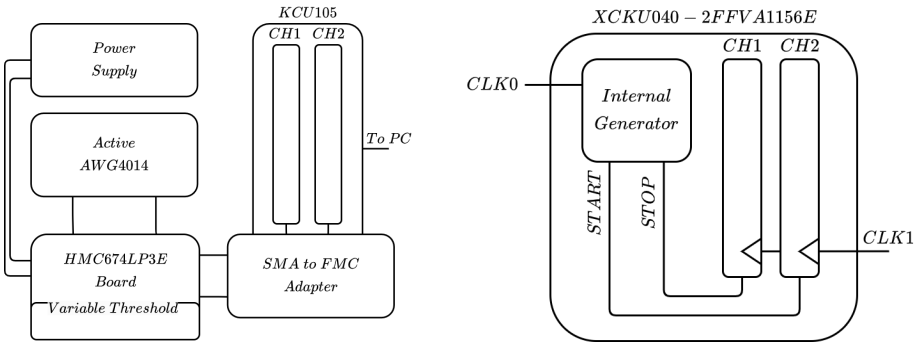


**Figure 16.** Experimental setup: the board had two independent oscillators (CLK0 and CLK1), ensuring there wwas no correlation between CLK0 and CLK1.




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

**Figure 17.** Schematic view of the experimental setup for the external signals (**a**) and for the internal ones (**b**).

### *3.2. Quantization Error and Sub-Interpolation Order*

In order to represent the quantization error that, as can be seen in (2), represented the best theoretical achievable precision, our first analysis used the computation of the $LSB_{EQ}$ from the CT, as shown in Equation (4). When we had 2 channels in our TDC, with START being the channel that received the first event and STOP being the channel that received the last event, then we had to consider both the quantization errors, as shown in Equation (7), i.e., $LSB_{EQ,START}/\sqrt{12}$ for START and $LSB_{EQ,STOP}/\sqrt{12}$ for STOP, respectively:

$$\varepsilon_Q^2 = \frac{LSB_{START}^2 + LSB_{STOP}^2}{12} \tag{7}$$

The quantization error (i.e., $\varepsilon_Q$) could not be directly measured, due to jitters; however, it could be estimated as the mean real precision of the measurements (i.e., $\overline{\sigma_{REAL}}$) performed with the internal START and STOP signals that offered only the internal jitter offered by the FPGA (i.e., $\sigma_j$):

$$\overline{\sigma_{REAL}^2} = \sigma_j^2 + \varepsilon_Q^2 \tag{8}$$

The findings of this analysis are provided in Table 3 for the sum1s decoder and in Table 4 for the Log2 decoder (considering the UltraScale, the average propagation delay on the TDL was about 5 ps, while due to the non-uniformity of the taps the calculation of the equivalent LSB fluctuated between 8.5 and 9.5 ps). The experiments were conducted with both decoders, as the BEs were handled differently at various spatial sub-interpolation orders (i.e., $N_{TDL}$). This clearly showed that the sum1s algorithm provided slightly better resolution performance for $N_{TDL} \geq 2$, indicating a slightly better handling of BEs compared to the Log2 algorithm in this family of FPGAs, when sub-interpolation was addressed. Therefore, the choice of decoder did not significantly impact the quantization error.

**Table 3.** Sum1s decoder quantization error overview.

| $N_{TDL}$ | $LSB^2_{START}$ [ps$^2$] | $LSB^2_{STOP}$ [ps$^2$] | $\varepsilon^2_Q$ [ps$^2$] | $\varepsilon_Q$ [ps] | $\overline{\sigma_{REAL}}$ [ps] |
|---|---|---|---|---|---|
| 1 | 89.65 | 71.48 | 13.43 | 3.66 | 3.95 |
| 2 | 28.92 | 25.41 | 4.53 | 2.13 | 2.78 |
| 3 | 23.30 | 18.22 | 3.46 | 1.86 | 2.52 |
| 4 | 5.78 | 6.12 | 0.99 | 1 | 1.8 |
| 6 | 4.37 | 4.81 | 0.76 | 0.87 | 1.68 |
| 7 | 1.96 | 1.63 | 0.3 | 0.55 | 1.55 |
| 8 | 1.11 | 0.99 | 0.18 | 0.42 | 1.48 |

**Table 4.** Log2 decoder quantization error overview.

| $N_{TDL}$ | $LSB^2_{START}$ [ps$^2$] | $LSB^2_{STOP}$ [ps$^2$] | $\varepsilon^2_Q$ [ps$^2$] | $\varepsilon_Q$ [ps] | $\overline{\sigma_{REAL}}$ [ps] |
|---|---|---|---|---|---|
| 1 | 78.59 | 78.26 | 13.07 | 3.62 | 3.94 |
| 2 | 25.85 | 29.09 | 4.58 | 2.14 | 2.75 |
| 4 | 6.16 | 7.33 | 1.12 | 1.06 | 1.98 |
| 8 | 1.28 | 2.01 | 0.27 | 0.52 | 1.77 |

Regarding the sub-interpolation order, as reported in Figure 18, both decoding architectures (sum1s on the left and Log2 on the right) exhibited the expected trend of the quantization error (blue) falling between a $1/\sqrt{N_{TDL}}$ error (red) and a $1/N_{TDL}$ error (light green) factor of improvement, and the real precision (orange).

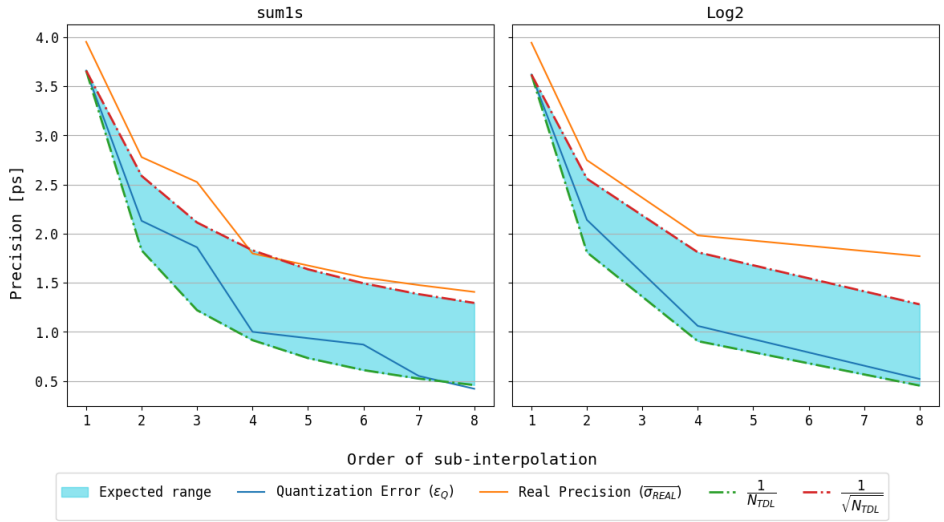



**Figure 18.** Behavior of quantization error (i.e., $\varepsilon_Q$ in blue) and real precision of measurements (i.e., $\overline{\sigma_{REAL}}$ in orange) as function of the sub-interpolation order $N_{TDL}$ considering sum1s (**left**) and Log2 (**right**) decoders, as reported in Tables 3 and 4, respectively.

### 3.3. Time Sweep

Using the hardware described in Section 3.1, with external signals, we conducted a delay sweep in the interval $[-2\,\text{ns}; 10\,\text{ns}]$ to accurately characterize the precision of our implementations considering different decoders and sub-interpolation orders, taking into account both the quantization error (i.e., $\varepsilon_Q$) and the jitter (i.e., $\sigma_j$) as a function of the measured delay. These measurements are reported in Figures 19–22, where four different orders of sub-interpolation (i.e., $N_{TDL} = 1$ none, $N_{TDL} = 2$, $N_{TDL} = 4$, and $N_{TDL} = 8$) are depicted. In the top part of the graph, the precision obtained with the sum1s decoder is shown in red, while that with the Log2 decoder is in blue; the bottom part shows the spread between the two approaches.

Referring to Equation (8), considering that AWG-4014 generated delay with START and STOP with a jitter $\sigma_{AWG}$, and considering that HMC674LP3E introduced jitter $\sigma_{HMC}$ both on the STOP and START signals, then we could derive the total precision ($\sigma_{SWEEP}$);

$$\sigma^2_{SWEEP} = \sigma^2_{AWG} + 2\sigma^2_{HMC} + \sigma^2_j + \varepsilon^2_Q \quad (9)$$

Moreover, to better highlight the trend of the precision over the sub-interpolation order, the results of Figures 19–22 (i.e., minimum, maximum, average precision, and the amplitude of the amplitude fluctuation of the precision) are also listed in Table 5. Observing Table 5, we demonstrate that the best precision was achieved using $N_{TDL} = 4$ and sum1s. However, when the minimum area occupancy was the target, observing Figure 20, we argue that the Log2 decoder was the best choice.

**Table 5.** Summary of performance: (**a**) sum1s; (**b**) Log2.

| (**a**) | | | | |
|---|---|---|---|---|
| $N_{TDL}$ | **Measured Precision $\sigma_{SWEEP}$ [ps]** | | | |
| | **Mean** | **Max** | **Min** | **Amplitude** |
| 1 | 7.79 | 8.23 | 7.35 | 0.88 |
| 2 | 6.07 | 6.87 | 5.3 | 1.57 |
| 3 | 6.39 | 7.85 | 5.42 | 2.43 |
| 4 | 6.01 | 7.06 | 4.54 | 2.52 |
| 6 | 6.64 | 7.59 | 4.86 | 2.73 |
| 7 | 5.95 | 7.78 | 4.35 | 3.43 |
| 8 | 7.49 | 11.11 | 4.61 | 6.5 |
| (**b**) | | | | |
| $N_{TDL}$ | **Measured Precision $\sigma_{SWEEP}$ [ps]** | | | |
| | **Mean** | **Max** | **Min** | **Amplitude** |
| 1 | 6.58 | 7.09 | 5.85 | 1.24 |
| 2 | 6.25 | 6.8 | 5.48 | 1.32 |
| 4 | 6.03 | 7.24 | 4.6 | 2.64 |
| 8 | 7.95 | 9.57 | 6.04 | 3.53 |

When sub-interpolation was adopted, as shown by Figures 20–22, from a precision perspective, the Log2 and sum1s became effectively comparable, i.e., this means that both decoding policies have the same amount of quantization error. Moreover, we observed an increase in the amplitude of the fluctuation of the precision with the sub-interpolation order. This was justified by an increase in the jitter ($\sigma_j$) due to the sub-interpolation process. Mostly, adding more TDLs caused the decoder and all subsequent blocks to grow in size as well. Congestion and increased electronic noise were thus brought about, mostly as a result of excessive switching activity close to the TDL, which increased the jitter. When we plotted the precision against the sub-interpolation order, as shown in Figure 23, independently of the decoder solutions, we observed a fluctuation of the precision induced by the increasing jitter. In conclusion, we can observe that the system achieved a fair trade-off between maximal precision and stability between two and four TDLs.

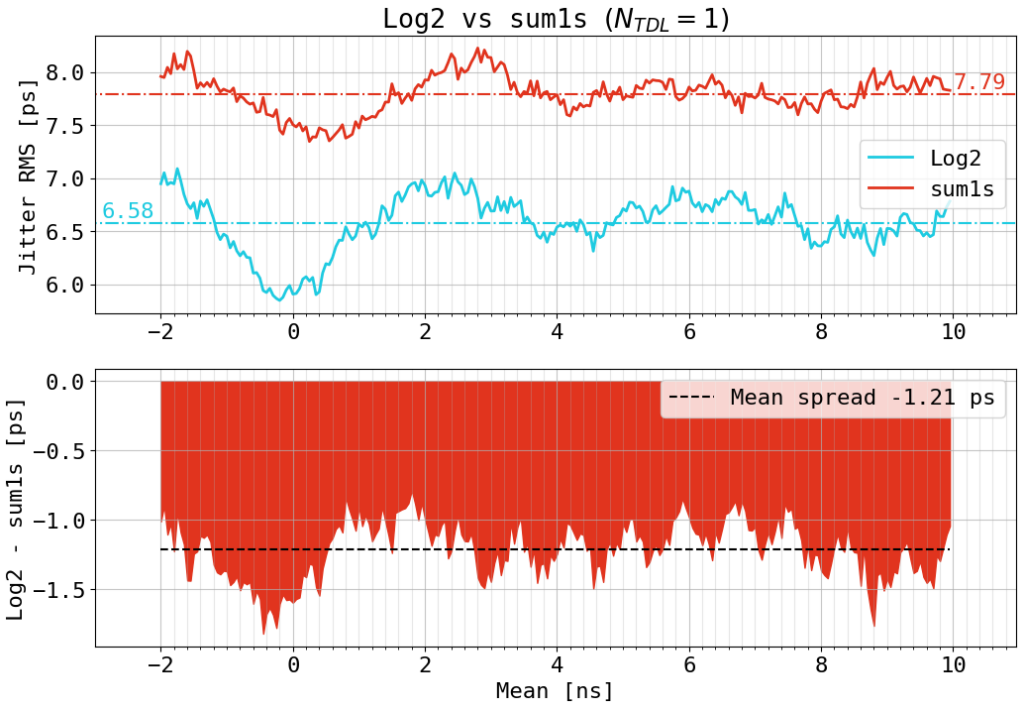



**Figure 19.** Single TDL precision (**top**) and spread (**bottom**) of the sum1 (red) and the Log2 (blue) implementations, with respect to the measured delay.

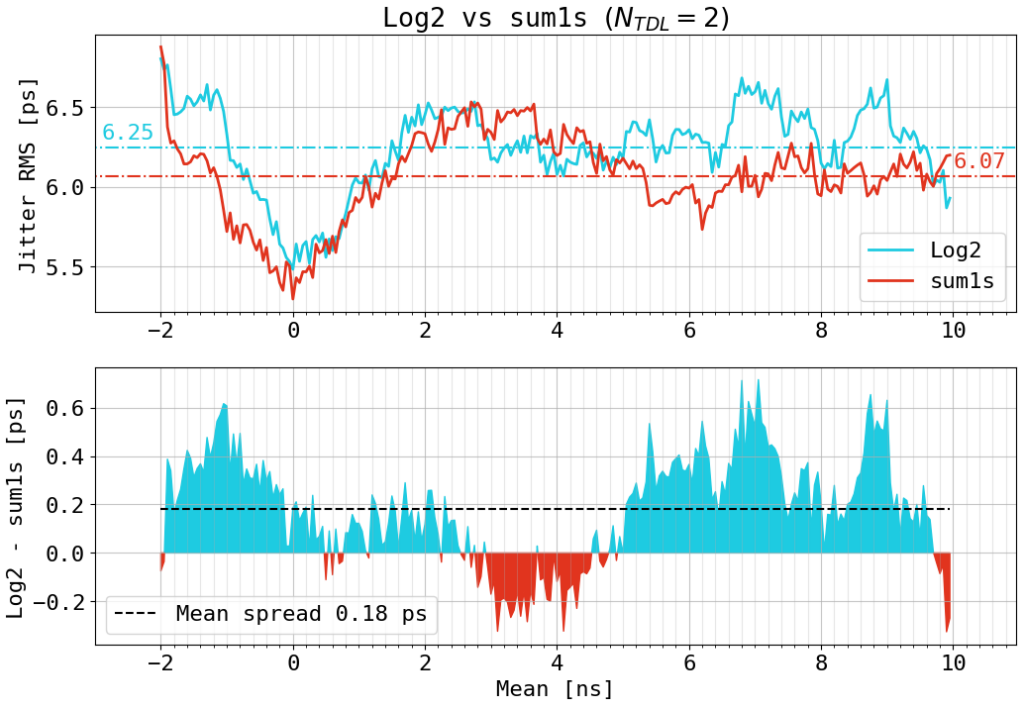



**Figure 20.** $N_{TDL} = 2$ sub-interpolated TDL precision (**top**) and spread (**bottom**) of the sum1 (red) and the Log2 (blue) implementations, with respect to the measured delay.

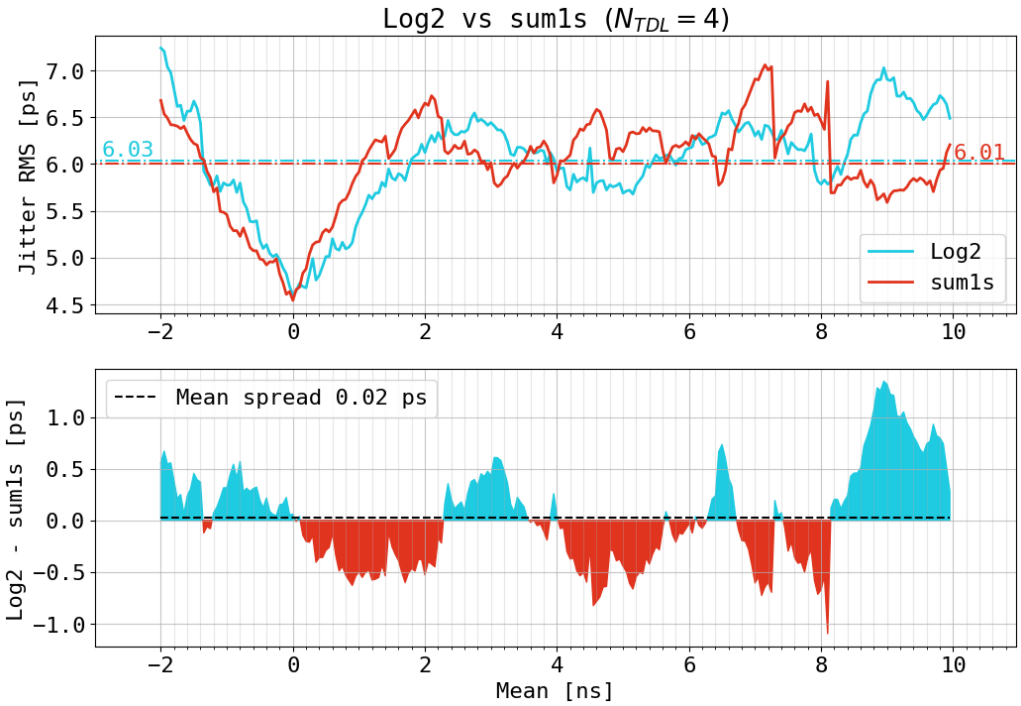



**Figure 21.** $N_{TDL} = 4$ sub-interpolated TDL precision (**top**) and spread (**bottom**) of the sum1 (red) and the Log2 (blue) implementations, with respect to the measured delay.

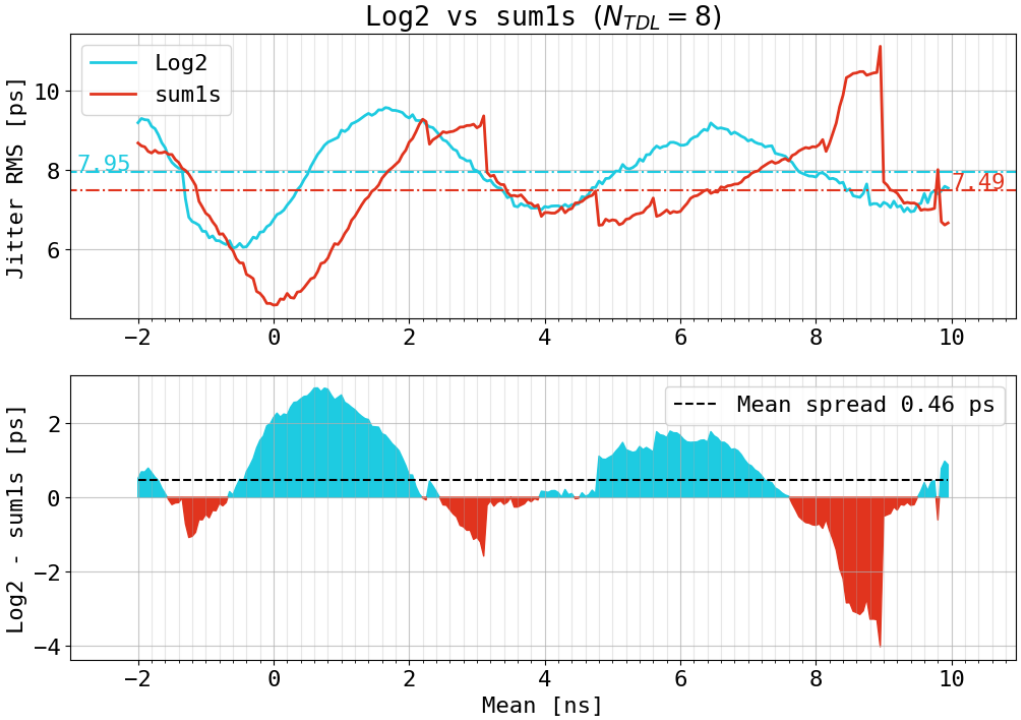



**Figure 22.** $N_{TDL} = 8$ sub-interpolated TDL precision (**top**) and spread (**bottom**) of the sum1 (red) and the Log2 (blue) implementations, with respect to the measured delay.

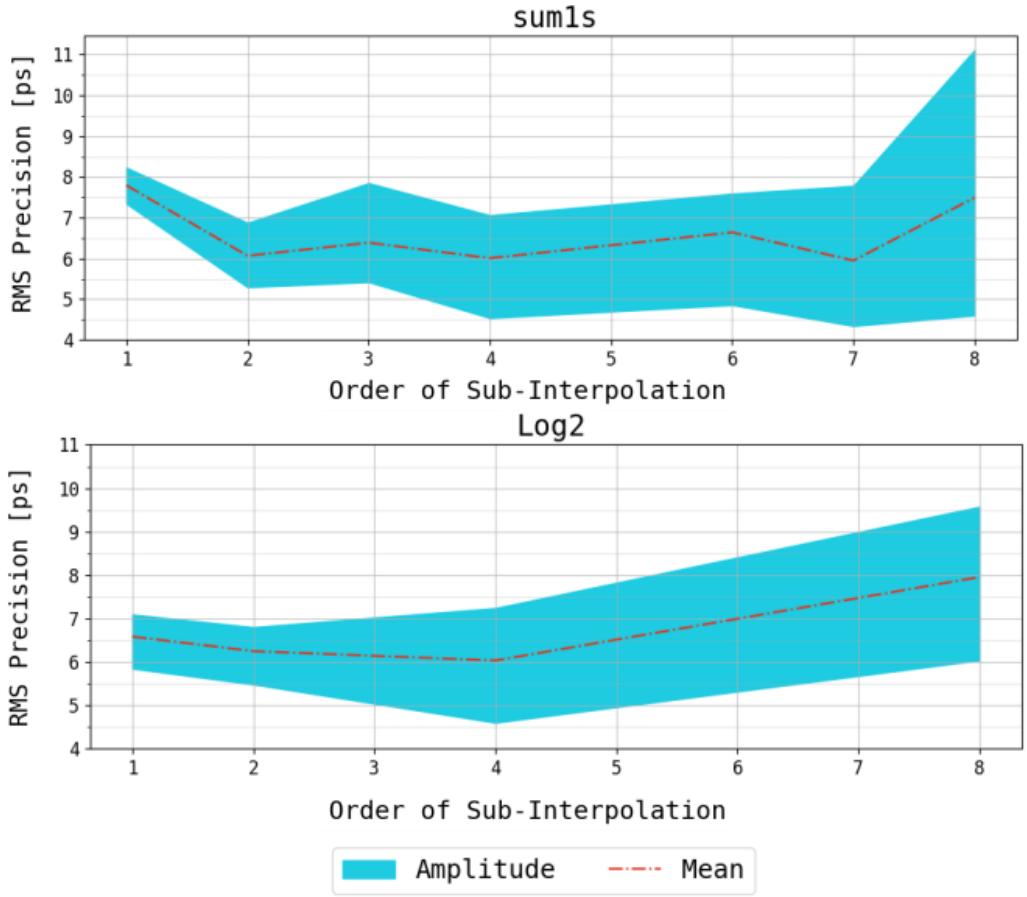



**Figure 23.** Precision of the sum1s (**top**) and Log2 (**bottom**) implementations, with respect to the order of sub-interpolation, reporting both the mean (red) and the amplitude (blue), defined as the difference between the maximum and minimum precision measured.

### *3.4. Single-Shot Precision Characterization*

Section 3.3 took into account a series of noise contributions not directly related to the TDC itself, as shown in Equation (9), in particular the jitter between the channels of the AWG4014 (i.e., $\sigma_{AWG}$). To correctly characterize the precision of the proposed TDCs, the jitter of the AWG4014 had to be estimated and removed; to achieve this, the precision was measured in three different scenarios:

- As $\overline{\sigma_{REAL}}$, Section 3.2 and Equation (8); using the internal signal, to consider only the internal jitter (i.e., $\sigma_j$) and the quantization error (i.e., $\varepsilon_Q$).
- As $\sigma_{SWEEP}$, Section 3.3 and Equation (9); using 2 distinct channels of the AWG4014, to consider both the jitter of the AWG4014 (i.e., $\sigma_{AWG}$) and the jitter of the HMC674LP3E (i.e., $\sigma_{HMC}$).
- As $\sigma_{split}$, this paragraph and Equation (10); using only 1 channel of the AWG4014 split into two (i.e., START and STOP), to consider only the jitter of of the HMC674LP3E (i.e., $\sigma_{HMC}$):

$$\sigma_{split}^2 = 2\sigma_{HMC}^2 + \sigma_j^2 + \varepsilon_Q^2 \tag{10}$$

In these terms, and considering Equations (8)–(10) we could ascertain the following:

1. The difference between $\overline{\sigma_{REAL}}$ and the quantization error (i.e., $\varepsilon_Q$) was the internal jitter (i.e., $\sigma_j$), i.e., $\overline{\sigma_{REAL}^2} - \varepsilon_Q^2 = \sigma_j^2$;
2. The difference between $\sigma_{split}$ and $\overline{\sigma_{REAL}}$ was the jitter of of the two HMC674LP3E (i.e., $\sqrt{2}\sigma_{HMC}$), i.e., $\sigma_{split}^2 - \overline{\sigma_{REAL}^2} = 2\sigma_{HMC}^2$;

3. The difference between $\sigma_{SWEEP}$ and $\sigma_{split}$ was the jitter between the 2 channels of the AWG4014 (i.e., $\sigma_{AWG}$) that interacted with the HMC674LP3E comparators, i.e., $\sigma^2_{SWEEP} - \sigma^2_{split} = \sigma^2_{AWG}$.

These values are reported in Tables 6 and 7.

**Table 6.** Summary of the measured precision for different signal modes: (**a**) sum1s; (**b**) Log2.

| (**a**) | | | | |
|---|---|---|---|---|
| $N_{TDL}$ | $\sigma_{SWEEP}$ [ps r.m.s] | $\sigma_{split}$ [ps r.m.s] | $\overline{\sigma_{REAL}}$ [ps r.m.s] | $\varepsilon_Q$ [ps r.m.s] |
| 1 | 7.79 | 6.68 | 3.95 | 3.66 |
| 2 | 6.07 | 4.7 | 2.77 | 2.12 |
| 4 | 6.01 | 3.73 | 1.79 | 1 |
| 8 | 7.49 | 4.01 | 1.40 | 0.42 |
| (**b**) | | | | |
| $N_{TDL}$ | $\sigma_{SWEEP}$ [ps r.m.s] | $\sigma_{split}$ [ps r.m.s] | $\overline{\sigma_{REAL}}$ [ps r.m.s] | $\varepsilon_Q$ [ps r.m.s] |
| 1 | 6.58 | 5.31 | 3.94 | 3.62 |
| 2 | 6.25 | 4.93 | 2.74 | 2.14 |
| 4 | 6.03 | 3.77 | 1.98 | 1.06 |
| 8 | 7.95 | 5.44 | 1.76 | 0.52 |

**Table 7.** Summary of the measured jitters for different signal modes: (**a**) sum1s; (**b**) Log2.

| (**a**) | | | |
|---|---|---|---|
| $N_{TDL}$ | $\sigma_{AWG}$ [ps r.m.s] | $\sqrt{2}\sigma_{HMC}$ [ps r.m.s] | $\sigma_j$ [ps r.m.s] |
| 1 | 4.01 | 5.39 | 1.49 |
| 2 | 3.84 | 3.80 | 1.78 |
| 4 | 4.71 | 3.27 | 1.48 |
| 8 | 6.33 | 3.76 | 1.34 |
| (**b**) | | | |
| $N_{TDL}$ | $\sigma_{AWG}$ [ps r.m.s] | $\sqrt{2}\sigma_{HMC}$ [ps r.m.s] | $\sigma_j$ [ps r.m.s] |
| 1 | 3.89 | 3.56 | 1.56 |
| 2 | 3.84 | 2.96 | 1.71 |
| 4 | 4.71 | 3.21 | 1.67 |
| 8 | 5.80 | 5.15 | 1.68 |

From Table 7, it can be observed that the $\sigma_{HMC}$ contribution was greater than the nominal value of 0.2 ps r.m.s. This is attributable to the threshold jitter (i.e., $Sl/\sigma_{Th}$) caused by the signal's slew rate (i.e., $Sl$), in our case 2 V/0.833 ps (i.e., 2.4 V/ns), and the electronic noise present at the threshold (i.e., $\sigma_{Th}$). Since the entire TDC (i.e., TDL, Nutt interpolation, decoder, calibrator, measurement histograms) was integrated into a single firmware, the level of threshold noise varied slightly from firmware to firmware. Thus, a noise level of a few millivolts r.m.s., compatible with what was measured experimentally, was calculated. All of this is reported in Table 8.

As for $\sigma_{AWG}$, we took into consideration that the jitter contribution AWG-4014 should have been independent of the TDC firmware, and that this contribution also involved jitters due to the interaction of the two HMC674LP3E. However, in practice, we observed small fluctuations of a few picoseconds that varied from firmware to firmware; so, we hypothesized that the fluctuations were principally due to the interaction between the AWG-4014 and the AWG-4014 (more suggestions will be listed in Section 4).

As expected, the additional jitter contribution of the AWG4014 was quite relevant, and the real precision of the TDC was correctly represented by $\sigma_{split}$. In this scenario, considering that the START and STOP signals and channels were equal, we defined the Single-Shot Precision (SSP) as

$$\sigma_{SSP}^2 = \frac{\sigma_{split}^2}{2} = \frac{\sigma_j^2 + \varepsilon_Q^2}{2} + \sigma_{HMC}^2, \tag{11}$$

which is reported in Table 9.

**Table 8.** Summary of the measured ($\sigma_{Th}$) and computed (i.e., $Sl/\sigma_{HMC}$) threshold jitters for different TDC implementations: (**a**) sum1s; (**b**) Log2.

| (**a**) | | | |
|---|---|---|---|
| $N_{TDL}$ | $\sigma_{HMC}$ **[ps r.m.s]** | $Sl/\sigma_{HMC}$ **[mV r.m.s]** | $\sigma_{Th}$ **[mV r.m.s]** |
| 1 | 3.81 | 6.35 | 6.23 |
| 2 | 2.68 | 4.47 | 4.38 |
| 4 | 2.31 | 3.86 | 3.91 |
| 8 | 2.66 | 4.43 | 4.22 |
| (**b**) | | | |
| $N_{TDL}$ | $\sigma_{HMC}$ **[ps r.m.s]** | $Sl/\sigma_{HMC}$ **[mV r.m.s]** | $\sigma_{Th}$ **[mV r.m.s]** |
| 1 | 2.52 | 4.20 | 4.19 |
| 2 | 2.90 | 4.83 | 4.25 |
| 4 | 2.27 | 3.78 | 3.82 |
| 8 | 3.64 | 6.07 | 5.63 |

**Table 9.** Summary of the SSP for different signal modes: (**a**) sum1s; (**b**) Log2.

| (**a**) | | |
|---|---|---|
| $N_{TDL}$ | $\sigma_{split}$ **[ps r.m.s]** | $\sigma_{SSP}$ **[ps r.m.s]** |
| 1 | 6.68 | 4.72 |
| 2 | 4.7 | 3.32 |
| 4 | 3.73 | 2.64 |
| 8 | 4.01 | 2.84 |
| (**b**) | | |
| $N_{TDL}$ | $\sigma_{split}$ **[ps r.m.s]** | $\sigma_{SSP}$ **[ps r.m.s]** |
| 1 | 5.31 | 3.75 |
| 2 | 4.93 | 3.49 |
| 4 | 3.77 | 2.67 |
| 8 | 5.44 | 3.85 |

### *3.5. Linearity*

Two linearity tests were performed; the first test focused on the linearity of the timestamps generated by the VTDL (Section 3.5.1), like the tests conducted in [28,53], while the second test concentrated on the linearity of the time intervals, considering the Nutt interpolation (Section 3.5.2), like the tests conducted in [64,65]. Both tests were conducted using CDT [50].

#### 3.5.1. Virtual Tapped Delay Line Linearity

The DNL and INL were derived from the CT and CC of each TDC. Specifically, a histogram (i.e., CDT) of the hits per tap for each TDC implementation (i.e., $h[i]$ with $i$ was the tap) was created. Considering that the FSR, thanks to the Nutt interpolation, corresponded to the clock period (i.e., $T_{CLK}$), the CT was derived, and, thus, the absolute (i.e., value in ps) tap-by-tap DNL (i.e., $dnl[i]$) was calculated, as reported in Equations (12) and (13):

$$CT[i] = \frac{h[i]}{\sum_i h[i]} \cdot T_{CLK} \tag{12}$$

$$dnl[i] = CT[i] - LSB \tag{13}$$

The absolute (i.e., value in ps) tap-by-tap INL (i.e., $inl[i]$) was derived like the tap-by-tap difference between the CC (i.e., $CC[i] = \sum_{k=0}^{k=1} CT[k]$) and its linear fitting (i.e., $r[i]$), as shown in Equation (14):

$$inl[i] = CC[i] - r[i] \tag{14}$$

The DNL and INL figures reported in Table 10, shown here in absolute terms (i.e., in picoseconds), were, therefore, extracted as the maximum amplitude value of the $dnl[i]$ and $inl[i]$ curves, i.e., $DNL = \max\{|dnl[i]|\}$, and $INL = \max\{|inl[i]|\}$.

The LSB was derived as the average value of the CT, while the ultra-bin was determined as the maximum value. Figure 24 shows the CTs, as well as the tap-by-tap DNLs and INLs for the different TDC implementations. Table 10 summarizes the ultra-bins, LSBs, and the absolute peak-to-peak values of the DNLs and INLs.

Moreover, observing Table 10, we can see that DNL, LSB, and ultra-bin, whether using sum1 or Log2, benefited from sub-interpolation, unlike INL, which showed only slight improvements for $N_{TDL} = 1$, $N_{TDL} = 2$, $N_{TDL} = 4$, and a worsening at $N_{TDL} = 8$. We can also note that for low orders of sub-interpolation (i.e., none, $N_{TDL} = 1$, and $N_{TDL} = 2$) the Log2 implementation was better, compared to sum1. For Ntdl = 1, both in the sum1s case and in the Log2 case, the presence of an ultra-bin at position 241 of 29.672 ps and 27.221 ps, respectively, was observed, as hypothesized in Section 2.3.

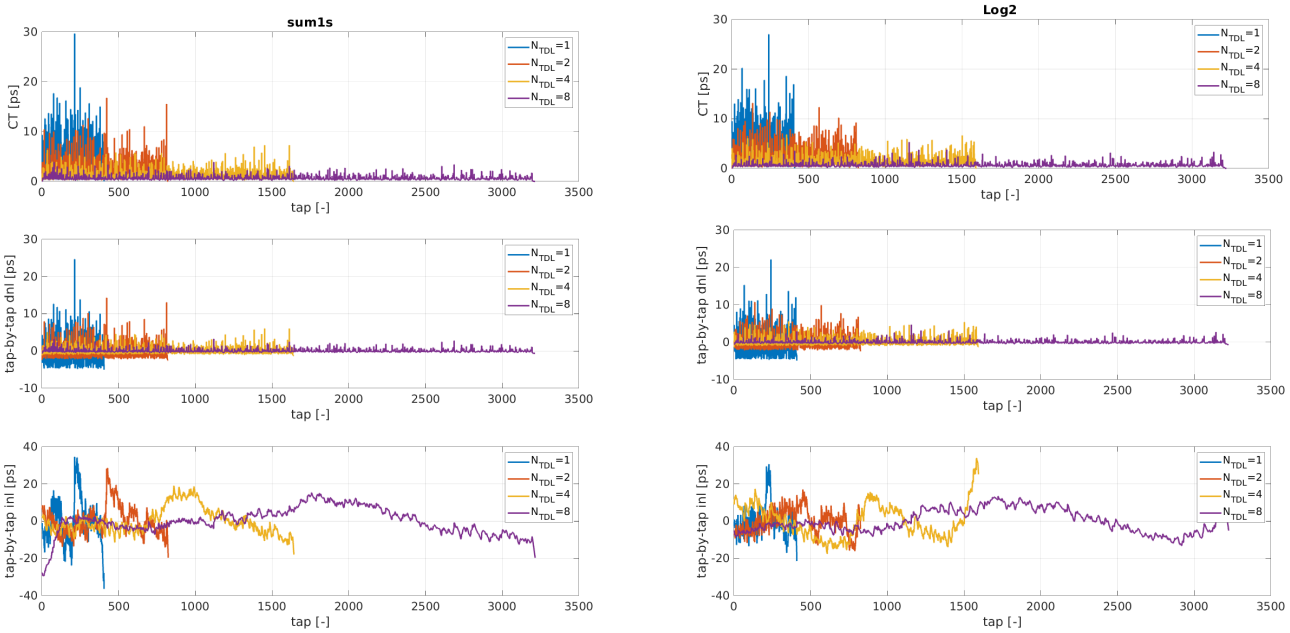



**Figure 24.** CT (**top**), tap-by-tap DNL (**center**), and tap-by-tap INL (**bottom**) of the VTDL for sum1s (**left**) and Log2 (**right**) TDC implementations at different sub-interpolated orders.

**Table 10.** DNL, INL, LSB, and ultra-bin for sum1s and Log2 TDC implementations at different sub-interpolated orders: (**a**) sum1s; (**b**) Log2.

| (**a**) | | | | | | |
|---|---|---|---|---|---|---|
| $N_{TDL}$ | **DNL [ps]** | | **INL [ps]** | | **LSB [ps]** | **Ultra-Bin [ps]** |
| 1 | [−5.010; 24.657] | 24.657 | [−36.347; 34.557] | 36.347 | 4.938 | 29.672 |
| 2 | [−2.465; 14.306] | 14.306 | [−19.631; 28.481] | 28.481 | 2.430 | 16.771 |
| 4 | [−1.231; 6.033] | 6.033 | [−17.926; 19.060] | 19.060 | 1.218 | 7.264 |
| 8 | [−0.620; 3.272] | 3.272 | [−29.463; 15.351] | 29.463 | 0.622 | 3.892 |
| (**b**) | | | | | | |
| $N_{TDL}$ | **DNL [ps]** | | **INL [ps]** | | **LSB [ps]** | **Ultra-Bin [ps]** |
| 1 | [−4.911; 22.343] | 22.343 | [−21.434; 30.606] | 30.606 | 4.878 | 27.221 |
| 2 | [−2.420; 10.748] | 10.748 | [−16.075; 17.018] | 17.018 | 2.424 | 13.169 |
| 4 | [−1.246; 5.580] | 5.580 | [−17.557; 33.934] | 33.934 | 1.255 | 6.826 |
| 8 | [−0.620; 4.6883] | 4.689 | [−13.153; 13.406] | 13.406 | 0.628 | 6.645 |

### 3.5.2. Nutt-Interpolated Linearity

For this section, the complete linearity test was considered, also taking into account the Nutt interpolation performed in a range between 0 and 100 ns (i.e., $\Delta T = 100$ ns). The CDT was performed by providing 2 channels of the various TDC implementations with uncorrelated events, thus collecting the CDT with a binning of 15.6 ps (i.e., $BIN = 15.6$ ps); the histogram was entirely acquired in the FPGA [63].

In this context, given the histogram $H[b]$ where $b$ denoted the bin, and in a manner completely analogous to Equations (12)–(14), considering that the measurement range was no longer $T_{CLK}$ but rather $\Delta T$, the absolute (value in ps) bin-by-bin DNL and INL were calculated as reported in Equations (15) and (16):

$$dnl[b] = \frac{H[b]}{\sum_b H[b]} \cdot \Delta T - BIN \tag{15}$$

$$inl[b] = \sum_{k=0}^{k=b} dnl[k] \tag{16}$$

The DNL and INL shown in Table 11, shown here in absolute terms (i.e., in picoseconds), were, therefore, extracted as the maximum amplitude value of the $dnl[b]$ and $inl[b]$ curves, i.e., $DNL = \max\{|dnl[b]|\}$ and $INL = \max\{|inl[b]|\}$.

Figure 25, for the sum1s decoder, and Figure 26, for the Log2 implementations, show the bin-by-bin DNL and bin-by-bin INL, considering only the sub-interpolation orders (i.e., $N_{TDL}$) of 1 (i.e., none), 2, and 4, which were the most interesting ones, as discussed in the previous paragraph (i.e., lower jitter). The INL and DNL values referred to the 15.6 ps bin are also provided in Table 11. As we can see, the order of sub-interpolation did not yield a significant effect on the DNL while the INL increased with it.

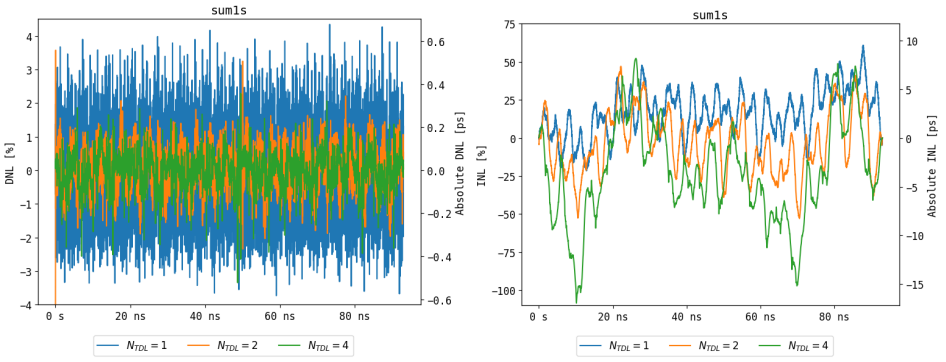



**Figure 25.** DNL (**left**) and INL (**right**) in the range 0–100 ns binned at 15.6 ps for the sum1s TDC implementations at different sub-interpolation orders.

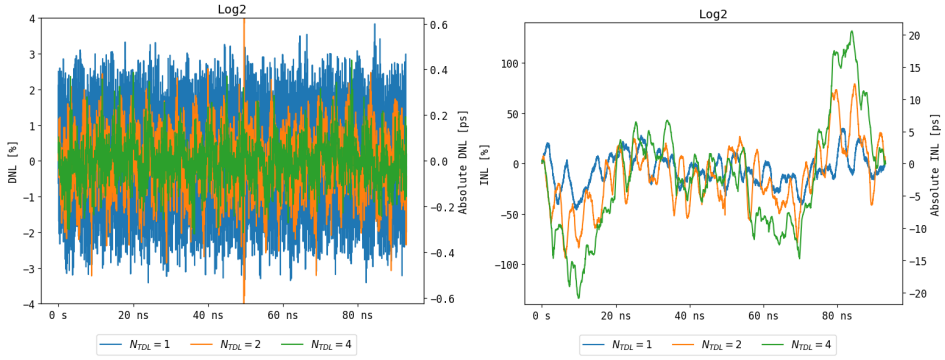



**Figure 26.** DNL (**left**) and INL (**right**) in the range 0–100 ns binned at 15.6 ps for the Log2 TDC implementations at different sub-interpolation orders.

**Table 11.** Linearity test (i.e., DNL and INL) in the range 0–100 ns result for the TDC implementations: (**a**) sum1s; (**b**) Log2.

| (**a**) | | | | |
|---|---|---|---|---|
| $N_{TDL}$ | **DNL [ps]** | | **INL [ps]** | |
| 1 | [−0.583; 0.678] | 0.678 | [−3.373; 9.507] | 9.507 |
| 2 | [−0.633; 0.558] | 0.663 | [−8.255; 7.333] | 8.255 |
| 4 | [−0.523; 0.357] | 0.523 | [−16.939; 8.157] | 16.939 |
| (**b**) | | | | |
| $N_{TDL}$ | **DNL [ps]** | | **INL [ps]** | |
| 1 | [−0.534; 0.599] | 0.599 | [−7.151; 5.494] | 7.151 |
| 2 | [−0.601; 0.605] | 0.605 | [−14.69; 12.381] | 14.69 |
| 4 | [−0.32; 0.441] | 0.441 | [−20.925; 20.611] | 20.925 |

### *3.6. Final Results*

From Tables 3–6, it is evident that the sub-interpolated TDL TDCs using the sum1s decoder achieved better precision compared to those using Log2; hence, we deduce that the sum1s decoder is preferred for sub-interpolated TDL TDCs, while Log2 is more suitable for non-sub-interpolated ones. As a conclusion, the optimal order of sub-interpolation was identified as 4 for the optimization of the precision of the TDC, in conjunction with the sum1s decoder reaching an SSP of 2.64 ps rms, a quantization error of 1 ps r.m.s., an LSB of 1.218 ps, a peak DNL over the V-TDL of 6.033 ps, and a peak DNL on the Nutt-interpolated TDC of 0.523 ps. In reference to precision vs. sub-interpolation order, observing Figure 23, a "bathtub curve"—similar to that obtained in [36] (in detail, $N_{TDL}$ TDLs were implemented in parallel, each executing a Wave Union A (WUA) algorithm for a total of $2 \times N_{\mathrm{TDL}}$ orders of sub-interpolation) by the same research group—was observed on a 28 nm Xilinx 7-Series Artix-7 200T. Obviously, since both the technology node and the sub-interpolation algorithm were different, the optimum in [36] was $N_{\mathrm{TDL}} = 6$, different from the 4 obtained in this paper.

However, such a "bathtub curve" was not observed in [33] (28 nm Xilinx 7-Series Kintex-7 160T) and [46] (40 nm Xilinx 6-Series Virtex-6 200T), where a trend of continuous improvement was observed with increasing $N_{\mathrm{TDL}}$. This was hypothesized to be due to the absence of processing blocks in [33,46] (i.e., decoder and calibrator), as the thermometer codes of the TDLs were acquired directly and subsequently processed offline on a PC. In fact, in this paper (and also in [36], although not specified in the text), all processing (i.e., decoding, calibration, and measurement) was performed on the FPGA, which significantly worsened the signal integrity of the FPGA (i.e., higher $\sigma_j$) as $N_{\mathrm{TDL}}$ increased.

Another interesting implementation was the single TDL with the Log2 decoder providing great linearity and acceptable precision, reaching an SSP of 3.75 ps r.m.s., a quantization

error of 3.62 ps r.m.s., an LSB of 4.878 ps, a peak DNL over the TDL of 15.308 ps, and a peak DNL on the Nutt-interpolated TDC of 0.599 ps. On the other hand, a single TDL with the sum1s provided an SSP of 4.72 ps r.m.s., a quantization error of 3.66 ps r.m.s., an LSB of 4.938 ps, a peak DNL over the TDL of 24.657 ps, and a peak DNL on the Nutt-interpolated TDC of 0.678 ps. Comparing the quantization errors, LSBs, and DNLs of these two implementations, we deduced that ignoring BEs, even if they were present in the thermometer code, was more effective in the 20 nm Xilinx UltraScale FPGAs if a single TDL (without sub-interpolation) was implemented. Therefore, even though the TDL was affected by BEs, in our opinion the main informational content resided in the Most Signification Bit (MSB) of the thermometric code, and the bubbles did not help to increase the precision. In contrast, increasing the order of sub-interpolation, the bubbles helped to improve the quantization errors, LSBs, and DNLs, probably because they somehow helped to decorrelate the measurements coming from the different TDLs, making the sub-interpolation more effective.

The single TDL with the Log2 can be also exploited for implementations where a high number of channels is needed (64 in the proposed XCKU040-2FFVA1156E FPGA), since it utilizes less than 1% of each type of the resources in the FPGA per channel, while the four sub-interpolated TDLs with sum1s can be used for up to 24 channels in the proposed XCKU040-2FFVA1156E FPGA. Table 12 summarizes the area occupation, in terms of the raw number of blocks, considering the TDL, the decoder, and the calibrator. For comparison, the best-performing implementation occupied as much as 4x the area, which greatly limited the maximum number of channels. Table 13 shows the percentage area occupancy of the entire firmware and of a single TDC channel, demonstrating that the TDC is a minimal part of the overall firmware in considering a real measurement setup (e.g., with processing end transmission). To provide a pictorial overview, Figure 27 shows the floorplan of the FPGA for the sum1s implementation with 2 channels and four TDLs, highlighting the TDLs (violet for channel 1 and pink for channel 2), the decoder (yellow for channel 1 and ocher for channel 2), the decoder (green for channel 1 and blue for channel 2), the histogram engine (orange), and the auxiliary logic (blue).

**Table 12.** Summary of area occupation.

| $N_{TDL}$ | Decoder | LUTs | DFFs | CARRY8 | CLB | Block RAM |
|---|---|---|---|---|---|---|
| 1 | Log2 | 955 | 1529 | 103 | 259 | 72 kb (2 blocks) |
| 4 | sum1s | 3607 | 6564 | 337 | 883 | 126 kb (3.5 blocks) |

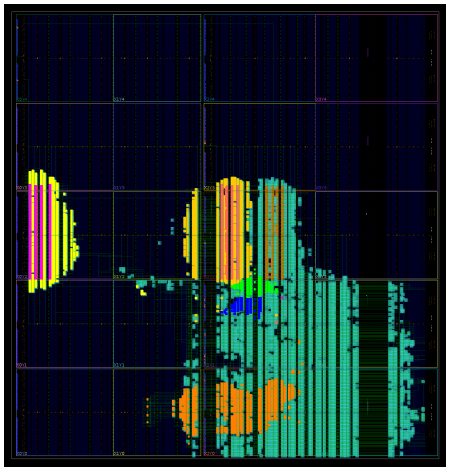


**Figure 27.** Floorplan for the sum1s implementation with 2 channels and four TDLs, highlighting the TDLs (violet for channel 1 and pink for channel 2), the decoder (yellow for channel 1 and ocher for channel 2), the decoder (green for channel 1 and blue for channel 2), the histogram engine (orange), and the auxiliary logic (blue).

**Table 13.** Percentage area occupancy of the entire firmware (**a**) and a single channel (**b**) of the TDC in the various implementations.

| (**a**) | | | | | |
|---|---|---|---|---|---|
| $N_{TDL}$ | **Type** | **LUTs** | **DFFs** | **BRAM** | **DSP** |
| 1 | sum1s | 8.88 | 4.05 | 17.2 | 0.75 |
| | Log2 | 8.63 | 3.79 | 17.2 | 0.75 |
| 2 | sum1s | 9.89 | 4.69 | 17.2 | 0.75 |
| | Log2 | 9.38 | 4.16 | 17.2 | 0.75 |
| 4 | sum1s | 11.9 | 10.35 | 17.2 | 0.75 |
| | Log2 | 10.88 | 9.29 | 17.2 | 0.75 |
| 8 | sum1s | 15.93 | 10.85 | 17.2 | 0.75 |
| | Log2 | 13.93 | 10.74 | 17.2 | 0.75 |
| (**b**) | | | | | |
| $N_{TDL}$ | **Type** | **LUTs** | **DFFs** | **BRAM** | |
| 1 | sum1s | 0.48 | 0.72 | 0.33 | |
| | Log2 | 0.39 | 0.32 | 0.33 | |
| 2 | sum1s | 0.82 | 0.93 | 0.50 | |
| | Log2 | 0.65 | 0.32 | 0.50 | |
| 4 | sum1s | 1.49 | 1.35 | 0.58 | |
| | Log2 | 1.15 | 0.56 | 0.58 | |
| 8 | sum1s | 2.83 | 1.35 | 0.67 | |
| | Log2 | 2.16 | 1.48 | 0.67 | |

## 4. State of the Art and Future Development

### *4.1. State of the Art*

It is important to review the current state of the art on the subject matter, to understand where our contribution stands. In particular, in the TDL TDC space, research has moved towards the implementation of two main techniques to improve measurement precision. The first technique is Dual Sampling (DS) [7,14,38], which samples both the C and O outputs of the TDL, effectively halving the propagation delay of the taps, at the cost of linearity. The second technique is Sub-TDL [12,14], where taps of the same TDL are grouped into smaller chains and then sub-interpolated, to reduce the impact of BEs. While this last approach is valid for reducing BEs by elongating the tap delay, our focus was on finding a general decoding policy to address these errors globally. By implementing placement optimizations, we achieved similar results. An interesting future development, since nothing similar has been found in the literature, would be to verify the tuned delay line technique [27] and compare it accordingly. A comparison between this work and other relevant UltraScale implementations is provided in Table 14. Unlike all the works reported in Table 14, in this paper not only the TDC but also the entire histogramming and measurement mechanism were implemented directly in the FPGA, thereby contributing to the increases in disturbance and noise that were converted in an increase of jitter (from Figure 1 of [38], Figure 6 and Table I of [7], Figures 1 and 2 of [66], Figure 12 of [12], Figure 5 of [13], and Figure 14 of [14], it is possible to see that only the TDC was implemented in the FPGA).

**Table 14.** Summary of the state of the art.

| Ref. | Mode | Precision [ps r.m.s.] |
|---|---|---|
| [38]-2016 | DS | 3.9 |
| [7]-2016 | DS, Bin Decimation | 4.2 |
| [66]-2017 | Bin Decimation | 4.7–5.6 |
| [12]-2019 | Sub-TDL | 5.37 |
| [13]-2021 | LSPM | 5.55 |
| [14]-2022 | WUA, Sub-TDL, DS | 3.63 |
| This work | One-TDL Log2 | 5.31 |
| | Four-TDL sum1s | 3.73 |

### *4.2. Potential Applications Enabled*

As mentioned in the Introduction, time measurements are crucial in quantum technologies, life sciences, and nuclear physics. State-of-the-art experiments demand very high precision (i.e., a few picoseconds) and a significant number of channels in a single chip (i.e., tens), which has led, until now, to the use of Time-to-Amplitude Converters (TACs), analog time converters on ASICs inherently more precise (i.e., sub-picosecond) than a TDC [67], and ASIC-based TDCs, such as CERN's PicoTDC [68,69], PETsys's TOFPET2 [70,71], and Weeroc's Temporoc [72].

However, the use of ASICs for time measurement imposes significant limitations, in terms of flexibility and time-to-market, characteristics that are increasingly required in both academic and industrial research. As a result, various sectors where FPGA-based TDC technology is already mature, such as Cross Delay Line (CDL) detection systems [73], are transitioning toward FPGA-based TDCs, moving away from ASIC-based solutions like ACAM's TDCs [74].

In emerging research fields, such as high-energy physics [71], TOF-PET [75] and particle therapy [76,77], and quantum technologies [78,79], FPGA-based TDCs are progressively reaching ASIC performance levels, in terms of both precision and channel count, thanks to continuous optimization efforts. Therefore, optimization processes and analyses, such as those discussed in this paper, are essential to opening the doors for FPGA-based TDCs in these advanced fields. In this context, the single TDL with Log2 decoder achieved an SSP of 3.75 ps r.m.s. with an area occupation per channel of only 259 CLBs and 72 kb of BRAM, enabling the implementation up to 64 channels on the proposed FPGA (i.e., XCKU040-2FFVA1156E). This system is, thus, comparable in terms of both precision and channel count to the PicoTDC, and superior in precision to the TOFPET2 and Temporoc ASICs. On the other hand, the four-TDL sub-interpolated TDC with sum1s decoder, although limited to a maximum of only 24 channels on the proposed FPGA (i.e., 883 CLBs and 126 kb per channel), achieved better precision (i.e., SSP of 2.64 ps r.m.s.) than the PicoTDC. A summary is reported in Table 15.

**Table 15.** Comparative table between proposed FPGA-based TDCs and state-of-the-art ASIC-based TDCs used in advanced applications, such as high-energy physics, medical diagnostics, and quantum technologies.

| Ref. | Name | LSB [1] | SSP [2] | Channels |
|---|---|---|---|---|
| [68,69] | PicoTDC | 3 ps | 3.7 ps r.m.s. | 64 |
| [70] | TOFPET2 | 30 ps | - | 64 in 36 × 25 mm chip |
| [72] | Temporoc | 50 ps | 20 ps r.m.s. | 64 in 20 × 20 mm chip |
| This work | One-TDL Log2 | <8.9 ps | 3.75 ps r.m.s. | <64 in 35 × 35 mm chip |
| | Four-TDL sum1s | <2.47 ps | 2.64 ps r.m.s. | <24 in 35 × 35 mm chip |

[1] $LSB_{EQ}$ was considered for "This work". [2] Also, the jitter due to the comparator was considered in "This work".

### *4.3. Open Issues and Pending Experiments*

In the following sub-paragraphs, all open issues and pending experiments identified in this manuscript are listed, which are, therefore, left for future developments.

#### 4.3.1. AWG4014 Jitter Fluctuation

In Section 3.4 and in Table 7, it is possible to observe fluctuations of $\sigma_{AWG}$ that represent the jitter between the channels of the AWG-4014 when connected to the two HMC674 comparators in order to perform Time Sweep tests (Section 3.3). Obviously, the state of the TDC should not have influenced in any way the jitter that existed only between the 2 channels of the AWG-4014, but the TDC firmware could modify the state of the KCU105 EVB (e.g., noise on the power lines, ground lines, and common mode fluctuations). Therefore, we hypothesize that the jitters between the AWG-4014, the two HMC674 comparators, and the KCU105 varied slightly from firmware to firmware. An estimate of the jitter existing between the 2 AWG-4014 channels using, for instance, an oscilloscope was not performed, as it would not only be difficult to distinguish the jitter component of the AWG-4014 alone from the measurement precision and jitter of the oscilloscope, but also because oscilloscopes have an analog front end that conditions the signals, thereby practically altering (e.g., amplifying and filtering) the input signals and possibly also the jitter.

Consequently, as found in the scientific literature, the figure of merit for precision used was the SSP, evaluated as indicated in Equation (11).

#### 4.3.2. Most Significant Information in the MSB of the Thermometric Code

Another interesting open issue would be to mathematically demonstrate, not just experimentally, that the Log2 decoding is more effective in terms of precision than the sum1s decoding in the case of non-sub-interpolated TDLs, whereas the bubble compression offered by the sum1s algorithm helps to improve sub-interpolation. This would validate or refute the hypothesis proposed in Section 3.6, where we deduced that the most significant information is contained in the MSB of the thermometer code and that the BEs only help improve sub-interpolation, likely because they somehow help to decorrelate measurements from different TDLs, thereby making sub-interpolation more effective.

#### 4.3.3. Absence of the "Bathtub Curve"

From a metrological point of view, it would also be interesting to conduct a mathematical analysis to demonstrate the absence of the "bathtub curve" shape in the relationship between precision and sub-interpolation, depending on the size of the firmware, which induces an increase in noise and disturbances resulting in greater jitter (Section 3.3, Figure 23).

#### 4.3.4. Clock Skew Pattern Behavior

Another interesting point would to investigate—and, thus, formulate a more accurate hypothesis—the phenomenon reported in Section 2.2, Figure 10. Specifically, to reassess the clock skew pattern in the presence and absence of the TDC firmware observed experimentally in post-implemenation on the XCKU040-2FFVA1156E FPGA and not observed in the scientific literature Xilinx 28 nm 7-Series FPGA (e.g., [33]).

## 5. Conclusions

In conclusion, within the context of 20 nm FPGA technology, we have established design rules to mitigate BEs, which are the most significant error sources in TDL TDC implementations in FPGAs. This is particularly relevant when spatial sub-interpolation is used to maximize resolution. We identified that the optimal number of parallel TDLs, or sub-interpolation order, is four. Additionally, the optimal decoding policy was determined both in the presence and absence of sub-interpolation. When sub-interpolation was employed, the sum1s decoder achieved an SSP of 2.64 ps r.m.s., with a DNL of 0.523 ps and an INL of 16.939 ps. This configuration occupied 883 CLBs and 126 kb of BRAM, and it could fit up to a 24-channel implementation in the proposed FPGA (i.e., XCKU040-2FFVA1156E). In contrast, for a single TDL, the Log2 decoder achieved a single-shot precision of 3.75 ps r.m.s., with a DNL of 0.599 ps and an INL of 7.151 ps, while occupying only 259 CLBs and 72 kb of BRAM, and it could fit up to a 64-channel implementation in the proposed FPGA (i.e., XCKU040-2FFVA1156E). These two distinct implementations provide different benefits:

the former offers very high precision, while the latter provides adequate precision in 1/4 of the area.

Moreover, this study helps to further narrow the gap between FPGA-based TDCs and ASIC solutions, which, being better-performing, are often used in cutting-edge applications (e.g., quantum technologies, life sciences, and nuclear physics). However, ASIC-based TDCs are less suitable for R&D and fast prototyping, as they are much less flexible, more expensive, and have long time-to-market, with respect to FPGA-based solutions.

**Author Contributions:** Methodology, N.L.; Software, G.F.; Validation, F.G.; Formal analysis, E.R.; Investigation, M.M.; Data curation, G.B.; Writing—original draft, A.G.; Visualization, A.C. All authors have read and agreed to the published version of the manuscript.

**Funding:** This research received no external funding

**Data Availability Statement:** Data are contained within the articles.

**Acknowledgments:** A special thanks to TEDIEL S.r.l., a spin-off of Politecnico di Milano.

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

## References

1. Yuan, F. *CMOS Time-Mode Circuits and Systems: Fundamentals and Applications*; CRC Press: Boca Raton, FL, USA, 2020.
2. Aguero, M.; Hnilo, A.; Kovalsky, M.; Nonaka, M. Testing a hypothetical transient deviation from quantum mechanics: Preliminary results. *J. Opt. Soc. Am. B* **2023**, *40*, C28–C34. [CrossRef]
3. Lusardi, N.; Garzetti, F.; Bulgarini, G.; Gourgues, R.; Los, J.; Geraci, A. Single photon counting through multi-channel TDC in programmable logic. In Proceedings of the 2016 IEEE Nuclear Science Symposium, Medical Imaging Conference and Room-Temperature Semiconductor Detector Workshop (NSS/MIC/RTSD), Strasbourg, France, 29 October–6 November 2016; pp. 1–4. [CrossRef]
4. Garzetti, F.; Salgaro, S.; Venialgo, E.; Lusardi, N.; Corna, N.; Geraci, A.; Charbon, E. Plug-and-play TOF-PET Module Readout Based on TDC-on-FPGA and Gigabit Optical Fiber Network. In Proceedings of the 2019 IEEE Nuclear Science Symposium and Medical Imaging Conference (NSS/MIC), Manchester, UK, 26 October–2 November 2019; pp. 1–4. [CrossRef]
5. Parsakordasiabi, M.; Vornicu, I.; Rodríguez-Vázquez, A.; Carmona-Galán, R. A Low-Resources TDC for Multi-Channel Direct ToF Readout Based on a 28-nm FPGA. *Sensors* **2021**, *21*, 308. [CrossRef] [PubMed]
6. Lusardi, N.; Garzetti, F.; Corna, N.; Reale, A.; Geraci, A.; Dobovicnik, E.; Cautero, G.; Dri, C.; Sergo, R.; Stebel, L. Advanced System in FPGA for 3D (X, Y, t) Imaging with Cross Delay-Lines. In Proceedings of the 2019 IEEE Nuclear Science Symposium and Medical Imaging Conference (NSS/MIC), Manchester, UK, 26 October–2 November 2019; pp. 1–4. [CrossRef]
7. Wang, Y.; Liu, C. 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.* **2016**, *63*, 2632–2638. [CrossRef]
8. Garzetti, F.; Lusardi, N.; Geraci, A.; Dobovicnik, E.; Cautero, G.; Dri, C.; Sergo, R.; Stebel, L. Fully FPGA-based and all-reconfigurable TDC for 3D (X, Y, t) Cross Delay-Line detectors. In Proceedings of the 2018 IEEE Nuclear Science Symposium and Medical Imaging Conference Proceedings (NSS/MIC), Sydney, NSW, Australia, 10–17 November 2018; pp. 1–3. [CrossRef]
9. Machado, R.; Cabral, J.; Alves, F.S. Recent Developments and Challenges in FPGA-Based Time-to-Digital Converters. *IEEE Trans. Instrum. Meas.* **2019**, *68*, 4205–4221. [CrossRef]
10. Szyduczyński, J.; Kościelnik, D.; Miśkowicz, M. Time-to-digital conversion techniques: A survey of recent developments. *Measurement* **2023**, *214*, 112762. [CrossRef]
11. Wang, Y.; Xie, W.; Chen, H.; Pei, C.; Li, D. A two-stage interpolation time-to-digital converter implemented in 20 nm and 28 nm FGPAs. *IEEE Trans. Ind. Electron.* **2024**, *71*, 15200–15210. [CrossRef]
12. Chen, H.; Li, D.D.U. Multichannel, Low Nonlinearity Time-to-Digital Converters Based on 20 and 28 nm FPGAs. *IEEE Trans. Ind. Electron.* **2019**, *66*, 3265–3274. [CrossRef]
13. Zhang, M.; Yang, K.; Chai, Z.; Wang, H.; Ding, Z.; Bao, W. High-Resolution Time-to-Digital Converters Implemented on 40-, 28-, and 20-nm FPGAs. *IEEE Trans. Instrum. Meas.* **2021**, *70*, 2002310. [CrossRef]
14. Xie, W.; Chen, H.; Li, D.D.U. Efficient Time-to-Digital Converters in 20 nm FPGAs With Wave Union Methods. *IEEE Trans. Ind. Electron.* **2022**, *69*, 1021–1031. [CrossRef]
15. Xie, W.; Wang, Y.; Chen, H.; Li, D.D.U. 128-Channel High-Linearity Resolution-Adjustable Time-to-Digital Converters for LiDAR Applications: Software Predictions and Hardware Implementations. *IEEE Trans. Ind. Electron.* **2022**, *69*, 4264–4274. [CrossRef]
16. Kim, J.; Jung, J.H.; Choi, Y.; Jung, J.; Lee, S. Linearity improvement of UltraScale+ FPGA-based time-to-digital converter. *Nucl. Eng. Technol.* **2023**, *55*, 484–492. [CrossRef]
17. Dadouche, F.; Turko, T.; Uhring, W.; Malass, I.; Dumas, N.; Normand, J.P.L. New Design-methodology of High-performance TDC on a Low Cost FPGA Targets. *Sens. Transducers* **2015**, *193*, 123.

18. Wang, Z.; Lu, J.; Nunez-Yanez, J. A Low-complexity FPGA TDC based on a DSP Delay Line and a Wave Union Launcher. In Proceedings of the 2022 25th Euromicro Conference on Digital System Design (DSD), Maspalomas, Spain, 31 August–2 September 2022; pp. 101–108. [CrossRef]
19. Kwiatkowski, P. Employing FPGA DSP blocks for time-to-digital conversion. *Metrol. Meas. Syst.* **2019**, *26*, 631–643. [CrossRef]
20. Chaberski, D.; Frankowski, R.; Zieliński, M.; Zaworski, Ł. Multiple-tapped-delay-line hardware-linearisation technique based on wire load regulation. *Measurement* **2016**, *92*, 103–113. [CrossRef]
21. Wang, Y.; Kuang, J.; Liu, C.; Cao, Q. A 3.9-ps RMS Precision Time-to-Digital Converter Using Ones-Counter Encoding Scheme in a Kintex-7 FPGA. *IEEE Trans. Nucl. Sci.* **2017**, *64*, 2713–2718. [CrossRef]
22. Carra, P.; Bertazzoni, M.; Bisogni, M.G.; Cela Ruiz, J.M.; Del Guerra, A.; Gascon, D.; Gomez, S.; Morrocchi, M.; Pazzi, G.; Sanchez, D.; et al. Auto-Calibrating TDC for an SoC-FPGA Data Acquisition System. *IEEE Trans. Radiat. Plasma Med. Sci.* **2019**, *3*, 549–556. [CrossRef]
23. Garzetti, F.; Corna, N.; Lusardi, N.; Geraci, A. Time-to-Digital Converter IP-Core for FPGA at State of the Art. *IEEE Access* **2021**, *9*, 85515–85528. [CrossRef]
24. 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]
25. Randall, R.; Tordon, M. Data Acquisition. In *Encyclopedia of Vibration*; Braun, S., Ed.; Elsevier: Oxford, UK, 2001; pp. 364–376. [CrossRef]
26. Szplet, R.; Jachna, Z.; Kwiatkowski, P.; Rozyc, K. A 2.9 ps equivalent resolution interpolating time counter based on multiple independent coding lines. *Meas. Sci. Technol.* **2013**, *24*, 035904. [CrossRef]
27. Won, J.Y.; Lee, J.S. Time-to-Digital Converter Using a Tuned-Delay Line Evaluated in 28-, 40-, and 45-nm FPGAs. *IEEE Trans. Instrum. Meas.* **2016**, *65*, 1678–1689. [CrossRef]
28. Won, J.Y.; Kwon, S.I.; Yoon, H.S.; Ko, G.B.; Son, J.W.; Lee, J.S. Dual-Phase Tapped-Delay-Line Time-to-Digital Converter With On-the-Fly Calibration Implemented in 40 nm FPGA. *IEEE Trans. Biomed. Circuits Syst.* **2016**, *10*, 231–242. [CrossRef] [PubMed]
29. Zhang, S.; Wang, S.; Lin, X.; Ren, G. A 6-bit low power flash ADC with a novel bubble error correction used in UWB communication systems. In Proceedings of the 2014 IEEE International Conference on Electron Devices and Solid-State Circuits, Chengdu, China, 18–20 June 2014; pp. 1–2. [CrossRef]
30. Ghoshal, P.; Sen, S.K. A bit swap logic (BSL) based bubble error correction (BEC) method for flash ADCs. In Proceedings of the 2016 2nd International Conference on Control, Instrumentation, Energy & Communication (CIEC), Kolkata, India, 28–30 January 2016; pp. 111–115. [CrossRef]
31. Jaworski, Z. Verilog HDL model based thermometer-to-binary encoder with bubble error correction. In Proceedings of the 2016 MIXDES—23rd International Conference Mixed Design of Integrated Circuits and Systems, Lodz, Poland, 23–25 June 2016; pp. 249–254. [CrossRef]
32. Kwiatkowski, P.; Sondej, D.; Szplet, R. Bubble-Proof Algorithm for Wave Union TDCs. *Electronics* **2022**, *11*, 30. [CrossRef]
33. Kwiatkowski, P.; Szplet, R. Efficient Implementation of Multiple Time Coding Lines-Based TDC in an FPGA Device. *IEEE Trans. Instrum. Meas.* **2020**, *69*, 7353–7364. [CrossRef]
34. Lusardi, N.; Garzetti, F.; Corna, N.; Marco, R.D.; Geraci, A. Very High-Performance 24-Channels Time-to-Digital Converter in Xilinx 20-nm Kintex UltraScale FPGA. In Proceedings of the 2019 IEEE Nuclear Science Symposium and Medical Imaging Conference (NSS/MIC), Manchester, UK, 26 October–2 November 2019; pp. 1–4. [CrossRef]
35. Chaberski, D.; Frankowski, R.; Gurski, M.; Zielinski, M. Comparison of Interpolators Used for Time-Interval Measurement Systems Based on Multiple-Tapped Delay Line. *Metrol. Meas. Syst.* **2017**, *24*, 401–412. [CrossRef]
36. Lusardi, N.; Garzetti, F.; Geraci, A. The role of sub-interpolation for Delay-Line Time-to-Digital Converters in FPGA devices. *Nucl. Instrum. Methods Phys. Res. Sect. A Accel. Spectrometers Detect. Assoc. Equip.* **2019**, *916*, 204–214. [CrossRef]
37. Kwiatkowski, P.; Szplet, R. Time-to-Digital Converter with Pseudo-Segmented Delay Line. In Proceedings of the 2019 IEEE International Instrumentation and Measurement Technology Conference (I2MTC), Auckland, New Zealand, 20–23 May 2019; pp. 1–6. [CrossRef]
38. Liu, C.; Wang, Y.; Kuang, P.; Li, D.; Cheng, X. A 3.9 ps RMS resolution time-to-digital converter using dual-sampling method on Kintex UltraScale FPGA. In Proceedings of the 2016 IEEE-NPSS Real Time Conference (RT), Padua, Italy, 6–10 June 2016; pp. 1–3. [CrossRef]
39. Wang, Y.; Zhou, X.; Song, Z.; Kuang, J.; Cao, Q. A 3.0-ps rms Precision 277-MSamples/s Throughput Time-to-Digital Converter Using Multi-Edge Encoding Scheme in a Kintex-7 FPGA. *IEEE Trans. Nucl. Sci.* **2019**, *66*, 2275–2281. [CrossRef]
40. Wu, J.; Shi, Z. The 10-ps wave union TDC: Improving FPGA TDC resolution beyond its cell delay. In Proceedings of the 2008 IEEE Nuclear Science Symposium Conference Record, Dresden, Germany, 19–25 October 2008; pp. 3440–3446. [CrossRef]
41. Qi, J.; Gong, H.; Liu, Y. On-Chip Real-Time Correction for a 20-ps Wave Union Time-To-Digital Converter (TDC) in a Field-Programmable Gate Array (FPGA). *IEEE Trans. Nucl. Sci.* **2012**, *59*, 1605–1610. [CrossRef]
42. Szplet, R.; Sondej, D.; Grzęda, G. Interpolating time counter with multi-edge coding. In Proceedings of the 2013 Joint European Frequency and Time Forum and International Frequency Control Symposium (EFTF/IFC), Prague, Czech Republic, 21–25 July 2013; pp. 321–324. [CrossRef]

43. Bayer, E.; Zipf, P.; Traxler, M. A multichannel high-resolution (<5 ps RMS between two channels) Time-to-Digital Converter (TDC) implemented in a field programmable gate array (FPGA). In Proceedings of the 2011 IEEE Nuclear Science Symposium Conference Record, Valencia, Spain, 23–29 October 2011; pp. 876–879. [CrossRef]
44. 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]
45. Daigneault, M.A.; David, J.P. A novel 10 ps resolution TDC architecture implemented in a 130nm process FPGA. In Proceedings of the 8th IEEE International NEWCAS Conference 2010, Montreal, QC, Canada, 20–23 June 2010; pp. 281–284. [CrossRef]
46. Shen, Q.; Liu, S.; Qi, B.; An, Q.; Liao, S.; Shang, P.; Peng, C.; Liu, W. A 1.7 ps Equivalent Bin Size and 4.2 ps RMS FPGA TDC Based on Multichain Measurements Averaging Method. *IEEE Trans. Nucl. Sci.* **2015**, *62*, 947–954. [CrossRef]
47. Wang, Y.; Cao, Q.; Liu, C. A Multi-Chain Merged Tapped Delay Line for High Precision Time-to-Digital Converters in FPGAs. *IEEE Trans. Circuits Syst. II Express Briefs* **2018**, *65*, 96–100. [CrossRef]
48. Kwiatkowski, P.; Szplet, R. Multisampling wave union time-to-digital converter. In Proceedings of the 2020 6th International Conference on Event-Based Control, Communication, and Signal Processing (EBCCSP), Krakow, Poland, 23–25 September 2020; pp. 1–5. [CrossRef]
49. Frankowski, R.; Gurski, M.; Szplet, R. Kintex UltraScale's multi-segment digital tapped delay lines with controlled characteristics for precise time-to-digital conversion. *Metrol. Meas. Syst.* **2024**, *31*. [CrossRef]
50. Lusardi, N.; Corna, N.; Garzetti, F.; Salgaro, S.; Geraci, A. Cross-Talk Issues in Time Measurements. *IEEE Access* **2021**, *9*, 129303–129318. [CrossRef]
51. Kuang, J.; Wang, Y.; Cao, Q.; Liu, C. Implementation of a high precision multi-measurement time-to-digital convertor on a Kintex-7 FPGA. *Nucl. Instrum. Methods Phys. Res. Sect. A Accel. Spectrometers Detect. Assoc. Equip.* **2018**, *891*, 37–41. [CrossRef]
52. Nutt, R. Digital Time Intervalometer. *Rev. Sci. Instrum.* **1968**, *39*, 1342–1345. [CrossRef]
53. Zhu, M.; Qi, X.; Cui, T.; Gao, Q. Tapped delay line for compact time-to-digital converter on UltraScale FPGA and its coding method. *Nucl. Instrum. Methods Phys. Res. Sect. A Accel. Spectrometers Detect. Assoc. Equip.* **2023**, *1056*, 168639. [CrossRef]
54. UG574, UltraScale Architecture Configurable Logic Block. Available online: https://users.ece.utexas.edu/~mcdermot/arch/articles/Zynq/ug574-ultrascale-clb.pdf (accessed on 11 November 2024).
55. Xilinx. UG949 UltraScale Device Clocking. Available online: https://docs.amd.com/r/2021.1-English/ug949-vivado-design-methodology/UltraScale-Device-Clocking (accessed on 11 November 2024).
56. UG474, 7 Series FPGAs Configurable Logic Block. Available online: https://www.eng.auburn.edu/~nelson/courses/elec4200/FPGA/ug474_7Series_CLB.pdf (accessed on 11 November 2024).
57. UG572, UltraScale Architecture Clocking Resources User Guide. Available online: https://docs.amd.com/r/en-US/ug572-ultrascale-clocking/UltraScale-Architecture-Clocking-Resources-User-Guide (accessed on 11 November 2024).
58. Kintex UltraScale FPGAs Data Sheet: DC and AC Switching Characteristics 2020. Available online: https://docs.amd.com/v/u/en-US/ds892-kintex-ultrascale-data-sheet (accessed on 11 November 2024).
59. Xilinx. UG906 Timing Analysis. Available online: https://docs.amd.com/r/en-US/ug906-vivado-design-analysis/Category-1-Timing (accessed on 11 November 2024).
60. Xilinx. Vivado Overview. Available online: https://www.xilinx.com/support/documents/sw_manuals/xilinx2022_1/ug892-vivado-design-flows-overview.pdf (accessed on 11 November 2024).
61. Xilinx. UG906 Report Utilization. Available online: https://docs.amd.com/r/en-US/ug906-vivado-design-analysis/Report-Utilization (accessed on 11 November 2024).
62. AWG4000 Series Aribitrary Waveform Generator. Available online: https://www.tek.com/en/datasheet/arbitrary-waveform-generators-awg4000-series-datasheet (accessed on 11 November 2024).
63. Costa, A.; Corna, N.; Garzetti, F.; Lusardi, N.; Ronconi, E.; Geraci, A. High-Performance Computing of Real-Time and Multichannel Histograms: A Full FPGA Approach. *IEEE Access* **2022**, *10*, 47524–47540. [CrossRef]
64. Tamborini, D.; Portaluppi, D.; Villa, F.; Zappa, F. Eight-Channel 21 ps Precision 10µs Range Time-to-Digital Converter Module. *IEEE Trans. Instrum. Meas.* **2016**, *65*, 423–430. [CrossRef]
65. Hari Prasad, K.; Chandratre, V.; Sukhwani, M. A high-density, 129-channel time-to-digital converter in FPGA for trigger-less data acquisition systems. *Nucl. Instrum. Methods Phys. Res. Sect. A Accel. Spectrometers Detect. Assoc. Equip.* **2023**, *1056*, 168657. [CrossRef]
66. Kuang, J.; Wang, Y.; Liu, C. A 128-Channel High Performance Time-to-Digital Converter Implemented in an UltraScale FPGA. In Proceedings of the 2017 IEEE Nuclear Science Symposium and Medical Imaging Conference (NSS/MIC), Atlanta, GA, USA, 21–28 October 2017; pp. 1–4. [CrossRef]
67. Acconcia, G.; Malanga, F.; Farina, S.; Ghioni, M.; Rech, I. A 1.9 ps-rms Precision Time-to-Amplitude Converter With 782 fs LSB and 0.79%-rms DNL. *IEEE Trans. Instrum. Meas.* **2023**, *72*, 2003711. [CrossRef]
68. CERN. PicoTDC. Available online: https://kt.cern/technologies/picotdc (accessed on 11 November 2024).
69. Altruda, S.; Christiansen, J.; Horstmann, M.; Perktold, L.; Porret, D.; Prinzie, J. PicoTDC: A flexible 64 channel TDC with picosecond resolution. *J. Instrum.* **2023**, *18*, P07012. [CrossRef]
70. PETsys. TOFPET2 ASIC. Available online: https://www.petsyselectronics.com/web/public/products/1 (accessed on 11 November 2024).

71. Nadig, V.; Hornisch, M.; Oehm, J.; Herweg, K.; Schulz, V.; Gundacker, S. 16-channel SiPM high-frequency readout with time-over-threshold discrimination for ultrafast time-of-flight applications. *EJNMMI Phys.* **2023**, *10*, 76. [CrossRef] [PubMed]
72. Weeroc. Temporoc. Available online: https://www.weeroc.com/read_out_chips/temporoc/ (accessed on 11 November 2024).
73. Lusardi, N.; Garzetti, F.; Costa, A.; Cautero, M.; Corna, N.; Ronconi, E.; Brajnik, G.; Stebel, L.; Sergo, R.; Cautero, G.; et al. High-Resolution Imager Based on Time-to-Space Conversion. *IEEE Trans. Instrum. Meas.* **2022**, *71*, 2004811. [CrossRef]
74. ACAM. ACAM TDC. Available online: https://www.sciosense.com/wp-content/uploads/2023/12/TDC-GP22-Datasheet.pdf (accessed on 11 November 2024).
75. Lapington, J.S.; Leach, S.A.; Sudjai, T.; Milnes, J.S.; Conneely, T.; Duran, A.; Hink, P. Investigating microchannel plate PMTs with TOFPET2 multichannel picosecond timing electronics. *Nucl. Instrum. Methods Phys. Res. Sect. A Accel. Spectrometers Detect. Assoc. Equip.* **2020**, *958*, 162758. [CrossRef]
76. Venturini, Y.; Maggio, C.; Tintori, C.; Ninci, D.; Mati, A.; Picchi, A.; Abba, A.; Locatelli, M.; Williams, C.; Williams, T.; et al. Characterization of a compact TDC unit with picosecond timing capabilities. In Proceedings of the 2024 IEEE Nuclear Science Symposium (NSS), Medical Imaging Conference (MIC) and Room Temperature Semiconductor Detector Conference (RTSD), Tampa, FL, USA, 26 October–2 November 2024. [CrossRef]
77. Milian, F.M.; Data, E.M.; Bersani, D.; Cirio, R.; Donetti, M.; Mazinani, M.F.; Ferrero, V.; Fiorina, E.; Hosseini, M.A.; Olivares, D.M.; et al. Use of the CERN picoTDC for timing application in particle therapy. In Proceedings of the 2024 IEEE Nuclear Science Symposium (NSS), Medical Imaging Conference (MIC) and Room Temperature Semiconductor Detector Conference (RTSD), Tampa, FL, USA, 26 October–2 November 2024. [CrossRef]
78. Bouchard, F.; England, D.; Bustard, P.J.; Heshami, K.; Sussman, B. Quantum Communication with Ultrafast Time-Bin Qubits. *PRX Quantum* **2022**, *3*, 010332. [CrossRef]
79. Joshi, C.; Sparkes, B.M.; Farsi, A.; Gerrits, T.; Verma, V.; Ramelow, S.; Nam, S.W.; Gaeta, A.L. Picosecond-resolution single-photon time lens for temporal mode quantum processing. *Optica* **2022**, *9*, 364–373. [CrossRef]