20-ps resolution Clock Distribution Network for a fast-timing single photon detector

N. Egidos, R. Ballabriga, F. Bandi, M. Campbell, D. Gascón, S. Gómez, J.M. Fernández-Tenllado, X. Llopart, R. Manera, J. Mauricio, D. Sánchez, A. Sanmukh, E. Santin

***Abstract*—The time resolution of active pixel sensors whose timestamp mechanism is based on Time-to-Digital Converters is critically linked to the accuracy in the distribution of the master clock signal that latches the timestamp values across the detector. The Clock Distribution Network that delivers the master clock signal must compensate process-voltage-temperature variations to reduce static time errors (skew), and minimize the power supply bounce to prevent dynamic time errors (jitter). To achieve sub-100ps time resolution within pixel detectors and thus enable a step forward in multiple imaging applications, the network latencies must be adjusted in steps well below that value. Power consumption must be kept as low as possible. In this work, a self-regulated Clock Distribution Network that fulfills these requirements is presented for the FastICpix single photon detector – aiming at a 65nm process. A 40 MHz master clock is distributed to 64x64 pixels over an area of 2.4x2.4 cm^2 using digital Delay-Locked Loops, achieving clock leaf skew below 20 ps with a power consumption of 26 mW. Guidelines are provided to adapt the system to arbitrary chip area and pixel pitch values, yielding a versatile design with very fine time resolution.**

***Index Terms*— Clock synchronization, Delay-locked loop, Fast timing, Phase Detector, Random jitter, Skew.**

I. Introduction

Active pixel detectors with very fine time resolution are an attractive alternative in a wide range of fast-timing imaging systems, such as medical diagnosis with positron-emission tomography (PET); molecular studies with Mass Spectrometry Imaging (MSI); and particle tracking in high-energy physics (HEP). There is a lot of active research aimed at developing detectors with sub-100ps time resolution, which can enable millimetric spatial resolution and real-time image processing; enhance molecule discrimination; and time tag an increasing number of particle collisions accurately, amongst others [1-5].

Manuscript received November 25, 2020. This work was supported by the ATTRACT project funded by the EC under Grant Agreement 777222.

N. Egidos (e-mail: nuria.egidos.plaja @ cern.ch), R. Ballabriga (e-mail: Rafael.Ballabriga @ cern.ch), M. Campbell (e-mail: Michael.Campbell @ cern.ch), J.M. Fernández-Tenllado (e-mail: jose.fernandez @ cern.ch), X. Llopart (e-mail: Xavier.Llopart @ cern.ch) are with CERN, 1211 Meyrin, CH.

F. Bandi is with IMSE-CNM (CSIC-Universidad de Sevilla), 41092 Sevilla, ES (e-mail: nahuel @ imse-cnm.csic.es).

D. Gascón (e-mail: dgascon @ fqa.ub.edu), S. Gómez (e-mail: sgomez @ fqa.ub.edu), R. Manera (e-mail: rafelmanera @ icc.ub.edu), J. Mauricio (e-mail: jmauricio @ fqa.ub.edu), D. Sánchez (e-mail: dsanchez @ fqa.ub.edu), A. Sanmukh (e-mail: asanmukh @ fqa.ub.edu) are with ICCUB, 08028 Barcelona, ES.

E. Santin was with CERN. He is now with AlpsenTek GmbH, 8050, Zürich, CH (e-mail: edineisantin @ gmail.com).

In the readout electronics, a Time-to-Digital Converter (TDC) can be used per group of pixels to time stamp the particle arrival. TDCs are dispersed across the pixel matrix and synchronized by means of a shared time reference (master clock). This signal is delivered by means of a Clock Distribution Network (CDN).

Due to process, voltage or temperature (PVT) variations, the circuit elements that compose the CDN may have a slightly different delay in the various branches. As a result of these non-idealities, there is a static time error or skew in the actual latencies or propagation delays from the source to the TDCs. On top of this variability, the delays will also be dynamically affected by perturbations in the supply voltage, voltage droop, or temperature gradients during operation; and due to noise coupled mainly from the power supply due to the switching activity of the circuitry (a.k.a. Power Supply Induced Jitter or PSIJ). These effects manifest as jitter on the clock edges. Jitter can also enter the CDN superimposed to the clock source, as a result of the non-idealities of the clock generator [3].

With the goal of an accurate clock distribution, which is indispensable for a reliable timestamp, the CDN must include mechanisms to self-regulate the latencies, so as to reduce the impact of skew and jitter. In this work, such a CDN is proposed for the FastICpix chip [7-8]. This ATTRACT phase-I funded project consists of a reconfigurable single photon pixel detector that can be tailored in area to different applications by means of adaptable pixel pitch and front-end signal summation, while providing a very fine single photon time resolution (SPTR). The target SPTR (10 ps_{RMS}) motivates a 20 ps TDC time bin. To achieve this time resolution, the latency of the CDN branches can be adjusted in steps finer than 20 ps, so that the maximum time error in the timestamp due to the CDN is ± 1 TDC count. Since the CDN adapts to the chip area and pixel pitch, the concept is also suitable for other designs that pursue a comparable time resolution.

In this work, the CDN requirements and some architectural alternatives are discussed in section II. The selected architecture is described in section III. Guidelines are provided to scale the design to arbitrary chip area and pixel pitch values in section IV, and the main contributions to the time errors are described in section V. To reduce the impact of such errors, a strategy to update the CDN latencies is described in section VI. The circuit simulated performance is summarized in section VII, followed by a discussion on the obtained results.

II. TOWARDS A PROPOSAL OF CDN ARCHITECTURE

The CDN architecture must fulfill these conditions:

1. Adaptability to chip area (area across which the CDN spans) and pixel pitch (number of sinks or target TDCs).
2. Time error due to the CDN at each of its sinks lower than the TDC time bin (20 ps). This implies that a) the total time error at each sink must be below 20 ps, and b) the latency must be adjustable in steps finer than 20 ps.

CDNs have traditionally exploited the network symmetries to limit skew [9-10]. However, open-loop strategies (trees, meshes, spines, etc.) become insufficient to achieve the aforementioned time errors in the envisaged large chip areas (few cm^2). A solution based on free-running, mutually coupled oscillators distributed across the chip (the output of which becomes the master clock delivered to the corresponding TDC) has the potential to reduce both static and dynamic time errors to the required margin [4]. However, the associated power consumption may be a concern in the largest chip areas. Alternatively, Delay-Locked Loops (DLLs) can reduce skew in a wide range of areas [13-15]. In some solutions, a local control action is applied to compensate skew between adjacent sinks, with the DLLs embedded into an H-tree or a mesh structure [5]. This may result in area and power overhead with respect to using a controller per branch. Besides, since the individual control actions are not synchronized, a stable latency from the clock source to the sinks cannot be guaranteed across PVT corners. As a result, it cannot be ensured that the clock will arrive distributed during one period to the different sinks, which might lead to PSIJ; and the timestamp error associated to the CDN can only be bound at a local level. These non-idealities are prevented in the Timepix4 pixel detector [6]: the CDN branches consist of digital DLLs (dDLLs) and local clock trees to distribute a 40 MHz master clock across an area of close to 7 cm^2 with a skew in the order of 100 ps. Digital low-pass filtering is used to reduce the impact of jitter. The aforementioned alternatives are benchmarked in Table I (see Appendix C for further details). The Timepix4 CDN has been selected as a starting point for this work. The complexity of scaling the CDN has been addressed by designing several dDLL flavors, as it will be seen in section IV. Besides the low time errors, power consumption and area overhead, this solution features a stable latency from the clock source to the sinks, which provides robustness to PVT variations in the TDC timestamp measurement; and it ensures that the clock arrival is distributed during one period, thus preventing PSIJ.

TABLE I
BENCHMARK OF SEVERAL CDN CONFIGURATION ALTERNATIVES

CDN config.	Power scaled to 40 MHz (mW/cm^2)	Area (% w.r.t. chip area)	Largest skew (ps)	Ease of scalability with chip area and pixel pitch
dDLLs [6]	25	2	100	dDLL flavors, complexity of the local clock trees
Mut. Coupled Oscil. (8x8) [4]	152	0.5	150 (600 Ω coupling resist.)	Interconnect more oscillators
Local de-skewing [5]	-	56 times more PDs than [16]	13	Interconnect more PDs and compensators
Grid [15]	0.62	-	75	Potentially tool-automated

III. FASTICPIX CDN ARCHITECTURE

Fig. 1 shows the CDN structure for the largest envisaged chip area (2.4x2.4 cm^2). The clock source is an external reference; it is located at the center of the chip and distributed to the CDN branches by means of a clock tree. The branches, which consist of dDLLs, span across half the chip height and are mirrored with respect to the opposite half. In small chip areas, the clock source is located in a side periphery and the branches span across the full chip height. An overview of the dDLL structure is provided on the right of Fig. 1. It consists of a phase detector (PD); a digitally-controlled delay line (DCDL) whose nominal delay is 1 master clock period; and a controller ("Ctrl") that provides the bits to regulate the DCDL delay. In this figure, the DCDL includes 32 Adjustable Delay Buffers (ADBs), highlighted in blue, half of them guiding the clock upwards in a column of pixels (U0-U15), and the other half driving it downwards in an adjacent column of pixels (D15-D0). The output of each ADB drives a local clock tree to deliver the clock to a group of TDCs (4 TDCs in this case, although this number will depend upon the pixel pitch). The dDLL structure is shown in more detail in Fig. 2, and its principle of operation will be explained next.

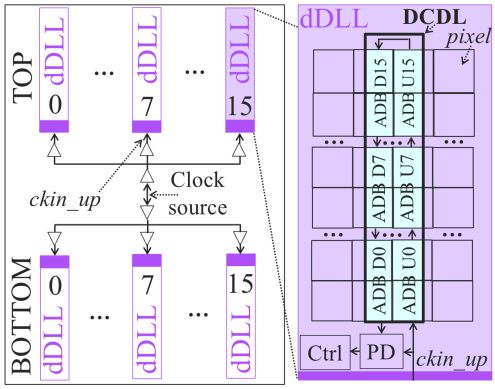


Fig. 1. Sketch of the CDN structure for large chip areas.

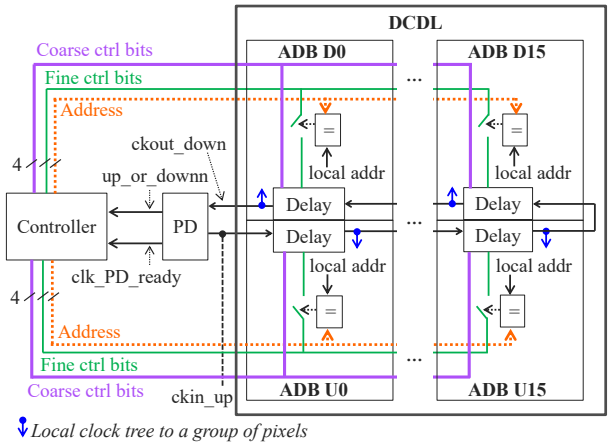


Fig. 2. High-level diagram of the dDLL structure.

The PD compares the rising edge of the clock entering the DCDL (*ckin_up*), which comes from the clock source, to the

rising edge at the output of the DCDL (*ckout_down*). A timing diagram illustrating the operation of the PD is shown in Fig. 3. If the output edge arrives earlier than the input edge (the delay of the line is shorter than 1 master clock period), the *up_or_downn* output is set to 1 so that the controller increases the delay of the line. In the case where the output edge arrives later than the *ckin_up* edge, the *up_or_downn* output is cleared to 0 to reduce the line delay. The time resolution of the PD is ~ 2 ADB LSB, and it changes accordingly with PVT corners. Only if the separation between the input and output edges is larger than ± 1 ADB LSB, a pulse is generated at the *clk_PD_ready* output, and its rising edge triggers the synchronous, finite state machine (FSM) of the controller.

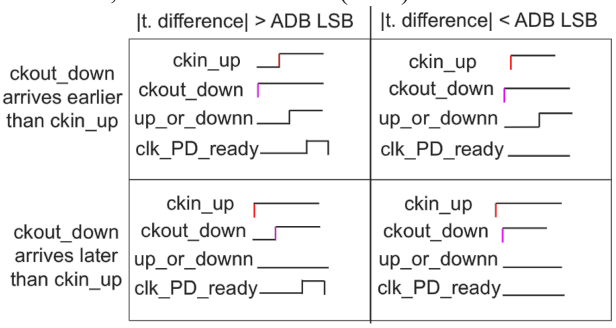


Fig. 3. Operation principle of the phase detector.

The PD outputs are digitally low-pass filtered to reduce the impact of jitter. Since the delivery of the master clock is distributed during one master clock period, the power supply pull is spread out throughout the period, which prevents PSIJ. According to the *up_or_downn* value, the controller will update the delay of the line by changing the control bits of the ADBs until the total delay is 1 master clock period ± 1 ADB LSB (lock is achieved). The adjustable delays are regulated by means of digital lines. They are composed of a coarse section (largest LSB is 80 ps in the slow corner), which is updated simultaneously in all stages via the coarse control bits, and a fine section (largest LSB is 7 ps in the slow corner), which can be regulated independently with the fine control bits and hence provide a fine adjustment of the line delay. To regulate the fine sections individually, the controller broadcasts the fine control bits and the address to be updated; the last is compared to the local address of each ADB and, if the comparison is successful, the value of the fine control bits is loaded to the selected stage. The delay cells were originally designed for Timepix4 and consist of full custom blocks that have been characterized with Cadence® Liberate™ to be integrated in the digital-on-top implementation flow.

The demonstrator dDLL is implemented for a chip of 2.4x2.4 cm^2 (64x64 pixels with 376 µm pixel pitch). This architecture can be applied to smaller chip sizes, for which an even better timing performance could be expected. A commercial 65nm process will be used, with 1.2 V voltage supply. The DCDL is composed of 32 ADBs, one per group of 4 pixels, and the master clock frequency is 40 MHz.

IV. CDN SCALABILITY WITH CHIP AREA AND PIXEL PITCH

The presented architecture can be adapted to different chip area and pixel pitch dimensions as follows:

- The number of DCDL stages and dDLLs increases with the chip area. To limit the number of DCDL flavors to be implemented, two situations are proposed: for small chip areas (up to 1.2x1.2 cm^2), the master clock source is located on one side of the chip and the DCDLs span across the full chip height, while for greater chip areas the clock source is at the center of the chip, as in Fig. 1.
- The same ADB design can be used in all cases, except for the smallest chip area. In this case, the ADB introduces half the delay by reducing the coarse section contribution. This choice is explained in the next point.
- To reuse the ADB design for different chip areas, the master clock frequency increases for shorter DCDL lengths, so that the total delay can be adjusted to 1 period. The TDCs are based on a ring oscillator running at 2 GHz with a tap delay of 20 ps (the TDC time bin). The change in the master clock frequency will have an impact on the TDC output count. To avoid using TDCs with different measurement ranges, and to limit the required ADB flavors, the variation in the master clock frequency is limited to a factor 2 across the range of used frequencies.
- The same PD design can be used in all cases.
- The same controller design can be used in all cases (the ADB indexing shall be adapted to the DCDL length).

Table II compiles numeric examples of these guidelines.

TABLE II
GUIDELINES TO SCALE THE CDN WITH THE CHIP AREA

Chip area (cm^2)	Number of pixels (pixel pitch = 376 µm)	Number of DCDL stages	Master clock frequency (MHz)	Number of dDLLs in the CDN
0.3x0.3	8x8	8 [a]	80 [c]	2
0.6x0.6	16x16	16 [a]	75	4
0.9x0.9	24x24	24 [a]	50	6
1.2x1.2	32x32	32 [a]	40	8
1.5x1.5	40x40	20 [b]	60	20
1.8x1.8	48x48	24 [b]	50	24
2.1x2.1	56x56	28 [b]	45	28
2.4x2.4	64x64	32 [b]	40	32

[a]Clock from one side of the chip (dDLL spans across full chip height).
[b]Clock from the center of the chip (dDLL spans across half the chip height).
[c]The ADB introduces half the delay in the rest of chip areas, so that the maximum spread in the range of master clock frequencies is bound to a factor 2 between the largest and the smallest frequencies.

Adaptation to the pixel pitch is handled at the local clock tree that starts at the output of each ADB and drives the TDCs in the corresponding group of pixels. For 376 µm pixel pitch, this clock tree drives 4 TDCs. For a smaller pitch, and for the same chip area, the number of sinks to be served by the local clock tree will increase by a certain factor (376 µm/new pixel pitch). The variation in power consumption associated to the different chip areas will be discussed in section VII.C.

V. CDN TIME RESOLUTION

The main contributions to the dDLL time errors are the non-idealities of the DCDL and PD, as well as jitter. Section V.A introduces the time errors associated to the DCDL and the controller, while section V.B is focused on the PD.

A. Time errors in the DCDL

A different latency or propagation delay from the clock source to the output of the ADBs causes skew or time offset between sinks, which has two components: skew by design (the arrival of the master clock is distributed over a clock period along the line), which can be compensated offline; and the static time error on top of the skew by design. The second is due to the following factors: 1) differences in the layout of the ADBs; 2) cell delay variation over PVT corners; and 3) divergence in the value of fine control bits along the line when lock is achieved, since the fine sections are regulated independently. A useful figure to understand the impact of skew is the Integral Non-Linearity (INL) of the DCDL when lock is achieved, which is calculated as:

$$INL(k) = \sum_{i=U1..k} DNL(i) \quad (1)$$

$$DNL(k) = [l(k) - l(k-1)] - [l_i(k) - l_i(k-1)] \quad (2)$$

With l the actual latency, l_i the ideal latency, k, i the indexes representing the ADBs from U1 onwards [7]. Note that in this work the INL will be expressed in time units (picoseconds), and not normalized to the LSB.

The ideal latency is obtained when all stages introduce the same delay (it represents the skew by design). Hence the INL provides the distance between the ideal and actual latencies or, in other words, the static time error to be minimized.

With this purpose, the ADBs are carefully laid out to ensure the physical symmetry between the stages that propagate the clock upwards in the column of pixels (U0, U1…) and those that propagate it downwards (… D1, D0). And the controller follows an algorithm to update the fine sections in such an order that seeks to reduce the INL associated to the divergence in the fine control bit values along the line when lock is achieved. This algorithm will be explained in section VI.

Concerning dynamic time errors, the aim of this work is to provide a budget for jitter, which is modelled by adding a dynamic variation to the edges of *ckin_up*. The half period of this signal changes as (*ideal half period of the master clock* + *random delay*), where *random delay* is a random magnitude with Gaussian distribution and 0 mean. Different values of standard deviation of this magnitude are considered, to determine which is the largest variability for which the time error target is still met. The highest total time error must be bound to the TDC time bin:

$$\max(|INL(k)|) + 3\sigma_j < 20\ ps \quad (3)$$

Where max(|INL(k)|) represents the maximum of the absolute value of the INL among all stages; and σ_j is the standard deviation of jitter. Since a Gaussian distribution is considered to model jitter, the variability is expected to be comprised within 3 standard deviations (three-sigma rule of thumb [8]).

The presence of random jitter leads to the PD behaviour explained in section V.B. This type of jitter is expected from the clock source, due to supply and temperature variations, etc. The clock lines are shielded to prevent the injection from (and to) other periodic signals, thus preventing periodic jitter.

B. Time errors in the phase detector

To understand the origin of the PD non-idealities, an overview of its architecture (sketched in Fig. 4) will be provided first. A fully digital PD architecture has been selected, which is the most suitable for the digital-on-top approach followed for the dDLL implementation. The detection range is ± half the master clock period [9]. Standard cell flip-flops (FFs) sample the time difference between the input and output clocks of the DCDL [16,20-21]. Since these signals can have an arbitrary time difference depending on the delay of the DCDL and jitter, there can occur setup-and-hold time violations in such FFs, which could lead to a metastable output. The propagation of a metastable signal is prevented by adding a second FF in a row, which samples the output of the first after a certain time, so that the metastable signal collapses to a stable 0 or 1 (which of the two cannot be foreseen) [10]. This yields a 2-FF synchronizer [11], denoted by B3 and B4 in Fig. 4. B3 is used to determine whether the delay of the line should increase or decrease (*up_or_downn_aux* should be 1 or 0, respectively).

B1 and B2 are the same fine delay cells used to compose the fine section of the ADBs. The first introduces the smallest available delay plus ADB LSB, while the second introduces the smallest available delay. In practice, this means that an artificial offset of ADB LSB is introduced between the inputs of the 2-FF synchronizers denoted by B4. The purpose of this offset is to define the ± ADB LSB target resolution window. Using the same cells as in the ADB enables tracking the variation of ADB LSB with the PVT corners.

Ideally, (a) will be 1 if *ckout_down* arrives later than *ckin_up* by a time difference larger than ADB LSB, while (b) will be 1 if *ckout_down* arrives earlier than *ckin_up* by a time difference larger than ADB LSB. These signals will be 0 if the aforementioned time differences are smaller than ADB LSB.

As a result, the OR of (a) and (b) will be high only when the time difference between *ckin_up* and *ckout_down* is larger than ADB LSB (in absolute value), indicating that a pulse should be generated in *clk_PD_ready_aux*.

The generation of this pulse is triggered with the falling edge of the last clock to arrive, either *ckin_up* or *ckout_down*, which is selected with a multiplexer and some auxiliary logic in the "Generate trigger" block. This choice of polarity and clock enables that the involved signals are stable when the trigger signal is to be selected, thus yielding a valid stimulus.

One *clk_PD_ready* pulse should be generated per input clock pulse, as long as the time difference between *ckin_up* and *ckout_down* is larger than the sensitivity window of the PD.

Due to the jitter superimposed to *ckin_up* (which is propagated to *ckout_down*), the sampled time difference is distorted. And when setup-and-hold time violations occur in the first FF of the synchronizers, its output, although having a stable value,

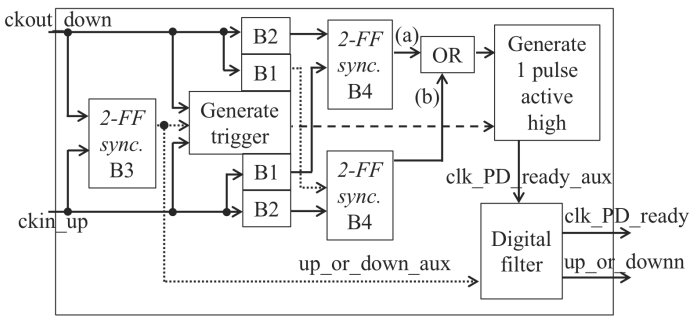


Fig. 4. Overview of the PD architecture.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TNS.2021.3057581, IEEE Transactions on Nuclear Science

might not have the right polarity. These two effects are reflected as a ringing in *up_or_downn_aux*; and in the OR of (a) and (b), which leads to the presence of *clk_PD_ready_aux* pulses when they should not be generated, or their absence when they should be generated. A digital low-pass filter has been implemented to mitigate the errors in *up_or_downn_aux* and *clk_PD_ready_aux*. Its impact is shown in Fig. 5.

The top half of the figure represents the outputs of an ideal PD. The line delay is swept from values lower than the master clock period (*ckout_down* arrives earlier than *ckin_up*), for which *up_or_downn* is 1; towards values larger than one period (*ckout_down* arrives later than *ckin_up*), for which *up_or_downn* is 0. *clk_PD_ready* pulses are generated when *ckout_down* arrives earlier (later) than *ckin_up* by an amount larger than ADB LSB, which is labelled as *S(E)*. *S* and *E* represent the start (-ADB LSB) and end (+ADB LSB) of the ideal sensitivity window of the PD, or the range of time differences for which no *clk_PD_ready* pulse is generated.

The bottom half of the figure represents the actual behavior of the PD in the presence of jitter and taking into account the setup-and-hold time violations of the FFs. Both effects are reflected in the un-filtered outputs, *up_or_downn_aux* (ringing) and *clk_PD_ready_aux* (generation of a pulse for small time differences or absence of a pulse for large time differences). As a result, lock cannot be achieved: the *clk_PD_ready_aux* pulses trigger the controller and force a continuous change in the delay of the line, toggling between incrementing and decrementing 1 ADB LSB.

up_or_downn_aux and *clk_PD_ready_aux* are low-pass filtered to reduce the ringing in the first; and to reduce the range with wrong pulse generation (i.e. the range of time differences between points *S* and *E*), so that the sensitivity window after the filter approaches ± ADB LSB. The digital filter works as follows: if the value of *up_or_downn_aux* remains stable for W consecutive *clk_PD_ready_aux* pulses, one pulse is generated at *clk_PD_ready* and this value of *up_or_downn_aux* is propagated to *up_or_downn*. If the value of *up_or_downn_aux* toggles before completing the filter window, the count is reset and neither *up_or_downn* nor *clk_PD_ready* are updated. W (depth of the filter window) has been set to 16, the smallest value that yields the required time resolution after the filter, as it will be shown in section VII.

The PD layout must prevent distorting the time difference between the input and output clocks of the line. On the one hand, the internal clock paths must be symmetric; and the parasitic load in the interface PD-DCDL must match the load of the interconnection between ADBs inside the DCDL.

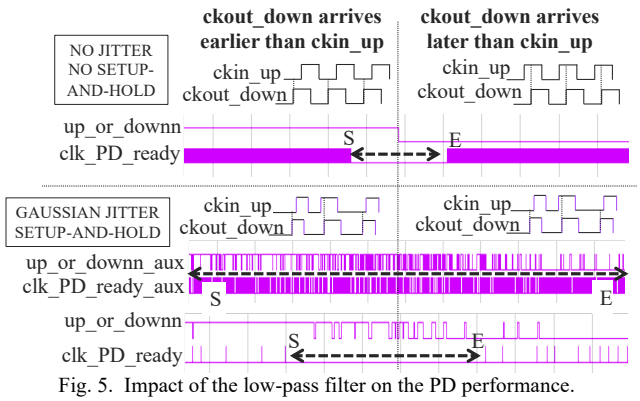


Fig. 5. Impact of the low-pass filter on the PD performance.

Summarizing the concepts introduced in this section, the ideal time resolution of the PD is ± ADB LSB, but it can be deteriorated due to the following sources of time error:

1. Node capacitance and resistance in the connection to the DCDL: the routing of *ckout_down* and *ckin_up* must be symmetric and introduce the same parasitics as the interconnection between the intermediate stages of the DCDL. Otherwise, an artificial offset is added to the time difference of interest.
2. Setup-and-hold window of the FFs that sample the time difference between *ckin_up* and *ckout_down*.
3. The jitter superimposed to *ckin_up*, which is propagated and thus observed at *ckout_down* as well. Jitter distorts the time difference to be measured and causes ringing in the PD outputs, which forces the unnecessary update of the controller and prevents the achievement of lock.

The impact of effect 1. can be reduced with a careful layout; 2. and 3. can be mitigated by low-pass filtering the PD outputs.

VI. ALGORITHM TO UPDATE THE FINE CONTROL BITS TO MINIMIZE THE DCDL STATIC TIME ERROR

The controller can update the fine control bits of the ADBs individually by selecting the address of the concerned stage and sending the new value of fine control bits. This enables the fine adjustment of the latencies in steps of ADB LSB; but it also opens the door to suffering static time error (INL) on the intermediate stages of the DCDL. To understand the impact of the fine control bit distribution along the line on the INL, an ideal DCDL of 8 stages is considered in this introduction. To achieve lock, four of the stages have their fine control bits at 0, and the other four have their fine control bits set to 1. Fig. 6 shows the DCDL INL for different distributions of the fine control bits along the line, as indicated in the subplot title. The shape of the error is relevant at this point, not its magnitude. From Fig. 6 we can conclude that:

1. The INL depends on the distribution of fine control bits along the line.
2. It is minimized when different values of fine control bits are evenly distributed (e.g. Fig. 6 (c) and Fig. 6 (f)).

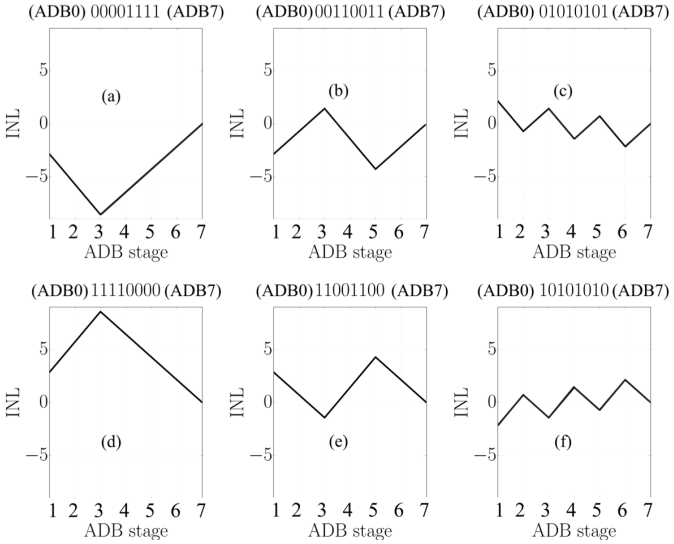


Fig. 6. INL of an example DCDL of 8 stages and different combinations of fine control bit values along the line.

The order in which the controller updates the ADB fine control bits until a given distribution is reached is called update sequence. It should guarantee that, every time the line delay is incremented or decremented by 1 ADB LSB, the new and the former fine control bit values are distributed as evenly as possible along the line. Next an algorithm is proposed to determine an update sequence that:

1. Can be implemented with binary logic, yielding a low area, power consumption and latency associated to the control action.
2. Ensures that only 1 ADB is modified between consecutive delay settings, while the rest of stages retain the former value of fine control bits. This prevents transient fluctuations in the line delay, and thus avoids switching noise and increasing the jitter to be handled by the PD.
3. The resulting INL when lock is achieved is lower than the TDC time bin, 20 ps.

The algorithm is first derived in section VI.A for a DCDL of 4 stages, and then expanded to lines of arbitrary length in section VI.B. The controller performs a random access to the fine control bits of one stage at a time, following the update sequence defined by this algorithm.

A. Updating the fine control bits of a 4-stage DCDL

This section is focused on an example DCDL composed of 4 stages (ADB_0 to ADB_3), whose fine control bits can take the value 0 or 1. The aim is to define an optimal update sequence, which minimizes INL when lock is achieved. Initially, the controller clears the fine control bits of all stages to 0, and then proceeds to set them to 1, one stage at a time, until lock is achieved. Depending on which stage is updated first, there are 4 possible update sequences that pursue an even distribution of the fine control bits along the line. The 4 options are shown in Table IX (Appendix A). These alternatives have been expanded to the demonstrator DCDL size (32 stages) following the indications that will be provided in section VI.B, and the resulting dDLLs have been simulated. Update sequence B yields the best performance in terms of INL (see Fig. 9 in Appendix B), so it will be used from here on as the optimal update sequence. Table III compiles the optimal update sequence of the 4-stage DCDL and the evolution of the fine control bits along the line as the sequence is applied.

The stages can be addressed by means of a 2-bit "Ordering code" depending on their location along the line. The optimal sequence can be implemented by means of a 2-bit binary ripple counter, also shown in the table for convenience. Each word of the 2-bit binary counter is translated to the stage address by means of the mapping function: $o_1 = b_0$, $o_0 = 1\text{-}b_1$.

TABLE III
ALGORITHM TO UPDATE THE FINE CONTROL BITS (4-STAGE DCDL)

Evolution of the fine control bits				Up-date seq.	Ordering code		Binary counter	
ADB_0	ADB_1	ADB_2	ADB_3		o_1	o_0	b_1	b_0
0	1	0	0	ADB_1	0	1	0	0
0	1	0	1	ADB_3	1	1	0	1
1	1	0	1	ADB_0	0	0	1	0
1	1	1	1	ADB_2	1	0	1	1

B. Updating the fine control bits for lines of arbitrary length

The algorithm explained in section V.A will first be expanded to the longest DCDL (32 stages), which can be addressed with 5 bits, and then generic expressions will be provided for the case of N-bit ordering codes (in the case of FastICpix, N ∈ [3,5] for the DCDL lengths defined in Table II).

Here the fine control bits of the stages will take values 0 or 1 to simplify the algorithm, but the actual controller can replace 0, 1 for any pair of consecutive values than can be covered with the 4-bit control words.

The 32-stage DCDL is divided into 4 quartiles, Q_0 (which comprises ADB_0 to ADB_7) up to Q_3 (which comprises ADB_{24} to ADB_{31}). Analogously to the optimal sequence defined in section V.A, these quartiles will be updated starting with Q_1, then Q_3, Q_0 and Q_2. This is equivalent to applying the 2-bit ordering code defined in Table III to positions MSB (o_4) and MSB-1 (o_3) of the 5-bit ordering code.

Inside each quartile, the 8 corresponding stages are divided into sub-quartiles, which will also be updated following the aforementioned order. This is equivalent to applying the 2-bit ordering code defined in Table III to positions o_2 and o_1 of the 5-bit ordering code (for every o_4o_3 combination).

Finally, the order in which the 2 stages belonging to a sub-quartile is updated does not impact the peak of the INL, only its sign. This means that the LSB of the ordering code will be 0 for half the range and 1 for the other half, and which half comes first does not impact the resulting static time error.

In Table IV, the optimal update sequence is shown on the right of the evolution of fine control bits along the 32-stage DCDL. An even distribution of the initial and final fine control bit values is achieved in the middle of the update sequence.

TABLE IV
EVOLUTION OF FINE CONTROL BITS ALONG THE LINE AND OPTIMAL UPDATE SEQUENCE FOR THE 5-BIT ORDERING CODE

0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	
0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	ADB10
0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0	ADB26
0 0 1 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0	ADB2
0 0 1 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 1 0 0 0 0 0	ADB18
0 0 1 0 0 0 0 0 0 0 1 0 0 0 1 0 0 0 1 0 0 0 0 0 0 0 1 0 0 0 0 0	ADB14
0 0 1 0 0 0 0 0 0 0 1 0 0 0 1 0 0 0 1 0 0 0 0 0 0 0 1 0 0 0 1 0	ADB30
0 0 1 0 0 0 1 0 0 0 1 0 0 0 1 0 0 0 1 0 0 0 0 0 0 0 1 0 0 0 1 0	ADB6
0 0 1 0 0 0 1 0 0 0 1 0 0 0 1 0 0 0 1 0 0 0 1 0 0 0 1 0 0 0 1 0	ADB22
0 0 1 0 0 0 1 0 1 0 1 0 0 0 1 0 0 0 1 0 0 0 1 0 0 0 1 0 0 0 1 0	ADB8
0 0 1 0 0 0 1 0 1 0 1 0 0 0 1 0 0 0 1 0 0 0 1 0 1 0 1 0 0 0 1 0	ADB24
1 0 1 0 0 0 1 0 1 0 1 0 0 0 1 0 0 0 1 0 0 0 1 0 1 0 1 0 0 0 1 0	ADB0
1 0 1 0 0 0 1 0 1 0 1 0 0 0 1 0 1 0 1 0 0 0 1 0 1 0 1 0 0 0 1 0	ADB16
1 0 1 0 0 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 0 0 1 0 1 0 1 0 0 0 1 0	ADB12
1 0 1 0 0 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 0 0 1 0 1 0 1 0 1 0 1 0	ADB28
1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 0 0 1 0 1 0 1 0 1 0 1 0	ADB4
1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0	ADB20
1 0 1 0 1 0 1 0 1 0 1 1 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0	ADB11
1 0 1 0 1 0 1 0 1 0 1 1 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 1 1 0 1 0	ADB27
1 0 1 1 1 0 1 0 1 0 1 1 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 1 1 0 1 0	ADB3
1 0 1 1 1 0 1 0 1 0 1 1 1 0 1 0 1 1 1 0 1 0 1 0 1 0 1 1 1 0 1 0	ADB19
1 0 1 1 1 0 1 0 1 0 1 1 1 0 1 1 1 0 1 1 1 0 1 0 1 0 1 1 1 0 1 0	ADB15
1 0 1 1 1 0 1 0 1 0 1 1 1 0 1 1 1 0 1 1 1 0 1 0 1 0 1 1 1 0 1 1	ADB31
1 0 1 1 1 0 1 1 1 0 1 1 1 0 1 1 1 0 1 1 1 0 1 0 1 0 1 1 1 0 1 1	ADB7
1 0 1 1 1 0 1 1 1 0 1 1 1 0 1 1 1 0 1 1 1 0 1 1 1 0 1 1 1 0 1 1	ADB23
1 0 1 1 1 0 1 1 1 1 1 1 1 0 1 1 1 0 1 1 1 0 1 1 1 0 1 1 1 0 1 1	ADB9
1 0 1 1 1 0 1 1 1 1 1 1 1 0 1 1 1 0 1 1 1 0 1 1 1 1 1 1 1 0 1 1	ADB25
1 1 1 1 1 0 1 1 1 1 1 1 1 0 1 1 1 0 1 1 1 0 1 1 1 1 1 1 1 0 1 1	ADB1
1 1 1 1 1 0 1 1 1 1 1 1 1 0 1 1 1 1 1 1 1 0 1 1 1 1 1 1 1 0 1 1	ADB17
1 1 1 1 1 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 1 1 1 1 1 1 1 0 1 1	ADB13
1 1 1 1 1 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 1 1 1 1 1 1 1 1 1 1	ADB29
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 1 1 1 1 1 1 1 1 1 1	ADB5

The 5-bit ordering code corresponding to this update sequence is shown in Table V. It can be implemented by means of a 5-bit binary ripple counter, which is also shown in the table for convenience, by applying the bit mapping shown in the rightmost column, for N = 5. This mapping function can also be applied to the rest of DCDL lengths by adapting N to the number of bits required to address a particular length. This algorithm has been implemented at the controller as a synchronous FSM, which updates the total DCDL delay following the aforementioned sequence. The benefits resulting from the algorithm action can be quantified from simulation, as it will be shown with Fig. 7.

TABLE V
ALGORITHM TO UPDATE THE FINE CONTROL BITS, LONGEST DCDL

Binary counter					Ordering code					Mapping ordering code-binary counter
b_4	b_3	b_2	b_1	b_0	o_4	o_3	o_2	o_1	o_0	
0	0	0	0	0	0	1	0	1	0	Odd stages: $o_{N-1-2i} = b_{2i}$ $i \in \left[0, \left[\frac{N-1}{2}\right]\right]$
0	0	0	0	1	1	1	0	1	0	
0	0	0	1	0	0	0	0	1	0	
0	0	0	1	1	1	0	0	1	0	
0	0	1	0	0	0	1	1	1	0	
0	0	1	0	1	1	1	1	1	0	
0	0	1	1	0	0	0	1	1	0	
0	0	1	1	1	1	0	1	1	0	
0	1	0	0	0	0	1	0	0	0	Even stages: $o_{N-1-2i-1} = 1-b_{2i+1}$ $i \in \left[0, \left[\frac{N-2}{2}\right]\right]$
0	1	0	0	1	1	1	0	0	0	
0	1	0	1	0	0	0	0	0	0	
0	1	0	1	1	1	0	0	0	0	
0	1	1	0	0	0	1	1	0	0	
0	1	1	0	1	1	1	1	0	0	
0	1	1	1	0	0	0	1	0	0	
0	1	1	1	1	1	0	1	0	0	
1	0	0	0	0	0	1	0	1	1	$\left[\frac{N-1}{2}\right]$ and $\left[\frac{N-2}{2}\right]$ stand for the integer part of these magnitudes
1	0	0	0	1	1	1	0	1	1	
1	0	0	1	0	0	0	0	1	1	
1	0	0	1	1	1	0	0	1	1	
1	0	1	0	0	0	1	1	1	1	
1	0	1	0	1	1	1	1	1	1	
1	0	1	1	0	0	0	1	1	1	
1	0	1	1	1	1	0	1	1	1	
1	1	0	0	0	0	1	0	0	1	
1	1	0	0	1	1	1	0	0	1	
1	1	0	1	0	0	0	0	0	1	
1	1	0	1	1	1	0	0	0	1	
1	1	1	0	0	0	1	1	0	1	
1	1	1	0	1	1	1	1	0	1	
1	1	1	1	0	0	0	1	0	1	
1	1	1	1	1	1	0	1	0	1	

VII. TIME AND POWER PERFORMANCE OF THE DLL

The dDLL performance for a DCDL of 32 stages, 40 MHz master clock and ordering option B is presented. Three PVT corners are considered: slow (125 °C, 1.08 V, SS), typical (25 °C, 1.2 V, TT) and fast (-40 °C, 1.32 V, FF). The following results have been obtained with a digital simulation of the post-layout netlist of the dDLL, flattened (taking into account the load effects from the interconnection of the different blocks), back-annotated (using the actual propagation delays of all cells and interconnects), with all timing checks enabled (including the setup-and-hold window limitation in the PD). Different values of standard deviation of the jitter superimposed to *ckin_up*, σ_j, are considered.

A Value Change Dump (VCD) file has been generated from these simulations, containing information on the switching activity of all nets in the circuit [12]. This file has been used to perform a static power analysis with Cadence® Voltus™ [13].

A. Time performance

The ADB LSB and the range of adjustment of the DCDL delay are reported in Table VI. The latencies can be updated in steps finer than the TDC time bin (20 ps), and the master clock period (25 ns) can be accommodated in the range of available delays in all corners. The number of master clock cycles required to lock from the time when an asynchronous reset is applied is listed for different σ_j values. The time required to lock 1) increases when σ_j is comparable to the ADB LSB, because there is a more significant ringing in *up_or_downn_aux*, and thus the counter of the PD filter is reset more often (more cycles need to be processed to generate a pulse at *clk_PD_ready*); and 2) depends on the corner according to the delay sweep performed by the controller: in the fast corner, the sweep relies mainly on the coarse control bits, while in the slow corner the controller sweeps mainly the fine control bits, which is a slower operation.

TABLE VI
TIME PERFORMANCE OF THE CDN

Corner	ADB LSB (ps)	Min. delay line (ns)	Max. delay line (ns)	Number of clock cycles required to lock for various values of the standard deviation of jitter, σ_j (ps)			
				1	2	3	4
Fast	4	11.14	26.24	6481	8389	10297	5512
Typ	5	16.21	41.04	11386	15113	17697	7105
Slow	7	24.64	67.57	14088	14182	14712	15144

The absolute value of the DCDL INL is represented for the different corners in Fig. 7, for σ_j = 3 ps. This result takes into account the non-idealities in the implementation of the dDLL (ADB layout imbalances, load effects in the interface PD-DCDL, etc.) and the divergence in the fine control bit values along the line. 3 ps is the largest standard deviation for which the time error target defined in equation (3) is met in all corners: the peak of the INL absolute value is at most 11 ps, which leaves a room of 9 ps for jitter and other non-idealities.

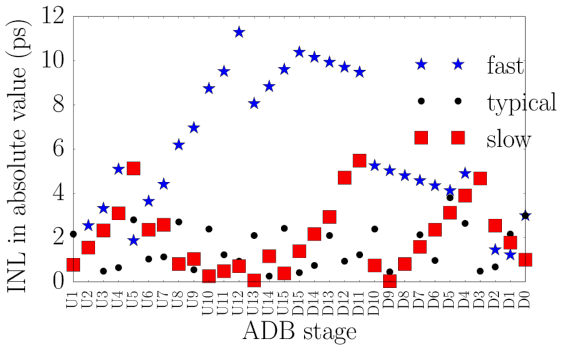


Fig. 7. Total INL (in absolute value) of the DCDL when the dDLL is in lock (back-annotated simulation), for ordering option B and σ_j = 3 ps.

B. Time resolution of the phase detector

The time resolution of the PD is reported as the start and end of the sensitivity window (point *S* and *E* in Fig. 5), both before (*S before*, *E before*) and after the digital filter (*S after*, *E after*), to evaluate its impact on the time performance of the PD.

In Fig. 8, these variables are depicted a function of σ_j for the tree PVT corners considered. The horizontal, unbroken lines at the center represent the ideal start (*S ideal*) and end (*E ideal*) of the sensitivity window (- ADB LSB and + ADB LSB, respectively). The tilted lines in the top half of the image are the linear fit of *E before*, while the tilted lines in the bottom

half of the image are the linear fit of *S before*. *E before* and *S before* are shown with square, star and dot markers, the trend of which is illustrated with the linear fit. When σ_j = 0 ps, the resolution window before the filter is dominated by the setup-and-hold window of the FFs that sample the time difference between the input and output clocks of the DCDL. As σ_j increases, the resolution window before the filter is widened with a slope close to $3\sigma_j$ (as it was introduced in section V.A, this is the largest expected time deviation caused by jitter). Due to the non-idealities of the PD and the jitter superimposed to the input clock, the resolution window before the filter clearly drifts apart from ± ADB LSB.

The tilted lines closer to the center of the figure are the linear fit of *E after*. *S after* is not available from the performed simulations; given the symmetry between *E before* and *S before*, *S after* could be extrapolated as *–E after*. *E after* can be approximated as *E before*/4, where the reduction factor stands for the square root of the digital filter window, W = 16. This is the smallest depth that yields the required sensitivity window after the filter. With this configuration, the digital filter provides a 4-fold enhancement in the resolution with respect to the sensitivity window before the filter, which enables achieving the desired resolution of ± ADB LSB.

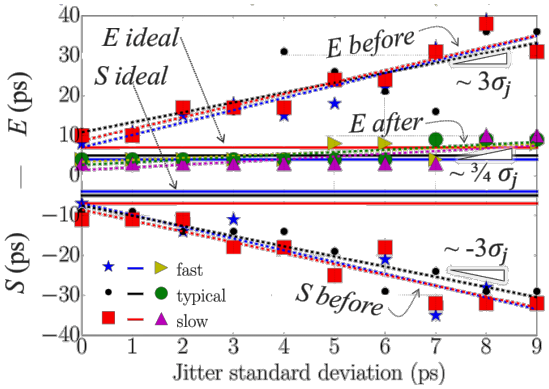


Fig. 8. Start (*S*) and end (*E*) of the sensitivity window of the phase detector as a function of the standard deviation of the jitter superimposed to *ckin_up*.

C. Power consumption

Table VII shows the total dDLL power consumption, including switching, leakage and internal components. The highest allowed σ_j (3 ps) is reported. This result corresponds to a simulation in which the dDLL is reset, let run until lock is achieved and remain in lock for a few thousand cycles (the same number of cycles is reported for the three corners).

Table VIII shows the estimated CDN power consumption at the chip level for the different chip areas and 376 µm pixel pitch. It is calculated from the values reported in Table VII, for the worst-case power consumption (fast corner) and scaling the consumption with the number of stages, number of dDLLs in the chip and master clock frequency (according to the guidelines provided in Table II) as:

$$P_{CDN} = k_{dDLL} * P_{dDLL} \quad (4)$$

Where P_{CDN} is the estimated total power consumption of the CDN at the chip level, k_{dDLL} is the number of dDLLs and P_{dDLL} is the estimated power consumption of 1 dDLL:

$$P_{dDLL} = k_f * (P_{ctrl} + P_{PD} + k_{ADB} * P_{ADB}) \quad (5)$$

- k_f: scale factor related to the master clock frequency, calculated as frequency in the particular scenario (MHz)/40 MHz, since the switching frequency is the dominant contribution (over 90% of the power reported in Table V, while leakage has a negligible contribution) and it scales linearly with frequency [14].
- P_{ctrl}, P_{PD}, P_{ADB}: controller, PD and ADB power consumption, respectively.
- k_{ADB}: 0.5 for the smallest chip area, since in this case the ADBs introduce half the delay and thus have a smaller coarse section; 1 for the rest of scenarios.

TABLE VII
POWER CONSUMPTION OF ONE DLL

Corner	Power PD (µW)	Power controller (µW)	Power ADB (µW)	Power dDLL (µW)
Fast	45.655	1.617	23.394	795.895
Typ	34.833	1.128	15.582	534.583
Slow	26.699	1.822	10.854	375.862

TABLE VIII
ESTIMATED POWER CONSUMPTION OF THE CDN AT THE CHIP LEVEL

Chip area (cm^2)	0.3 x 0.3	0.6 x 0.6	0.9 x 0.9	1.2 x 1.2	1.5 x 1.5	1.8 x 1.8	2.1 x 2.1	2.4 x 2.4
P_{CDN} (mW)	0.6	1.7	3.7	6.4	10.3	14.6	19.7	25.5

The CDN power consumption is mainly related to the chip area. For a smaller pixel pitch, the power consumption due to the dDLL is not expected to change, since the DCDL, PD and controller design will be the same.

VIII. DISCUSSION

A self-regulated CDN for the timestamp mechanism of the FastICpix chip has been presented. The selected architecture 1) can adapt to the chip area and pixel pitch, and 2) is robust to static and dynamic time errors, so that the total time error in the delivery of the master clock to all target TDCs is bound to the TDC time bin, 20 ps. The reported performance corresponds to the most challenging scenario: largest chip area; post-layout, back-annotated, flattened netlist of the dDLL. The CDN latencies can be adjusted in steps of 7 ps and the DCDL static time error is below 20 ps in all corners. Contrasted with the starting point of this work, the Timepix4 CDN, the presented solution has the potential to enhance the accuracy in the master clock distribution by an order of magnitude, while providing the versatility to tailor the readout chip to the application to optimize the signal collection.

APPENDIX A

As it was introduced in section VI.A, there are four possible update sequences for the 4-stage DCDL pursuing that the initial and final fine control bits values are evenly distributed along the line. These update sequences are labelled as A-D in Table IX. This table compiles the evolution of the fine control bit values in every stage as the sequence is applied; the sequence itself; the encoding of such a sequence with bits o_1, o_0 and the relation between these bits and the 2-bit binary counter used to implement it.

TABLE IX
ALGORITHM TO UPDATE THE FINE CONTROL BITS IN THE 4-STAGE DCDL

Order. options	Evolution of the fine control bits in ADB number…				Seq. update fine control bits	Order. code		Binary counter		Map. binary counter – order. code
	0	1	2	3		o_1	o_0	b_1	b_0	
A	1	0	0	0	ADB_0	0	0	0	0	$o_1 = b_0$ $o_0 = b_1$
	1	0	1	0	ADB_2	1	0	0	1	
	1	1	1	0	ADB_1	0	1	1	0	
	1	1	1	1	ADB_3	1	1	1	1	
B	0	1	0	0	ADB_1	0	1	0	0	$o_1 = b_0$ $o_0 = 1\text{-}b_1$
	0	1	0	1	ADB_3	1	1	0	1	
	1	1	0	1	ADB_0	0	0	1	0	
	1	1	1	1	ADB_2	1	0	1	1	
C	0	0	1	0	ADB_2	1	0	0	0	$o_1 = 1\text{-}b_0$ $o_0 = b_1$
	1	0	1	0	ADB_0	0	0	0	1	
	1	0	1	1	ADB_3	1	1	1	0	
	1	1	1	1	ADB_1	0	1	1	1	
D	0	0	0	1	ADB_3	1	1	0	0	$o_1 = 1\text{-}b_0$ $o_0 = 1\text{-}b_1$
	0	1	0	1	ADB_1	0	1	0	1	
	0	1	1	1	ADB_2	1	0	1	0	
	1	1	1	1	ADB_0	0	0	1	1	

As it was indicated in section VI.B, the mini-matrix can be expanded to arbitrary DCDL lengths or, in other words, the 2-bit ordering code can be expanded to an up-to-5 bits ordering code by relating the ordering code to the binary counter, which is shown in Table X for the different ordering options. The resulting sequence in which the stages are updated is also shown, with the stages named according to the ADB nomenclature introduced in Fig. 1.

TABLE X
ORDERING CODES AND SEQUENCES FOR ALL ORDERING OPTIONS

Ordering options	Mapping binary counter – ordering	Sequence to update fine control bits
A	$o_4 = b_0$ $o_3 = b_1$ $o_2 = b_2$ $o_1 = b_3$ $o_0 = b_4$	U0 → D15 → U8 → D7 → U4 → D11 → U12 → D3 → U2 → D13 → U10 → D5 → U6 → D9 → U14 → D1 → U1 → D14 → U9 → D6 → U5 → D10 → U13 → D2 → U3 → D12 → U11 → D4 → U7 → D8 → U15 → D0
B	$o_4 = b_0$ $o_3 = 1\text{-}b_1$ $o_2 = b_2$ $o_1 = 1\text{-}b_3$ $o_0 = b_4$	U10 → D5 → U2 → D13 → U14 → D1 → U6 → D9 → U8 → D7 → U0 → D15 → U12 → D3 → U4 → D11 → U11 → D4 → U3 → D12 → U15 → D0 → U7 → D8 → U9 → D6 → U1 → D14 → U13 → D2 → U5 → D10
C	$o_4 = 1\text{-}b_0$ $o_3 = b_1$ $o_2 = 1\text{-}b_2$ $o_1 = b_3$ $o_0 = b_4$	U5 → D10 → U13 → D2 → U1 → D14 → U9 → D6 → U7 → D8 → U15 → D0 → U3 → D12 → U11 → D4 → U4 → D11 → U12 → D3 → U0 → D15 → U8 → D7 → U6 → D9 → U14 → D1 → U2 → D13 → U10 → D5
D	$o_4 = 1\text{-}b_0$ $o_3 = 1\text{-}b_1$ $o_2 = 1\text{-}b_2$ $o_1 = 1\text{-}b_3$ $o_0 = b_4$	U15 → D0 → U7 → D8 → U11 → D4 → U3 → D12 → U13 → D2 → U5 → D10 → U9 → D6 → U1 → D14 → U14 → D1 → U6 → D9 → U10 → D5 → U2 → D13 → U12 → D3 → U4 → D11 → U8 → D7 → U0 → D15

In table XI, the mapping between the ordering code and the binary counter presented in Table X is expressed in a generic fashion for all the DCDL lengths to which the algorithm will be applied. *N* represents the number of bits required to address a certain DCDL length; *i* represents the bit position in the ordering code or the binary counter; $\left[\frac{N-1}{2}\right]$ and $\left[\frac{N-2}{2}\right]$ represent the integer part of these magnitudes.

TABLE XI
ORDERING CODES FOR ALL ORDERING OPTIONS AND GENERIC DCDL LENGTH

Ordering options	Mapping binary counter – ordering
A	$o_{N-1-2i} = b_{2i}$, $N \in [3,5]$, $i \in \left[0, \left[\frac{N-1}{2}\right]\right]$ $o_{N-1-2i-1} = b_{2i+1}$, $N \in [3,5]$, $i \in \left[0, \left[\frac{N-2}{2}\right]\right]$
B	$o_{N-1-2i} = b_{2i}$, $N \in [3,5]$, $i \in \left[0, \left[\frac{N-1}{2}\right]\right]$ $o_{N-1-2i-1} = 1 - b_{2i+1}$, $N \in [3,5]$, $i \in \left[0, \left[\frac{N-2}{2}\right]\right]$
C	$o_{N-1-2i} = 1 - b_{2i}$, $N \in [3,5]$, $i \in \left[0, \left[\frac{N-1}{2}\right]\right)$ $o_0 = b_{N-1}$ $o_{N-1-2i-1} = b_{2i+1}$, $N \in [3,5]$, $i \in \left[0, \left[\frac{N-2}{2}\right]\right]$
D	$o_{N-1-2i} = 1 - b_{2i}$, $N \in [3,5]$, $i \in \left[0, \left[\frac{N-1}{2}\right]\right)$ $o_0 = b_{N-1}$ $o_{N-1-2i-1} = 1 - b_{2i+1}$, $N \in [3,5]$, $i \in \left[0, \left[\frac{N-2}{2}\right]\right]$

The evolution of the fine control bits of every stage as the update sequence is applied (for the simplest scenario, when these bits can take the value 0 and 1) is shown in Table XII to Table XV, for the four ordering options. The ideal scenario, in which the alternance in the update is maximized, is located in the middle of the update range.

TABLE XII
EVOLUTION OF THE FINE CONTROL BITS AS THE UPDATE SEQUENCE IS APPLIED, ORDERING OPTION A

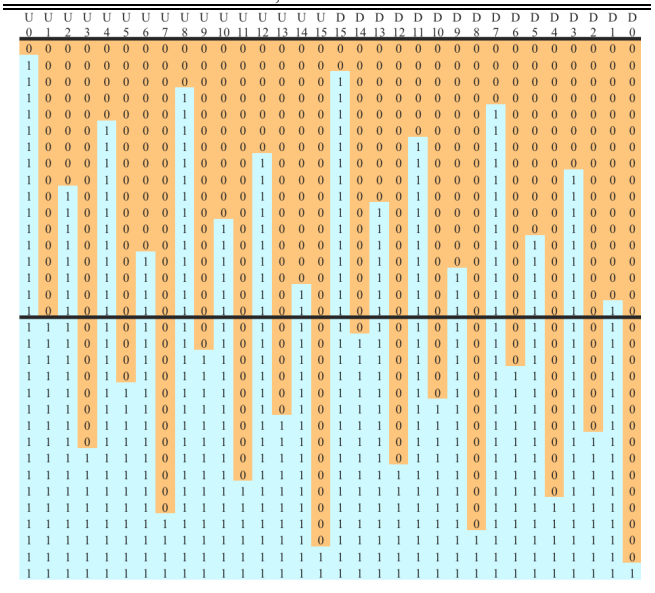


TABLE XIII
EVOLUTION OF THE FINE CONTROL BITS AS THE UPDATE SEQUENCE IS APPLIED, ORDERING OPTION B

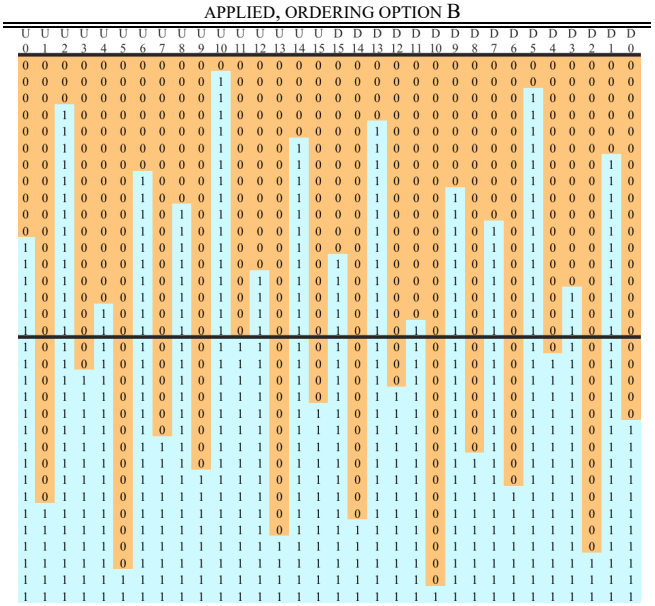


TABLE XIV
EVOLUTION OF THE FINE CONTROL BITS AS THE UPDATE SEQUENCE IS APPLIED, ORDERING OPTION C

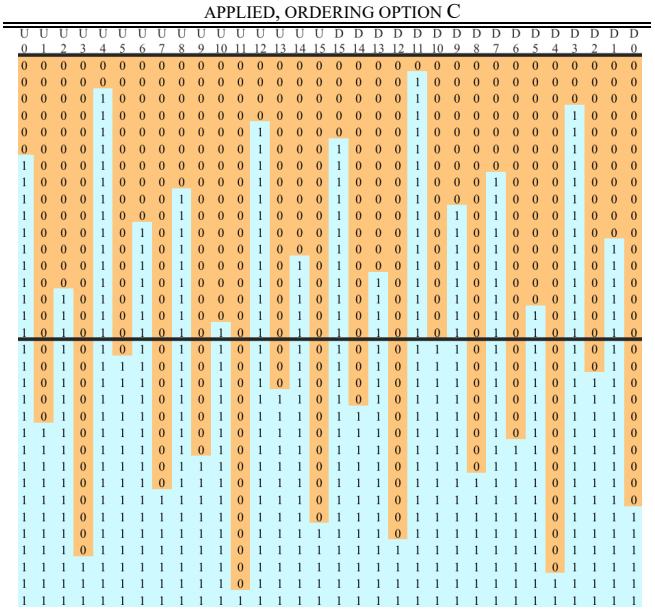
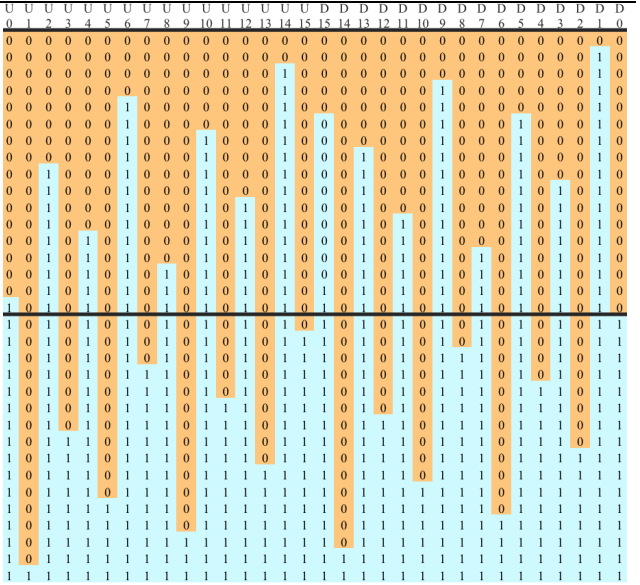


TABLE XV
EVOLUTION OF THE FINE CONTROL BITS AS THE UPDATE SEQUENCE IS APPLIED, ORDERING OPTION D



APPENDIX B

Four flavors of dDLL with 32 DCDL stages and master clock frequency of 40 MHz have been implemented, so as to evaluate the timing performance of the different ordering options (A-D). The INL obtained for these ordering options and different σ_j is shown in Fig. 9, for the same simulation conditions indicated in section VII.

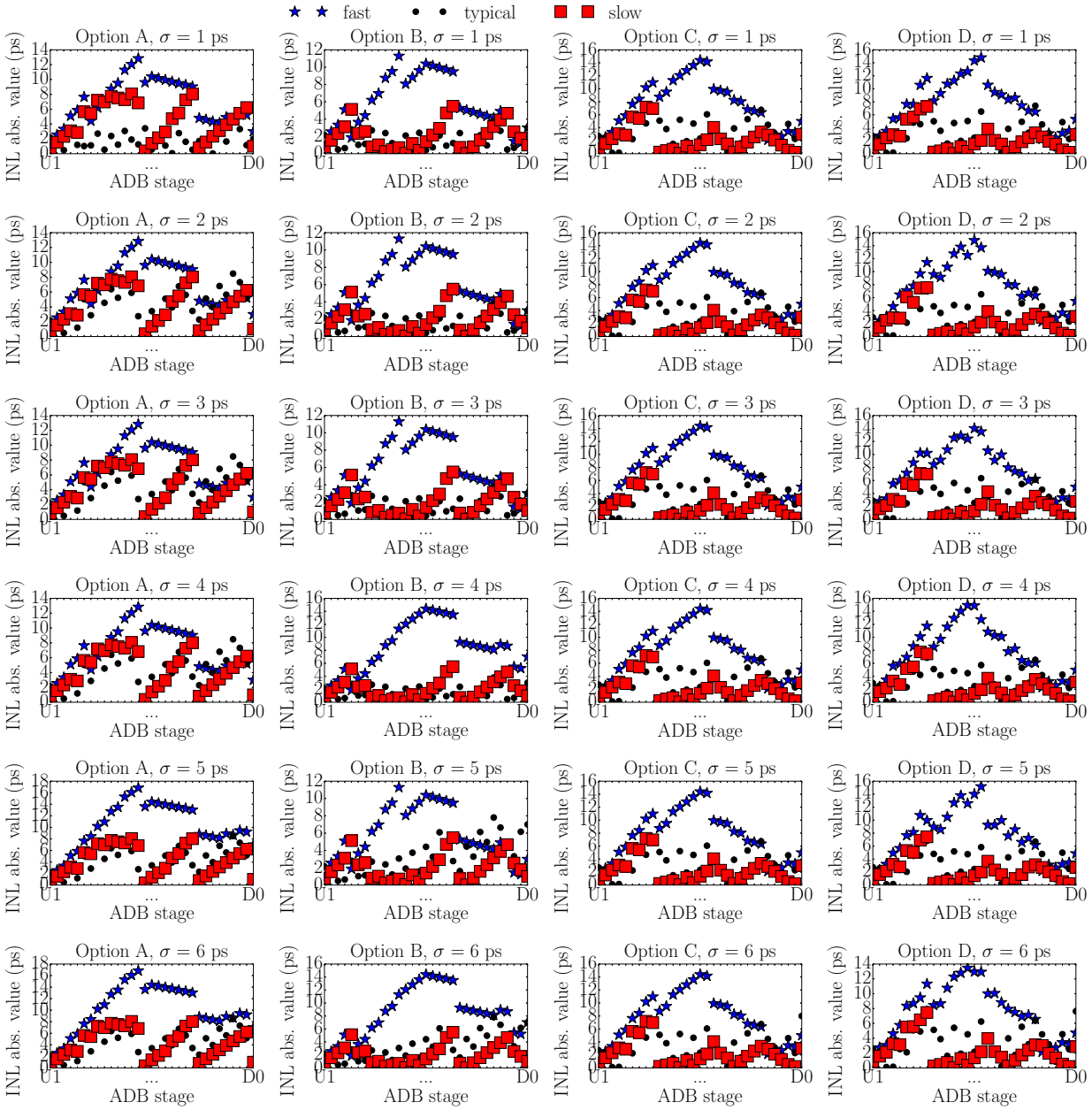


Fig. 9. Total INL (in absolute value) of the DCDL when the dDLL is in lock, for the four ordering options and different values of σ_j.

APPENDIX C

Table XVI expands Table I to clarify the criteria chosen to benchmark the different CDN configuration alternatives introduced in section II, as well as to understand the performance reported for each of the alternatives.

Four alternatives are benchmarked:

- The Timepix4 CDN [6], in which the branches consist of dDLLs that span across half the chip height.
- A solution based on mutually coupled oscillators [4]. The results shown here correspond to a matrix of 8x8 oscillators interconnected with a 600 Ω coupling resistance.
- A solution based on a mesh, in which the nodes implement a local de-skew action based on a PD and a compensator or adjustable delay [5].
- The CDN for the Alpha 21264 microprocessor, which features hierarchical grid levels [15].

The metrics used to perform the benchmark are the following:

A. CDN power consumption

These papers report the total power consumption of the network expressed for the clock frequency of operation, which differs amongst the considered solutions. Since the dynamic or switching component is usually dominant and it scales linearly with frequency [14], the reported power is scaled to 40 MHz so as to compare all options in the scenario of interest for this work. The values are further normalized to the chip area and expressed in mW/cm^2 for a more meaningful comparison. The power consumption values used to perform the benchmark are listed in the "CDN power consumption scaled to 40 MHz (mW/cm^2)" column. It can be seen that the microprocessor approach is the most power efficient, while the solution based on coupled oscillators is the most power hungry, which can be a concern for the largest areas envisaged.

B. CDN area

The area overhead associated to the network components is expressed as a percent of the total die area in the "CDN area (% w.r.t. chip area)" column. For [6], the CDN area includes the area of all PDs, controllers and ADBs. For [4] , the oscillator's area is considered, which stands for about ¼ of the TDC area. The remaining references do not provide the area associated to the CDN components, but the following extrapolation can be applied to relate [5] to the present work: in the selected configuration, the worst ratio between number of ADBs and PDs (i.e. the situation in which more PDs are required) is 1 PD per 8 ADBs, and it occurs for the smallest chip area reported in Table II. In [5], 28 PDs are used for 16 compensators (adjustable delays) or, alternatively, 10 PDs would be required for 8 compensators, hence requiring a significant component overhead compared to the selected configuration.

C. Largest static time error in the network or worst skew

The largest skew achieved by the different solutions is listed under the "Largest skew (ps)" column. In [6], it corresponds to a distance comparable to half the chip height, which is the area across which each dDLL spans. In the rest of cases, the worst skew occurs for sinks separated by the full chip height.

The dDLL solution, which is selected for this work, presents a skew comparable to the other alternatives, if not better, for a similar physical separation of the sinks. Yet it must be mentioned that [4] reports a reduction of $10 \cdot \log_{10}$(number of oscillators) in the phase noise or jitter, while the rest of configurations do not reduce the jitter present in the clock delivered to the sinks.

The excellent skew reported in [5] cannot be directly compared to the other results, due to the lack of area information.

On top of a low skew, the dDLL solution offers a major advantage, which is key for a pixel detector: it can guarantee a stable value (with a bounded static time error) of the latency from the clock source to the sinks across PVT variations. All solutions can guarantee the relative latency, i.e. a low skew, between the sinks, but only a dDLL-based solution can offer a

TABLE XVI
BENCHMARK OF SEVERAL CDN CONFIGURATION ALTERNATIVES

CDN configuration	Process node (nm)	Clock frequency	CDN power consumption (mW/cm^2)	CDN power consumption scaled to 40 MHz (mW/cm^2)	Chip area	CDN area (% w.r.t. chip area)	Largest skew (ps)	Ease of scalability with chip area and pixel pitch
dDLLs [6]	65	40 MHz	25	25	7 cm^2	2	100	Design multiple dDLL flavors, change the complexity of the local clock trees
Mutually Coupled Oscillators (8x8 matrix) [4]	65	500 MHz	$1.9 \cdot 10^3$	152	1.69 mm^2	0.5 (area of oscillators)	150 (600 Ω coupling resistance)	Interconnecting more oscillators, no redesign required
Local deskewing [5]	130	GHz	-	-	-	56 times more PDs than [16]	13	Interconnecting more PDs and compensators
Grid [15]	350	600 MHz	$9.3 \cdot 10^3$ (CDN stands for ~40% of total consumption)	0.62	3.1 cm^2	-	75	Potentially (at least partially) tool-automated

stable propagation delay from the clock source to the sinks regardless the corner. Having a low skew between sinks is translated to a low time error amongst the measurements of various TDCs for a particular corner; while ensuring the propagation latency across the corners is translated to a low time error on the measurement provided by a particular TDC when PVT variations occur.

D. Ease of scalability with the chip area and pixel pitch

The dDLL configuration might be the most complex to scale with the chip area and pixel pitch. In the rest of scenarios, the network can be expanded by adding more nodes (more oscillators in [4], more PDs and compensators in [5], more buffers in [15]). To scale a solution based on dDLLs, however, different flavors are required to adapt to different areas, which means some of the components ought to be redesigned; and the local clock tree that starts at the output of each ADB has to increase in complexity to adapt to smaller pixel pitch values.

Despite a larger complexity to scale the network with the chip area and pixel pitch, a solution based on dDLLs is preferred for this work, thanks to offering a suitable trade-off between the achievable skew; a low power consumption and area overhead associated to the network components; and thanks to the advantage of providing a stable latency from the clock source to the TDCs across PVT variations.
The Timepix4 CDN, in which the branches are composed of dDLLs, is considered as the starting point of this work.

REFERENCES

[1] J. H. Jungmann and R. M. A. Heeren, "Detection systems for mass spectrometry imaging: A perspective on novel developments with a focus on active pixel detectors," *Rapid Commun. Mass Spectrom.*, vol. 27, no. 1, pp. 1–23, 2013, doi: 10.1002/rcm.6418.

[2] E. Berg and S. R. Cherry, "Innovations in instrumentation for positron emission tomography," in *Seminars in nuclear medicine*, 2018, vol. 48, no. 4, pp. 311–331.

[3] P. Lecoq, "Pushing the limits in Time-Of-Flight PET imaging," *IEEE Trans. Radiat. Plasma Med. Sci.*, vol. 1, no. 6, pp. 473–485, Nov. 2017, doi: 10.1109/TRPMS.2017.2756674.

[4] "The 10 ps challenge: A step towards reconstruction-less TOF-PET." [Online]. Available: https://the10ps-challenge.org.

[5] B. Schmidt, "The High-Luminosity upgrade of the LHC: Physics and Technology Challenges for the Accelerator and the Experiments," *J. Phys. Conf. Ser.*, vol. 706, no. Section 2, 2016, doi: 10.1088/1742-6596/706/2/022002.

[6] T. Xanthopoulos, *Clocking in Modern VLSI Systems*. Boston, MA: Springer US, 2009.

[7] D. Gascón, "Integrated signal processing for a new generation of active hybrid single photon sensors with ps time resolution (FastICpix)," 2019. [Online]. Available: https://attract-eu.com/selected-projects/integrated-signal-processing-for-a-new-generation-of-active-hybrid-single-photon-sensors-with-ps-time-resolution-fasticpix/.

[8] N. Egidos *et al.*, "Self-regulated Clock Distribution Network for a fast-timing active hybrid single photon detector," in *Proceedings of the IEEE Nuclear Science Symposium and Medical Imaging Conference (NSS/MIC) (submitted)*, 2020.

[9] E. G. Friedman, "Clock Distribution Networks in Synchronous Digital Integrated Circuits," *Proc. IEEE*, vol. 89, no. 5, pp. 665–692, 2001, doi: 10.1109/5.929649.

[10] V. Oklobdzjja, V. Stojanovic, and D. Markovic, *Digital System Clocking*. Hoboken, New Jersey: John Wiley & Sons, Inc., 2003.

[11] A. R. Ximenes, P. Padmanabhan, and E. Charbon, "Mutually coupled time-to-digital converters (TDCs) for direct time-of-flight (dTOF) image sensors," *Sensors*, vol. 18, no. 10, p. 3413, 2018, doi: 10.3390/s18103413.

[12] C. E. Dike, N. A. Kurd, P. Patra, and J. Barkatullah, "A Design for Digital, Dynamic Clock Deskew," *IEEE Symp. VLSI Circuits, Dig. Tech. Pap.*, vol. 97124, pp. 21–24, 2003.

[13] Y.-B. Kim, "Signal de-skewing using programmable dual Delay-Locked Loop," US Patent 5880612, 1999.

[14] R. L. Aguiar and D. M. Santos, "Wide-area clock distribution using controlled delay lines," *Electron. Circuits Syst. 1998 IEEE Int. Conf.*, 1998, doi: 10.1109/ICECS.1998.814825.

[15] R. J. Yang and S. I. Liu, "A 2.5 GHz all-digital delay-locked loop in 0.13 μm CMOS technology," *IEEE J. Solid-State Circuits*, vol. 42, no. 11, pp. 2338–2347, 2007, doi: 10.1109/JSSC.2007.906183.

[16] X. Llopart *et al.*, "Study of low power front-ends for hybrid pixel detectors with sub-ns time tagging," *J. Instrum.*, vol. 14, no. 1, 2019, doi: 10.1088/1748-0221/14/01/C01024.

[17] S. Henzler, *Time-to-digital converters*. Springer Science & Business Media, 2010.

[18] G. Upton and I. Cook, *A Dictionary of Statistics*, 2nd ed. Oxford University Press, 2008.

[19] J. Choi, Y. Moon, K. Lee, D. Yeong, and M. Kim, "An All-Analog Multiphase Delay-Locked Loop Using a Replica Delay Line for Wide-Range Operation and Low-Jitter Performance," *IEEE J. Solid State Circuits*, vol. 35, no. 3, pp. 377–384, 2000.

[20] S. Tayyeb Ghasemi and A. Baradaranrezaeii, "A Novel High Speed, Low Power, and Symmetrical Phase Frequency Detector with Zero Blind Zone and π Phase Difference Detection Ability," *Circuits, Syst. Signal Process.*, vol. 39, no. 6, pp. 2880–2899, 2020, doi: 10.1007/s00034-019-01312-w.

[21] H. Lad Kirankumar, S. Rekha, and T. Laxminidhi, "A Dead-Zone-Free Zero Blind-Zone High-Speed Phase Frequency Detector for Charge-Pump PLL," *Circuits, Syst. Signal Process.*, vol. 39, no. 8, pp. 3819–3832, 2020, doi: 10.1007/s00034-020-01366-1.

[22] D. Chen *et al.*, "A comprehensive approach to modeling, characterizing and optimizing for metastability in FPGAs," in *Proceedings of the 18th annual ACM/SIGDA international symposium on Field programmable gate arrays*, 2010, pp. 167–176.

[23] S. L. Harris and D. M. Harris, "Sequential Logic Design," in *Digital Design and Computer Architecture*, 2nd ed., Elsevier, Inc., 2016, pp. 108–171.

[24] S. K. Nithin, G. Shanmugam, and S. Chandrasekar, "Dynamic voltage (IR) drop analysis and design closure: Issues and challenges," *Proc. 11th Int. Symp. Qual. Electron. Des. ISQED 2010*, pp. 611–617, 2010, doi: 10.1109/ISQED.2010.5450515.

[25] Cadence, "Voltus IC Power Integrity Solution." [Online]. Available: https://www.cadence.com/en_US/home/tools/digital-design-and-signoff/silicon-signoff/voltus-ic-power-integrity-solution.html.

[26] J. Rabaey, *Digital integrated circuits: a design perspective*, 2nd ed. Prentice-Hall, Inc., 2002.

[27] P. E. Gronowski, W. J. Bowhill, R. P. Preston, M. K. Gowan, and R. L. Allmon, "High-performance microprocessor design," *High-Performance Syst. Des. Circuits Log.*, vol. 33, no. 5, pp. 395–404, 1998, doi: 10.1109/9780470544846.ch4.