Experimental Demonstration of High Precision Time Transfer in FPGA for 5G and 6G Networks

Lakshmi Rajagopal*, Romerson D. Oliveira†, Andrew Lord*
Yeshpal Singh‡, Reza Nejabati†, Dimitra Simeonidou†
*Optical Networks Research Group, BT Group, Adastral Park, Ipswich, United Kingdom
{lakshmi.rajagopal, andrew.lord}@bt.com
†High Perfomance Networks Group, University of Bristol, Bristol, United Kingdom
{romerson.oliveira, reza.nejabati, dimitra.simeonidou}@bristol.ac.uk
‡Dept. of Physics and Astronomy, University of Birmingham, Birmingham, United Kingdom
{y.singh.1}@bham.ac.uk

***Abstract*—With the current roll out of 5G and efforts to accelerate the 6G framework, network and applications are evolving to a scenario where real-time requirements will be more critical and complex than today. Time transfer and synchronization mechanisms will therefore remain as a fundamental building block of the infrastructure. In this context, this paper demonstrates an FPGA-based high precision time transfer prototype to timestamp events and synchronise two nodes in a network. In the core of a time-to-digital converter, a 2048-bits delay line provides 3.9 ps of theoretical resolution and 4.97 ps achieved on-chip. The system reaches 11.7 ps scale precision when synchronizing central and remote nodes in the best case, while holding a single-shot precision of 42.9 ps. We incorporate the idea of periodically offset correction to account for the time taken for timestamps be transmitted over an optical network with several scenarios of experimentation. To compensate for physical imperfections, calibration is post-performed in software. Our results show that the system consumes less than 5% of a high-performance FPGA, which makes the solution suitable for deployment alongside multiple low-latency hardware designs for 5G and 6G components.**

***Index Terms*—TDC, Time synchronization, FPGA, 6G Networks, 5G Fronthaul**

I. Introduction

While the 5G networks have reached a mark of 145 thousands of commercially available deployments around the world [1], research institutions and operators are already exploring the 6G framework, expected to be available by 2030 [2] [3]. 6G aims to unveil a new human-centered field of applications in which the ambition to immerse the digital into physical reality will increase. Not only this will make the real-time requirements more critical, but also this immersion will bring the notions of time and simultaneity to a new level of complexity [4]. In this context, time synchronization, one of the key aspect in 5G and beyond networks [5], will certainly remain as a key component spanning from hardware to application layers.

In 5G-Advanced and 6G networks, absolute time is a key factor to ensure critical use-cases, synchronize media playout, and timestamp events to be used in machine learned contexts [6]. Today's distributed 5G RAN technology already demands a cost-effective precise timing solution in the 5G fronthaul, while in Optical Transport Networks, precise synchronisation is essential to aid in time-sensitive networking and delay monitoring. Taking a long term perspective, it is envisioned that future 6G systems will be boosted by quantum communication and computing technologies [7], a scenario where time distribution becomes essential for timestamping and synchronizing photons as realized in [8].

In modern systems, synchronization is achieved locally either with help of Global Navigation Satellite System (GNSS) receivers or transport technologies such as Precision Time Protocol (PTP), Synchronous Ethernet (SyncE) and Network Time Protocol (NTP) [9]. Currently, these systems achieve nanoseconds-precision. The high accuracy PTP White Rabbit (WR) synchronization protocol [10] goes a step further and provide sub-nanosecond accuracy and picoseconds synchronization. The work in [11], e.g., combines the WR protocol with digital dual mixer time difference (DDMTD) to achieve picosecond-precision packet timestamping in FPGA.

White Rabbit delivers its picoseconds precision with a combination of PTP, SyncE, phase detection and difference (via DDMTD), and link calibration techniques that need multiple (and complex) hardware circuitry to work efficiently. One thing to consider is that the amount of time-error added due to factors such as unknown link asymmetry and quality of syntonization from the PTP message's exchange rate can degrade precision particularly with the proliferation in number of nodes in a network.

Another way to provide time measurements in picoseconds is applying time-to-digital converter (TDC) to digitally represent the time events occur [12]. TDCs are implemented in different hardware platforms such as analogue circuits, FPGAs, and ASICs [13]. The high values of dead times in analogue circuits make FPGA and ASIC-based platforms preferred for implementing a TDC. FPGAs offer reprogrammability when compared to ASICs and they already provide high integration and complete digitalization. The work in [14] implements in FPGA a TDC with coarse and fine counts and achieves 21.2 ps

This work was supported by the Modular Systems for Advanced Integrated Quantum Clocks (MoSaiQC) project funded by the European Commission (EC) under H2020 Marie Skłodowska Curie Innovative Training Networks programme and the UK GOV DSIT (FONRC) project REASON.

978-3-903176-62-1

for synchronization of Ethernet frames between two nodes.

In 5G infrastructures, FPGAs have already been a useful platform for network planning given that they can provide accelerator for C-RAN, network slicing, characterization of massive MIMO and cognitive radio framework [15]. They have been shown as well suitable to provide secure data plane for the 5G Fronthaul [16] [17]. It is therefore plausible to integrate an FPGA-based timing solution into the hardware part of an equipment. As a result, precise timing method using FPGAs shows promise to be an effective way to achieve precise timestamping and synchronization at different nodes in a beyond 5G network.

In this context, this paper presents an experimental demonstration of a high precision time transfer technique using FPGA-based TDC which could be integrated to a 5G or 6G hardware substrate. The experiments aim to synchronise two nodes in a network to picoseconds scale precision. A novel technique for offset correction is applied, which helps to further improve the precision. The offset is computed based on two factors: 1) the time difference between timestamps generated from central and remote node, and 2) the time taken by each node to transmit and receive a timestamp over the network. By detailing the time transfer prototype realized in this research, the contributions are twofold:

- The design, resolution and synchronization precision of a fine grain TDC implemented in FPGA for a two-node network; and
- A testbed built for validation and demonstration of how the system can be used in integration with a deployed 5G or 6G xHaul.

Details about the TDC implementation and experimental testbed can be found in Section II. The results achieved from testing are reported in Section III. Section IV concludes this paper and presents the next steps of our research.

II. High Precision Time Transfer Solution

TDCs have numerous applications in network and security systems to measure sub-nanosecond time intervals. In this section, we provide the details of an FPGA implementation of a sampling TDC.

A. TDC Architectural Design

In terms or architecture, sampling TDCs can be constructed from a variety of techniques [13] such as a Vernier oscillator or a delay line [18] [19]. Because delay line TDCs make the conversion at every clock cycle, it reaches higher efficiency than the Vernier method. This work implements a tapped delay line TDC with functional architecture depicted by Fig. 1.

The delay line incorporates a 2048-bits carry chain with fast carry logic and associated flip-flops. An asynchronous signal, named trigger signal, divides the system's clock period into bins. The carry units generate output as 0's or 1's based on the number of hits they get and save each bit value to a D flip-flop. This stage's result is a thermometer code whose length is equivalent to the number of delay elements (n) used in the TDC design (n here is 2048). In an ideal case, the bin counts are uniform considering that the system's clock period is divided into equal bins. However, this is not the case in a practical scenario due to physical characteristics of the hardware in which the core is implemented. The system's clock period is divided at irregular intervals as a consequence, resulting in variation in the number of hits within the delay line elements. It results in unequal bin counts, hence varying thermometer codes possibly in every clock cycle.

The number of delay line elements and the system reference clock have direct impact on the TDC resolution, which defines the smallest measurable time by the system and is given by the equation:

$$TDC_{res} = \frac{T_{Clk}}{n} \tag{1}$$

where T_{Clk} denotes the period of sampling clock, and n is number of bins.

When implementing the TDC in two physically separated nodes, they receive a common trigger signal from the same source. Meanwhile, each node has its own local system clock. At first, the trigger signal reaches TDCs implemented in both nodes to enable timestamps be generated independently. These timestamps differ from each other resulting in poor synchronization between the central and remote nodes. Poor synchronization is as a result of the difference in timestamps and the delay over the transmission medium (expected due to signals arriving shifted in time from two different lines). Subsequently, the timestamps are exchanged between central and remote nodes to correct the differences between them. The time taken to transmit and receive a timestamp is also measured to quantify the delay in the link and network subsystem itself. These two factors are taken into account during offset's calculation. Offset refers to the value added to compensate for the drift in timestamps, and thereby improve the precision of the system. The offset after calculation is applied as a correction factor to both nodes improving the quality of synchronization. Periodically, the offset is recalculated and the correction factor is added to the timestamps to maintain the precision achieved.

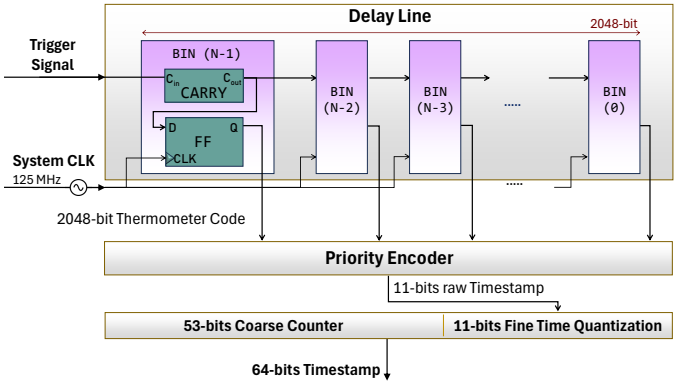

Fig. 1: Functional architectural design of the TDC implementation in FPGA with a carry chain as delay line.

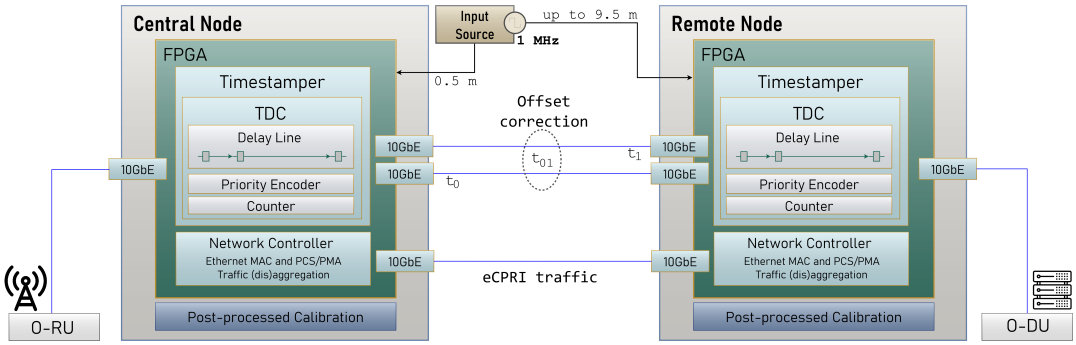

Fig. 2: Experimental setup for TDC validation with remote FPGA nodes. Input trigger signal at 1 MHz from wave function generator.

B. Experimental Setup and Validation System

The experimental testbed is detailed in Fig. 2. Two FPGAs as central (master) and remote nodes transport eCPRI data between 5G open remote unit (O-RU) and open distributed unit (O-DU). Each node hosts an AMD Virtex Ultrascale FPGA (VCU108 board) and is capable of generating timestamps with help of TDC implemented within. An FPGA Mezzanine Card (FMC) extends connectivity to eight Small Form-factor Pluggable (SFP) connectors for 10 GbE in each node. A common input signal from an external wave function generator (Rigol DG4102) is received via SMA connectors.

In terms of execution, the first step is the generation of timestamps in each individual node based on the input trigger. As previously mentioned, each node corrects its own timestamps over periodic intervals of time to enhance the system's precision (a step labeled TDC reconciliation). For the offset's calculation, the remote node communicates its local timestamps to the central node via 10 GbE link. The offset is then calculated at the central node and communicated back to the remote node – and all remaining nodes separately in case of a more complex network topology. For this process, the time taken by each node to transmit and receive timestamps is computed, reflecting the network behavior into the offset's calculation per link.

All generated timestamps are saved into AXI4-Stream FIFOs and wrapped into 64-bytes Ethernet frame before transmission. In order to determine the time difference between timestamps, measurements were taken by subtracting the stored values. As for the time taken by each node to transmit and receive a timestamp in the network, hardware counters were placed to measure the time required for frame transmission as realized in [20]. The TDC implemented in this work requires a separate synchronization channel for TDC reconciliation. In terms of deployment, this channel can be multiplexed and travel in coexistence with the data channel within the same fibre to reduce the total cost ownership.

TDC's coarse counter (Fig. 1) is driven by a 125 MHz system clock whilst a 1 MHz signal is generated from a wave function generator as the input to the carry chain. The delay line samples the input period into smaller bins and outputs the 2048-bits thermometer code, which has to be converted into binary to be usable in practice. The priority encoder then wraps the thermometer code and generates an 11-bits raw timestamp by applying logarithm to the number of hits (number of 1's) in the thermometer code. At the rising edge of the clock, raw tags are subtracted from the coarse counter value to generate a 64-bits timestamp.

III. Experimental Results

To evaluate TDC's performance, timing jitter measurements were conducted by inputting periodic triggers into the hardware and recording time differences between each step. In this study, TDC calibration is not addressed in details, nevertheless, we provide results of calibration post-processed in software.

A. TDC Resolution and Synchronization Precision

TDC's resolution is calculated as indicated in equation (1), where T_{Clk} is the clock period corresponding to 125 MHz and n is the number of delay line elements, i.e., 2048. Hence, the theoretical resolution of our TDC implementation is given by

$$TDC_{res} = \frac{8\,\eta s}{2048} = 3.9\,ps \tag{2}$$

however, later in this paper we discuss that the achieved precision is $\approx$ 1 ps far from the theoretical limit.

As stated before, after timestamps being generated, they are corrected by applying a correction factor. In this perspective, experimental scenarios (*exp*) were conducted to investigate the system's precision in both single node and distributed nodes topologies. The results are summarized in Table I, where the first two lines outline TDC's resolution. Initially, a single TDC instance in a single FPGA generated 100k timestamps from sampling a 1 MHz signal. Histogram and standard deviation were computed afterward to understand spread of time differences. The reported standard deviation is $\sigma = 60.8\,$ps, which makes the single shot precision (SSP) as:

$$SSP = \frac{\sigma}{sqrt(2)} = 42.9\,ps \tag{3}$$

marked as (*exp1*) in Table I.

TABLE I: Experimental results for TDC resolution and synchronization precision

Experiment description		Precision (ps)	Remarks
	TDC Resolution	4.97	Uncalibrated
	TDC Resolution	3.9	After calibration
Experimental scenario 1 (exp1)	Single Shot Precision	42.9	Single core TDC on single FPGA
Experimental scenario 2 (exp2)	Multiple TDC	[10.4 : 180]	On single FPGA chip
Experimental scenario 3 (exp3)	Central and remote node	[11.7 : 269.1]	Precision in a network with 2 nodes
Experimental scenario 4 (exp4)	Central and remote node	[11.7 : 760.5]	Precision in a network with 2 nodes (time shifted trigger signal)

Following that, the experiment was set up to explore the precision of two TDCs in a single FPGA (Table I (*exp2*)), where timestamps were generated from two TDC's instances in the same chip. Despite the fact that the two cores were in the same chip, timestamps were exchanged between master and slave nodes via external 10 GbE SFPs with a 1 m long optical fiber. 100k timestamps were collected and after offset correction the difference between them was computed. Fig. 3 shows a histogram plot of this difference. The interval for time error lies between 10.4 ps to 180 ps, hence, the highest precision achieved for synchronization over single FPGA chip was 10.4 ps.

The experiment was extended to explore the precision between TDCs instantiated on nodes physically separated (Table I (*exp3*)). In this scenario, timestamps were generated from both central and remote FPGAs and once again exchanged via 10 GbE. The reported interval for time error is between 11.7 ps to 269.1 ps, as shown in Fig. 4. Hence, the highest precision achieved for frame synchronization between two nodes is 11.7 ps.

In the next step, the setup was kept from *exp3* but now with an additional delay on the trigger signal to the remote node (by using a 9.5 m long cable in the signal distribution instead of the 0.5 m used so far). This scenario, Table I (*exp4*), provided us with information of the effect on synchronization precision when the trigger signal to central and remote node is manifestly shifted. In this case, the reported interval for time error is between 11.7 ps to 760.5 ps. Thus, the highest precision achieved for synchronization of frames has been left unchanged as 11.7 ps, however an significant impact (more than double) was recorded for the worst case.

The propagation delay is directly proportional to the length of the link connecting both nodes. The delay incurred to transmit and receive timestamps increases with the increase in the length of the link. This factor can impact the synchronization precision and is aggravated when yet additional time is required for buffering frames by link layer controllers. With this in mind, we measured the time taken to transmit 64-bytes frames with fiber 30 cm and 40 m (optimizing the measurement in [20] – Table I.a), and the recorded values from hardware counters are 332.8 ns and 432 ns, respectively. 40 m link with an optical cross connect in the middle resulted in 473.6 ns. In the interest of simulating, these results were combined with the timestamps generated in *exp4* and, after calculating in software the spread of time error, no impact on the synchronization precision was noticed, owing to the correction factor applied. As per the solution discussed in this paper, the synchronization precision is maintained irrespective of the changes in link length and propagation delay. In practice, the more frequently the offset and correction factor are recalculated, the better continuous synchronization among distributed nodes will be.

All results generated and reported until this point were measured from an uncalibrated TDC implementation. It is expected that factors such as routing, power fluctuation and temperature create non-linearities in the output code which eventually affects precision. It comes with the expense of possibly lowering the actual TDC's resolution. Fig. 5 shows the histogram for the 11-bits raw timestamp generated from the priority encoder. From the graph, some missing bins can

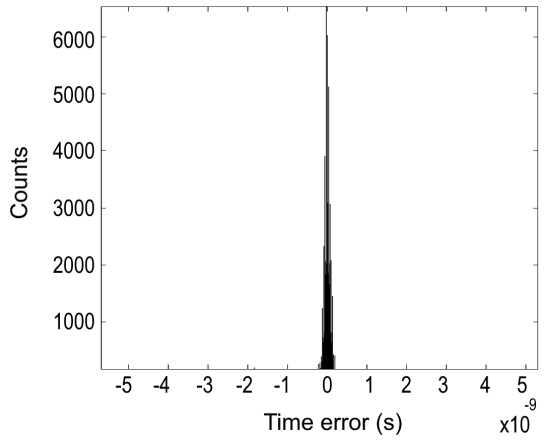

Fig. 3: Time error between timestamps after correction for two TDC instances implemented in a single FPGA.

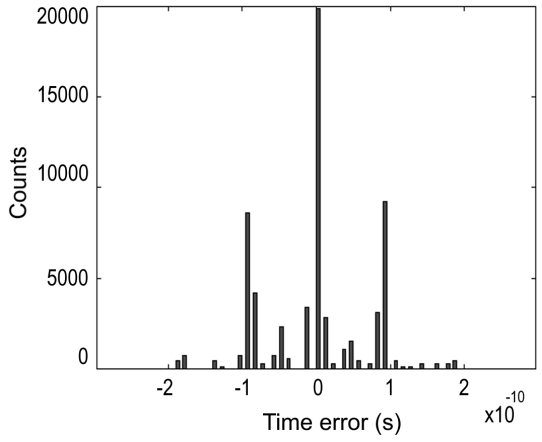

Fig. 4: Time error for between timestamps after correction for two TDC instances implemented in two FPGAs in a two-nodes network.

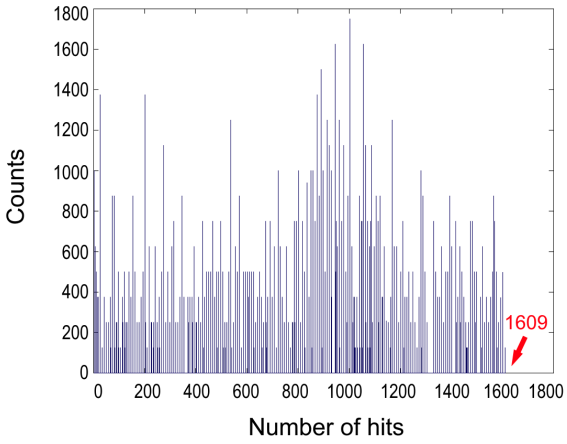

Fig. 5: Histogram of uncalibrated code. Number of bins getting hit in the carry chain, from hardware measurements, before calibration.

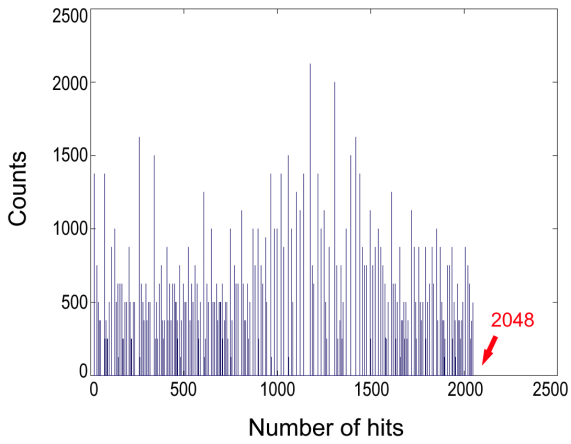

Fig. 6: Histogram of calibrated code. Number of bins getting hit in the carry chain after calibration post-processed in software.

be observed and the maximum hit is at bit 1609 instead of 2048. The actual resolution of our TDC was realized as

$$TDC_{res} = \frac{8\,\eta s}{1609} = 4.97\,ps \tag{4}$$

which is 1.07 ps far from the theoretical limit of 3.9 ps indicated by Equation 2

To compensate for non-linearities, linearization and calibration techniques such as bin by bin, direct or code density testing calibration can be applied, which will ultimately improve the precision. Fig. 6 shows the histogram of 11-bits raw timestamps from TDC after bin by bin calibration has been post-processed in software. The maximum bin number is stretched to 2048 and the resolution of TDC after calibration reaches the precision's theoretical limit of 3.9 ps.

B. Hardware Implementation

The hardware implementation reflects the architecture depicted in Fig. 1. In terms of RTL design, all codes were written in VHDL, with both bespoke blocks and IPs from AMD library available in Vivado v2019.2. The carry logic was implemented using the CARRY8 primitive (fast carry logic with look ahead) available in Ultrascale devices [21], and GTH transceivers connected the FPGA to the daughter FMC.

The resource usage in summarized in Table II. It reports two main hardware blocks, the Timestamper, which has the sampling TDC as a component underneath its hierarchy, and the FMC, the subsystem implementing eight network controllers with Ethernet MAC, PCS/PMA and Rx/Tx FIFOs for SFP interfaces. With the design prototyped in an AMD Virtex Ultrascale VU095 FPGA, the TDC consumes less than 1% of both LUTs and flip-flops, as well as CARRY8 primitives. The 2048-bits delay line itself is synthesized and implemented using less the 0.5% of CARRY8 hardware primitives. Despite the fact that four 10 GbE channels have been used per node, as depicted by Fig. 2, we maintained all eight instances to gather information about scalability.

As can be seen by Fig. 7, the design is concentrated in the top left area of the chip. The thin horizontal green

TABLE II: Hardware utilization for the prototype in FPGA.

HDL Block	Utilization (%)			
	LUT	FF	CARRY8	BRAM
Timestamper	3241 (0.6%)	6083 (0.57%)	311 (0.46%)	16.5 (0.57%)
of which TDC	3083 (0.57%)	5784 (0.54%)	295 (0.44%)	8 (0.54%)
FMC (8x10G)	29478 (5.48%)	36605 (3.4%)	344 (0.51%)	186 (3.4%)

line highlights the post-implemented carry chain, which is enveloped by a (pink) area of logic resources emphasizing the delay line block. The surrounding area in light blue is dedicated to the FMC control and all PCS/PMA, MAC and FIFO blocks. The low resource consumption endorses the feasibility of integrating the TDC core with other 5G and 6G related functions in the same chip, such as low-latency encryption, signal modulation and demodulation, channel coding and error correction, and parallel processing required for MIMO systems. The reported power consumption for the design is ≈7.6 W, including network, memory and additional debug blocks, with TDC alone responsible for ≈ 2 % of total (0.14 W).

All the logic elements and memory blocks have been automatically assigned by Vivado. Manual place and route interventions have not been made at this stage, although we hold to the assumption that they could improve the system's practical resolution combined with a hardware implementation of calibration.

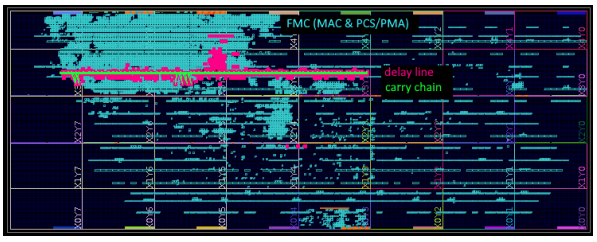

Fig. 7: Post-implementation FPGA Floorplan. Bulky blue area on top left corresponds to the FMC interface control. Pink horizontal cells are the delay line and the thin green line is the carry chain.

From Fig. 2, it can be seen that each TDC core is placed side by side to the network controller. By having both TDC and network switching inside the same FPGA, we are able to timestamp events such as Ethernet frame transmission and reception, as well as transmit the timestamps themselves to distributed nodes. This can be of use not only for synchronization, but also for cybersecurity when countermeasures are applied against timing attacks in the fronthaul. In the O-RAN fronthaul, data-embedded clock information can travel with PTP over Ethernet. In this direction, our prototype is currently able to generate timestamps within 5 ps resolution.

IV. Conclusions

In this work, we have demonstrated a prototype for high precision time transfer using FPGA across two nodes in a network. The reported TDC resolution in hardware is 4.97 ps. Software calibration was realised in software and it has improved the resolution to the theoretical limit of 3.9 ps. A precision of 10.4 ps was achieved on a single FPGA chip while central and remote nodes reached as low as 11.7 ps. The standard deviation of a single core TDC was observed as $\sigma = 60.8\,\text{ps}$, with a direct influence on the single shot precision, registered as 42.9 ps.

The synchronization precision for two TDC cores in the same chip belongs to the interval [10.4 : 180] ps. There is scope for improving the current findings and narrow down the margin between the best and worst case by bringing calibration to hardware. The same can be said about all the experimental scenarios reported in Subsection III-A. It is also important to highlight that achieving hardware calibration is a central aspect under investigation in a parallel research focused on improvements to the current design.

The work detailed in the paper used a function generator as source of the trigger signal. As for the road ahead, a stable frequency from an optical clock with a phase noise cancellation setup can be used to instantiate the TDC.

Our work is a step towards a system which considers more comprehensive timing measurements in combination with latency indicators for 6G deployments. Along with the wireless access, wired transport networks for the xHaul needs upgrading to support low latency and time critical services. In light of this, the solution demonstrated in the paper has potential to be significantly useful for a 5G or 6G architecture.

Acknowledgment

The authors would like to thank Peter Winzer (Nubis Communications) for his valuable support on the discussions and high-level design of the solution.

References

[1] Ookla. (2023, Feb) Ookla 5G map - Global 5G statistics. [Online]. Available: https://www.speedtest.net/ookla-5g-map

[2] ITU-R, "Framework and overall objectives of the future development of IMT for 2030 and beyond," International Telecommunication Union, Recommendation ITU-R M.2160-0, Nov 2023. [Online]. Available: https://www.itu.int/rec/R-REC-M.2160/en

[3] M. Uusitalo et al., "6G vision, value, use cases and technologies from european 6g flagship project hexa-x," *IEEE Access*, vol. 9, pp. 160 004–160 020, 2021, https://doi.org/10.1109/ACCESS.2021.3130030.

[4] P. Popovski, F. Chiariotti, K. Huang, A. E. Kalør, M. Kountouris, N. Pappas, and B. Soret, "A perspective on time toward wireless 6G," *Proceedings of the IEEE*, vol. 110, no. 8, pp. 1116–1146, 2022.

[5] ITU News. (2022, Dec) Synchronization technologies evolving for 5G and beyond. [Online]. Available: https://www.itu.int/hub/2022/12/synchronization-technologies-evolving-for-5g-and-beyond/#:~:text=Transport%2Dbased%20solutions%20rely%20on,frequency%20synchronization%20over%20the%20physical

[6] D. Chandramouli, P. Andres-Maldonado, and T. Kolding, "Evolution of timing services from 5G-A toward 6G," *IEEE Access*, vol. 11, pp. 35 150–35 157, 2023.

[7] C. Wang and A. Rahman, "Quantum-enabled 6G wireless networks: Opportunities and challenges," *IEEE Wireless Communications*, vol. 29, no. 1, pp. 58–69, 2022.

[8] K. Kapoor, S. Xie, J. Chung, R. Valivarthi, C. Peña, L. Narváez, N. Sinclair, J. P. Allmaras, A. D. Beyer, S. I. Davis, G. Fabre, G. Iskander, G. S. Kanter, R. Kettimuthu, B. Korzh, P. Kumar, N. Lauk, A. Mueller, M. Shaw, P. Spentzouris, M. Spiropulu, J. M. Thomas, and E. E. Wollman, "Picosecond synchronization system for the distribution of photon pairs through a fiber link between fermilab and argonne national laboratories," *IEEE Journal of Quantum Electronics*, vol. 59, no. 4, pp. 1–7, 2023.

[9] S. Ruffini, M. Johansson, B. Pohlman, and M. Sandgren, "5G synchronization requirements and solutions," *Ericsson Technology Review*, vol. 2021, no. 1, pp. 2–13, 2021.

[10] CERN. (2022, Fev) The white rabbit project. [Online]. Available: https://white-rabbit.web.cern.ch/

[11] F. Girela-López, E. Ros, and J. Díaz, "Precise network time monitoring: Picosecond-level packet timestamping for fintech networks," *IEEE Access*, vol. 9, pp. 40 274–40 285, 2021.

[12] J. Kalisz, R. Szplet, J. Pasierbinski, and A. Poniecki, "Field-programmable-gate-array-based time-to-digital converter with 200-ps resolution," *IEEE Transactions on Instrumentation and Measurement*, vol. 46, no. 1, pp. 51–55, 1997.

[13] J. Szyduczyński, D. Kościelnik, and M. Miśkowicz, "Time-to-digital conversion techniques: a survey of recent developments," *Measurement*, vol. 214, 2023, https://doi.org/10.1016/j.measurement.2023.112762.

[14] E. Arabul, R. D. Oliveira, R. Wang, R. Nejabati, and D. Simeonidou, "Experimental demonstration of integrated low-cost high-precision timing solution for optical transport networks supporting 5G," in *2023 Optical Fiber Communications Conference and Exhibition (OFC)*, 2023, pp. 1–3.

[15] V. Chamola, S. Patra, N. Kumar, and M. Guizani, "FPGA for 5G: Re-configurable hardware for next generation communication," *IEEE Wireless Communications*, vol. 27, no. 3, pp. 140–147, 2020.

[16] D. Dik and M. S. Berger, "Open-RAN fronthaul transport security architecture and implementation," *IEEE Access*, vol. 11, pp. 46 185–46 203, 2023.

[17] R. D. Oliveira, R. Wang, E. Arabul, S. Bahrani, M. Yang, C. Vrontos, S. Yan, R. Nejabati, and D. Simeonidou, "QKD-secured and dynamic multi-tenant O-RAN 5G fronthaul with quantum and classical channels co-existence," in *In Proceedings of the 49th European Conference on Optical Communications (ECOC)*. OSA, Oct 2023, In Press.

[18] E. Arabul, A. Girach, J. Rarity, and N. Dahnoun, "Precise multi-channel timing analysis system for multi-stop LIDAR correlation," in *2017 IEEE International Conference on Imaging Systems and Techniques (IST)*, 2017, pp. 1–6.

[19] G. W. Roberts and M. Ali-Bakhshian, "A brief introduction to time-to-digital and digital-to-time converters," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 57, no. 3, pp. 153–157, 2010.

[20] R. D. Oliveira, S. Bahrani, E. Arabul, R. Wang, R. Nejabati, and D. Simeonidou, "FPGA-based deterministic and low-latency control for distributed quantum computing," in *IEEE INFOCOM 2023 - IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, 2023, pp. 1–6.

[21] AMD. (2021, Oct) Ultrascale architecture libraries guide (ug974). [Online]. Available: https://docs.amd.com/r/2021.2-English/ug974-vivado-ultrascale-libraries/CARRY8