Can DSP48A1 adders be used for high-resolution delay generation?

Scott Tancock, Ekin Arabul and Naim Dahnoun
School of Computer Science, Electronic Engineering and Engineering Mathematics, University of Bristol
Bristol, United Kingdom
scott.tancock@bristol.ac.uk, ea0534@bristol.ac.uk, naim.dahnoun@bristol.ac.uk

Shahid Mehmood
Quaid-i-Azam University
Islamabad, Pakistan
raoshahid1580@gmail.com

Abstract— **Time to digital conversion is an important task in many systems. It involves the conversion of time-based signals (as opposed to the amplitude-based signals in analog-to-digital conversion) into digital numbers so that a purely digital system may process them. This is widely used in rangefinders, all-digital phase-locked loops and quantum experiments. In order to obtain high-resolution time-to-digital conversion, the generation of small delays is necessary. This paper examines the viability of using the DSP48A1 blocks present on Xilinx FPGAs to generate these small delays, and ultimately concludes they are unsuitable in isolation due to the high differential non-linearity, but may be suitable as a semi-fine stage of a multi-stage TDC or when combined in an equivalent coding line.**

Keywords- Time-to-Digital Converters; Field-Programmable Gate Arrays; Xilinx; Digital Signal Processing; Delay Generation

I. INTRODUCTION (HEADING 1)

A. State of the Art

Time to digital conversion is a relatively obscure topic compared to the more widely known analog-to-digital conversion. As such, major developments in the field are still relatively frequent. Until recently, most Time-to-Digital Converter (TDC) designs were focused on a two-stage process involving a Time-to-Analog Converter (TAC) followed by an Analog-to-Digital Converter (ADC) [1]. These TAC-ADC systems obtained higher resolutions and ranges when transistors remained large and propagation delays remained long, but as process nodes became smaller, digital delay generation methods such as delay lines, Vernier delay lines and Vernier rings have surpassed TAC-ADCs in both resolution and size. In addition, recent developments such as pulse-shrinking TDCs [1], Local-Passive Interpolation (LPI) TDCs [1] and Stochastic TDCs [1] have further increased the resolution.

Fundamental to all these digital delay generation methods is the delay element. The specifications of the delay element define the resolution of the TDC, and so improvements to the delay generation method is an important factor in increasing size of the delay element is the resolution (LSB) of the delay line, whereas more advanced methods such as the Vernier delay lines use the difference between parallel elements or some other property. In this paper we have investigated a new method of delay generation and evaluated its viability in comparison with other known methods.

Traditional methods of delay generation can be split into three categories: digital delays, analog delays and pulse modification. Digital delays utilise the inherent delay present in a digital component, such as an inverter (CMOS) or transmission gate (Pass Transistor Logic), to create a time difference. Examples of this include the delay line and Vernier delay line, which use the delay of an element and the difference between element delays respectively. As the difference between element delays can be tuned to a much higher precision, the Vernier delay line is generally superior in resolution. However, it suffers in conversion speed and is not applicable to all system architectures, such as FPGAs.

Analog delays are formed by passing the signal through analog components to delay it, rather than digital ones. These methods include inductive delays and switched capacitor arrays, of which the switched capacitor method is more popular as it can be easily integrated into an ASIC. Similarly to the Vernier delay lines, these methods are not suitable for some architectures, and hence belong exclusively in the field of ASIC TDCs. The use of switched capacitor arrays is most commonly seen in cyclic time-domain successive approximation (CTDSA) TDCs [1].

Pulse modification techniques involve encoding the time as the on time of a pulse, and then successively modifying the pulse until it reaches some detectable state. Most commonly, this is implemented as a looped pulse shrinker, which shortens the pulse by a known amount until it is no longer detectable [1]. While the linearity of this method is excellent, and careful selection of transistor sizes allows for excellent resolution, the conversion rate is relatively low for this method, and it is most often unsuitable for FPGAs.

When looking specifically at FPGA-compatible methods, the most common method utilised appears to be delay lines, as there is almost always some form of delay element available on the FPGA. Sometimes, an FPGA will have a suitable

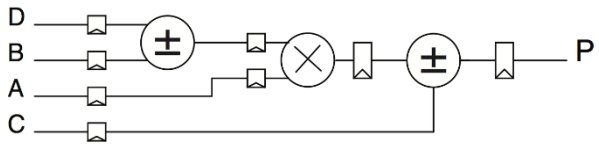

Figure 1: Simplified block diagram of the DSP48A1 [2].

architecture for implementing a Vernier delay line [1], and sometimes an element can be found which is suitable for pulse shrinking, but this is dependent on the particular FPGA architecture. Delay lines can be generated in various ways, such as with the look-up tables (LUTs) configured as buffers, but the highest resolution found so far appears to be the carry chain [1] made available for addition operations.

As FPGAs are often required to perform addition and subtraction, most vendors implement carry logic with a fast interconnect between logic blocks, as this is commonly the critical path, and so fast carry logic may drastically increase the speed of the design. The carry logic also generally propagates perpendicularly to the rest of the logic interconnects, as this allows parallel signals which are part of the same number to have similar routes through the FPGA, thereby increasing the probability of meeting timing requirements. These carry chains are also ideal for implementing fast delay lines which output a priority code based on the delay between the start and stop signals.

Despite the importance of the generating high-resolution delays, there are seemingly no reports of the viability of the FPGA's built-in DSP48A1 block as a delay generator, despite the architecture suggesting that it is suitable.

B. DSP48A1 Architecture

Almost every DSP algorithm has MAC (Multiply Accumulate) operations. In the Spartan-3A architecture, the DSP48A slice is available for MAC operations and it has been extended into the DSP48A1 slice in the Spartan-6 series. In Spartan-6 FPGAs, DSP48A1 slices are organized as vertical columns along with some additional dedicated logic and routing [1]. One of the most important features is the ability to cascade a result from one DSP48A1 slice to the next without the use of general fabric routing. Each DSP48A1 slice contains an 18-bit input pre-adder followed by an 18 x 18-bit two's complement multiplier and a 48-bit sign-extended post-adder/subtracter/accumulator, a function that is widely used in digital signal processing. Figure 1 shows a simplified block diagram of the DSP48A1.

All these arithmetic operations are hard-wired fabricated into the silicon. Every arithmetic function has pipeline registers on its input and output. These registers are very close to the corresponding function logic and can be bypassed, so arithmetic functions (Combinational Logic) can directly connect to FPGA fabric. The performance of DSP48A1 is explained in [1].

C. Applications of efficient high-resolution delay generation

With the development of single photon applications, high resolution time measurement became important in many different aspects of engineering and science. One of the most popular applications of time-to-digital converters is time of flight measurements such as LiDAR. In a LiDAR setup, a TDC is used for precisely measuring the time difference between the START signal of the emission and STOP signals corresponding to its reflections. As the resolution of the TDC increases, the LiDAR's capability of measuring shorter distances improves. This is desirable in applications such as positron emission tomography(PET). A high resolution TDC could also be used for coincidence measurements in quantum physics such as in Bell test experiments. In quantum physics experiments, high-resolution time measurement techniques would be a part of gating the events of interest from the background noise. Apart from single photon applications, high resolution TDCs can also be found in digital phase locked loops as a part of the phase detector circuit.

D. Structure

The rest of the paper will be formatted as follows: first will be a section on the design of a system that uses DSP48A1 blocks as delay generators, which will detail how the DSP blocks can be used to generate 48 delays per block. Following that is a section describing how the DSP blocks were tested to determine their viability. After that is the results section, where the results of the tests are shown. The penultimate section discusses the results obtained and how they relate to the performance of a TDC using DSP41A1 blocks. The paper is ended with a conclusion which summarises the results of our experiments and suggests some possible uses for the DSP block delay generation.

II. SYSTEM DESIGN

A. Generating a delay line inside a DSP48A1

The first task was to attempt to generate a delay line inside a single DSP48A1 block. A delay line requires a single input that changes from 0 to 1 (or vice-versa), and multiple outputs which transition successively. Preferably, it would also have unregistered inputs and outputs from adjacent blocks. For this purpose, we chose the post-adder in the DSP block. The benefit of the post-adder is the carry in and carry out connections, which can be configured to be unregistered. Also, it contains the shortest path to the output pins of the DSP48A1. Finally, the P register can be configured to provide the output discrimination required, thereby reducing the logic utilisation.

To configure the DSP48A1 block, all register bypass muxes were used to bypass their associated registers. The Z input to the post-adder was set to use the C input port, which had all 48 bits set to 1, thereby causing the post-adder to carry any additional non-zero input through the internal carry chain. The X input was set to use the 0 input on all DSPs except the first, which used the D:A:B concatenated input where B[0] is connected to the trigger and all other bits are connected to 0. The

connection of B[0] to the trigger on the first DSP allows for the trigger to be input to the DSP carry chain, while the other DSPs use the carry input to trigger the later sections of the delay line.

B. Placement on the FPGA Fabric

The carry in and carry out are dedicated connections between the DSPs, which are placed column-wise. By using the device-native blocks in the chosen hardware description language, the synthesis tool is forced to use adjacent DSP blocks with minimal inter-DSP latency. As a result, the delay between DSP48A1 blocks is minimised, resulting in optimal differential linearity in the delay line.

C. Triggering Logic

In order to avoid reading out when there is no valid data in the delay line, some triggering logic was implemented. The trigger was passed through a synchroniser which normalised it to the clock edge, and then the current value of the trigger was compared to the value of the trigger on the previous clock cycle. If the trigger is high on the current clock cycle and was low on the previous clock cycle (rising edge triggered), then this implies a new trigger came in and so there is valid data in the delay line. This is then used to enable the priority encoder and read-out logic, which convert the priority code to binary and transmit the binary to a host computer respectively.

To do this, the trigger signal is delayed by multiple clock cycles so that it stays in-line with the valid conversion as it propagates through the synchronisers and priority encoder, before finally becoming the enable signal for the read-out logic, which is described in the Section II.D

D. Read-Out Logic

Read-out was performed using the parallel interface provided by the Opal Kelly SDK [1]. This interface allows 32-bit quantities to be transferred to a host PC using the USB present on Opal Kelly FPGA carrier boards. As the tags are 10-bit quantities and the coarse count is not required for code density testing, the upper 6 bits of each word were set to 0, while the 10-bit bin number was stored in the lower bits of the word. A 10-bit quantity was required as 20 DSP blocks were implemented (960 bins) in order to ensure no tags exceeded the end of the delay line.

The transmitted data was collected by software on the host PC in a comma-separated variable format, then imported into GNU Octave for analysis.

III. Test Methodology

High-resolution and PVT-stable delay lines and capture registers are required for TDC applications. As we use the DSP48A1's post-adder carry propagation logic (48 bits) and the P register as the capture register, we can expect the delay line's characteristics to be similar to that of a carry look-ahead adder (the architecture used in most adders). The carry lookahead logic is defined by Equation **Error! Reference source not found.** where C_i is ith carry output and it is produced by $G_i = A_i . B_i$ (carry generation) or $P_i = A_i \oplus B_i$ (carry propagation).

$$C_i = G_i + P_i . C_{i-1} \quad (1)$$

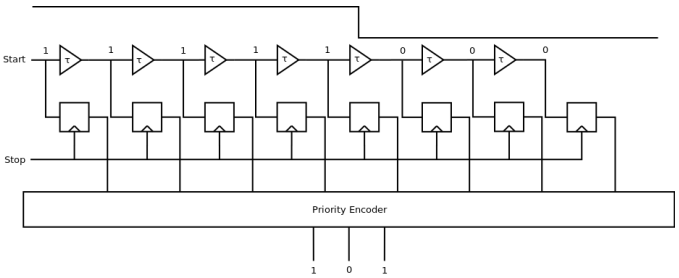

Figure 2: Generic model of a delay line.

We model the DSP delay line using the generic model shown in Figure 2. Each buffer here corresponds to the carry propagation logic between one bit of the adder and the previous bits of the adder (and in some cases may be negative where carry bypassing occurs). This allows us to use standard average delay and code density testing methods to determine the delay of each bit of the adder.

In the average delay tests, we determine the time taken for the start signal to reach the final time bin (end of the last 'buffer') as shown in Figure 2. This implies an output code of 0xFFFF...FFFF and the MSB being '1'. In the code density tests, we determine the propagation delay of each 'buffer' by providing random signals in the time domain, uniformly distributed with respect to the stop signal, and creating a histogram of the priority encoder outputs. In the case that the stop signal is not equally routed (there are buffers between each bin on the stop line), the bin size will increase by the delay to the corresponding register, and decrease by the delay to the previous register.

A. Average Delay Testing

In the average delay test, the post-adder was set up to have all the bits on one of its inputs high, and all the bits on the other input low except the LSB, which was connected to the trigger signal. As we are adding one to the maximum representable number, all the bits will roll around to 0 through the carry propagation logic. To determine the average delay, we calculate the time difference between the LSB and MSB transitioning.

To implement this, we first connected two probes from a 1GHz oscilloscope to adjacent FPGA outputs (same IOB). These outputs were both connected to the LSB of a DSP block with the D:A:B input (through the Z-mux) being 0xFFFFFFFFFFFF to enable the carry chain and the C input (through the X-mux) being 0 except the LSB, which was the trigger generated by an on-chip oscillator. The probe delays were then adjusted to tear (align) the signals before re-routing one output to the MSB of the DSP block, which resulted in an almost identical path while including the full delay of the DSP. The nets were inspected post-implementation to ensure matched routing.

B. Code Density Testing

In addition to the average delay testing, code density testing is an important component in determining the viability of a delay generation method. Even if the average delay is exceptionally low, a TDC is unusable unless the non-linearity characteristics are decent. A good parallel to this is the linearity of an ADC.

Even if an ADC can be found to have 100 bits of resolution (well in excess of most modern ADC designs), this is useless if one bin covers half the range of the ADC while the other 1023 cover the other half. In such a case, the effective resolution is at most 2 bits. Such an ADC is useful, however, as putting another ADC in parallel but offset in the voltage domain could allow the two ADCs to efficiently cover the entire voltage range. On the other hand, this increases the area requirements. The same can be said for a bad TDC; as long as the code density is decent, corrections can be made to cover for its weakness using another TDC in parallel.

1) Linear Code Density Testing

Code density testing is one of many techniques to measure the linearity of the TDC. The code density test provides random, uniformly distributed signals to the TDC and stores the conversion results. When provided with a uniformly random signal (in the time domain), the number of times a particular code will be output by the TDC will be proportional to the size of the time bin corresponding to that code. A larger time bin (longer delay until the next storage element) will accumulate more 'hits' (number of times the corresponding bin code is output) whereas a smaller time bin will accumulate less hits. Once a suitable number of hits has been accumulated to obtain an accurate measure of the width of each bin, the size of time bin i, τ_i, will be calculated as in Equation (2)

$$\tau_i = \frac{H_i \times T_{clk}}{H_{total}} \quad (2)$$

$$T_i = \sum_{j=0}^{i} \frac{H_j \times T_{clk}}{H_{total}} \quad (3)$$

In Equation (2), H_i is the number of hits for bin i, H_{total} is the total number of hits for all bins, and T_{clk} is the period of the system clk (which acts as our stop signal, and so the period is the maximum possible time difference). Similarly, the cumulative form of this equation, T_i, can be used to determine the delay up to a certain bin (and is hence used for calibration).

2) Cyclic Code Density Testing

In this test, the post-adder P register is enabled and 2 independent oscillators are used to produce a random phase drift. The first oscillator is similar to the average delay test and is connected in the same way to generate the first carry. A 1kHz trigger signal with a different oscillator is connected to the P pipeline register to capture the carry propagation randomly. In the capture register, invalid codes (0x000000000000 and 0xFFFFFFFFFFFF) are removed and the rest of the codes are transferred to a PC via a serial interface.

3) Subdivided Code Density Testing

In order to eliminate the possibility of carry look-ahead logic eliminating some bins by forwarding the carry across 24 bins at once, we implemented a method which simultaneously tests both the lower and upper 24 bits without continuity between them. In this test, a 10MHz trigger signal was input into a DSP block at positions 0 and 24 through the C input and Z-mux, while the D:A:B input and X-mux was set to 0x7FFFFF7FFFFF so that the carries would stop at positions 23 and 47 respectively. The number of cases of each output (23 and 47) being high was recorded on a host PC via a serial connection.

Figure 3: Time difference between two output signals on a 1GHz oscilloscope.

C. Validation

To ensure the results we achieved were accurate, we cross-validated our results across two labs, designs and pieces of hardware. The first design used linear code density testing on a Spartan-6 LX150 FPGA packaged by Opal Kelly [1] on a custom signal breakout board at the University of Bristol. The other device, a Spartan-6, used a cyclic code density test at Quaid-i-Azam University. If our results are to be believed, we expect that the individual bin sizes will not be the same, but the patterns in bin sizes should match. For example, if there is a recurring pattern every 4 bins on one device, we would also expect to see a similar recurring pattern on the other device. If this does not occur, then this suggests that the external components in the design (test input signal, read-out logic, system clock etc.) are interfering with the measurement and hence the design can be fixed to remove the effect of these external influences.

IV. RESULTS

A. Average Delay Testing

Under testing, it was found that the delay between the two output signals was 800 ps (Figure 3) representing the entirety of the DSP delay. As there are 48 bins, this evaluates to a mean bin width of 16.7 ps. A similar test was performed, but measuring the time to output bit 23 rather than bit 47, which gave a relative delay of 400 ps, showing that the total delays in the first and second halves of the DSP are equal.

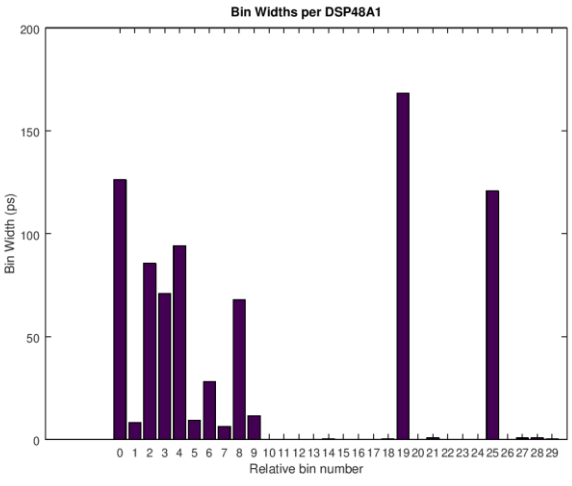

Figure 4: A histogram of a 384k-hit cyclic code density test.

B. *Code Density Testing*

Histograms from the code density tests conducted at the University of Bristol and Quaid-i-Azam University were collected, with the bin widths for a single DSP block shown in Figure 4 and the calibration graph (cumulative bin widths) for a large (960 bin = 20 DSP) delay line shown in Figure 5. The diagrams have been normalised to the number of hits and the clock period to obtain the delay times of each bin. As the delay is on the y axis, a large vertical jump (large bin width) is undesirable (low resolution), whereas a small, non-zero vertical jump is desirable (high resolution). No vertical ascension between bins implies a lost code, which can be ignored.

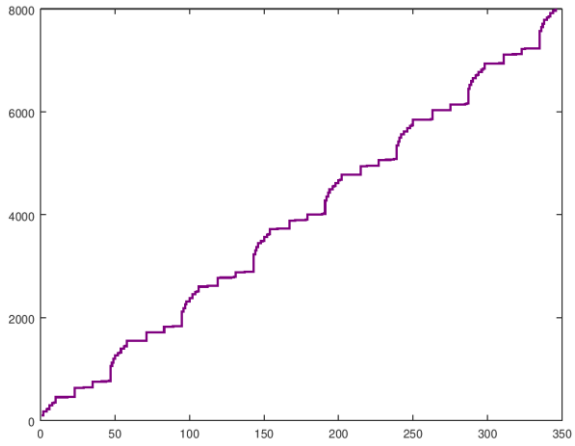

Figure 5: The calibration chart obtained from a histogram of a 384k-hit linear code density test. The x axis is the output code, and the y axis is the time difference between the start and stop signals.

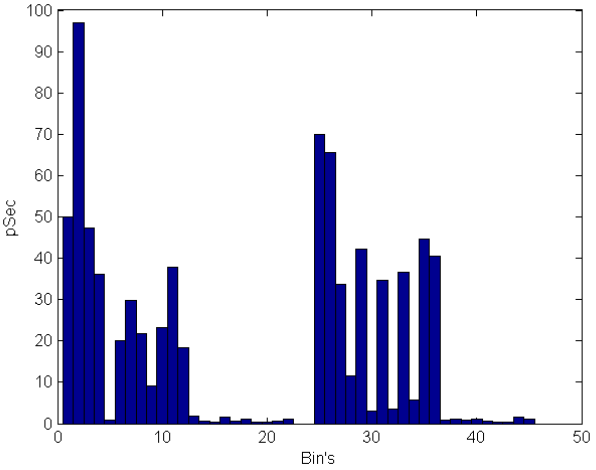

Figure 6: The histogram of the sub-divided test.

V. DISCUSSION

A. *Average Delay Testing*

The average delay testing showed that the signals that emerged from the output pins of the FPGA had a difference of 800 ps between them. This suggests that the average delay of the bins is 16.7 ps. This is a 20.5% improvement on the CARRY4 chain, presuming that all bins are in use (non-zero effective width). However, as bins that have a zero effective width do not contribute to the resolution of the system (they are missing bins), the more missing bins we discover in the code density test, the lower the effective resolution.

B. *Code Density Testing*

As can be seen from the code density tests in Figure 4 and Figure 5, a single DSP48A1 does not create effective delay lines. While the average bin size per DSP is small (16.7 ps from the average delay testing), the bin sizes are very widely skewed,

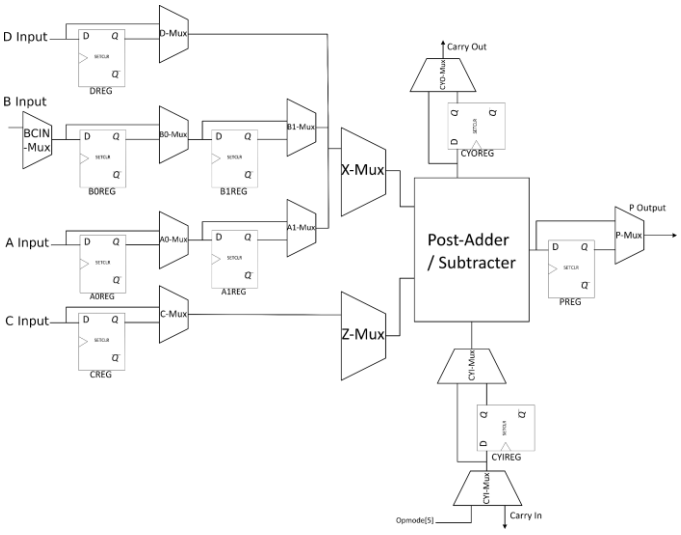

Figure 7: The internal architecture of the DSP48A1 block (utilised components only), based on [2].

with the vast majority of bins exhibiting an effective width of 0 ps (or so small that they could not be measured). Meanwhile, a minority of bins exhibited large delays in the range of 40-170 ps and the few bins at the boundary of a DSP block exhibited delays between 200 and 350 ps. As the missing bins (zero effective width) do not contribute to the resolution of the system, the actual resolution is 34.8 ps with a standard deviation of 51.3 ps.

This is a significantly lower resolution (larger time difference) than has been achieved by the carry chains ($(\mu, \sigma) = (34\ ps, 51\ ps)$ compared with $(21\ ps, 25\ ps)$). As a result, we can conclude that these devices are unsuitable for generating high-resolution delays (i.e acting as the final stage in a multi-stage TDC) in isolation. However, as they do generate a significant delay, they may be useful as a semi-fine stage in a multi-stage TDC, thereby reducing the required length of the fine delay line (and hence reducing the overall logic utilisation).

Alternatively, multiple DSP blocks could be utilised in a parallel staggered fashion, allowing delay lines to sub-divide each other in an equivalent coding line. As the resolution increases with the square root of the number of delay lines, 2.62 (i.e.~3) DSP48A1 delay lines would be needed to obtain the same resolution as a CARRY4 chain. The rest of this section will be dedicated to examining the cause of the observed delays.

First, we observe the large bins at multiples of 48 in the linear test. A comparatively large bin in a structure usually denotes some form of discontinuity in the logic structure, and in this case the discontinuity is the crossing between DSP48A1 blocks. Each DSP48A1 block provides 48 time bins, and then a carry out which is directly connected to the next DSP block. While this direct connection is fast compared to most FPGA routes, it is still significantly slower than the routes inside the DSP block as the signal must travel through three configuration multiplexers between the post-adders of each DSP block.

Second, we notice that the bins which are a multiple of 96 (even multiples of 48) are larger than the odd multiples of 48. Unlike the odd multiples, the even multiples of 48 not only need to propagate between two DSP48A1 blocks, but also across a clocking discontinuity. One in every four DSP interconnects is on a clock management tile (CMT) boundary, while another is bridging the CMT's dedicated clock routing area. These jumps cause the multiples of 96 to have larger bins than the odd multiples of 48.

Beyond this, we notice that bins which are an odd multiple of 24 are also relatively large. The available documentation from Xilinx gives no indication as to the cause of this, and so it can only be presumed that there is some discontinuity in the internal addition logic present in the DSP48A1. An example of this would be carry look-ahead logic, which in a 48-bit adder would forward the carry signal to multiples of 24.

Under the split test, we see that the presence of the bins after bin 24 becomes much more pronounced. This adds credibility to the suggestion that the missing bins are caused by carry look-ahead logic is bypassing bins to increase adding speed, but at the expense of delay line differential non-linearity.

VI. CONCLUSION

In conclusion, we have found that the post-adder in a DSP48A1 block can be used to generate delays with an average length of 34.8 ps and a standard deviation of 51.3 ps. This is a lower accuracy than the competing CARRY4 chain solutions, but this could be remedied by using multiple staggered DSP48A1 blocks in parallel. The larger delay generated by the DSP48A1 blocks was found to be suitable as a semi-fine delay generation as would be required in a multi-stage TDC. We discovered this using average delay calculation and code density testing.

In future work, we would like to place multiple DSP blocks in parallel to enable a high-resolution, linear delay line that does not use the CARRY4 chains. This could then be adopted as a technique for fitting extra delay lines into a multi-channel on-chip TDC where the number of channels is limited by the quantity of logic elements.

REFERENCES

[1] Opal Kelly, "Opal Kelly Front Panel SDK," Opal Kelly, 2014. [Online]. Available: https://www.opalkelly.com/products/frontpanel/. [Accessed 19 February 2018].

[2] Xilinx, "Spartan-6 FPGA DSP48A1 Slice User Guide (UG389)," 2014. [Online]. Available: https://www.xilinx.com/support/documentation/user_guides/ug389.pdf [Accessed 19 February 2018].

[3] Opal Kelly, "Opal Kelly XEM6010," Opal Kelly, 2014. [Online]. Available: https://www.opalkelly.com/products/xem6010/. [Accessed 19 February 2018].

[4] Xilinx, "Spartan-6 FPGA Data Sheet: DC and Switching Characteristics (DS162)," 2015. [Online]. Available: https://www.xilinx.com/support/documentation/data_sheets/ds162.pdf. [Accessed 19 February 2018].

[5] S. Alahdab, A. Mäntyniemi and J. Kostamovaara, "A time-to-digital converter (TDC) with a 13-bit cyclic time domain successive approximation interpolator with sub-ps-level resolution using current DAC and differential switch," in 2013 IEEE 56th International Midwest Symposium on Circuits and Systems (MWSCAS), 2013.

[6] Y. Liu, U. Vollenbruch, Y. Chen, C. Wicpalek, L. Maurer, Z. Boos and R. Weigel, "Multi-stage pulse shrinking time-to-digital converter for time interval measurements," in 2007 European Microwave Integrated Circuit Conference, 2007.

[7] K. Cui, Z. Ren, X. Li, Z. Liu and R. Zhu, "A High-Linearity, Ring-Oscillator-Based, Vernier Time-to-Digital Converter Utilizing Carry Chains in FPGAs," in IEEE Transactions on Nuclear Science, 2017.

[8] R. Narasimman, A. Prabhakar and N. Chandrachoodan, "Implementation of a 30 ps resolution time to digital converter in FPGA," in 2015 International Conference on Electronic Design, Computer Networks Automated Verification (EDCAV), 2015.

[9] M. Kanoun, Y. Berube-Lauziere and R. Fontaine, "High precision time-to-amplitude converter for diffuse optical tomography applications," in 2008 3rd International Conference on Design and Technology of Integrated Systems in Nanoscale Era, 2008.

[10] J. S. Tandon, S. Komatsu, T. J. Yamaguchi and K. Asada, "A comparative study of body biased time-to-digital converters based on stochastic arbiters and stochastic comparators," in 2016 14th IEEE International New Circuits and Systems Conference (NEWCAS), 2016.

[11] M. S. Kim, Y. B. Kim and K. K. Kim, "All-digital phased-locked loop with local passive interpolation time-to-digital converter based on a tristate inverter," in 2012 IEEE 55th International Midwest Symposium on Circuits and Systems (MWSCAS), 2012.

[12] Y. J. Park and F. Yuan, "A 12.88 MS/s 0.28 pJ/conv.step 8-bit stage-interleaved pulse-shrinking time-to-digital converter in 130 nm CMOS," in 2015 IEEE 58th International Midwest Symposium on Circuits and Systems (MWSCAS), 2015.