Constructive Use of Process Variations: Reconfigurable and High-Resolution Delay-Line

Wenhao Wang
Northeastern University
wang.wenh@northeastern.edu

Yukui Luo
Northeastern University
luo.yuk@northeastern.edu

Xiaolin Xu
Northeastern University
x.xu@northeastern.edu

***Abstract*—Delay-line is a critical circuit component for high-speed electronic design and testing, such as high-performance FPGA and ASICs, to provide timing signals of specific duration or duty cycle. However, the performance of existing CMOS-based delay-lines is limited by various practical issues. For example, the minimum propagation delay (resolution) of CMOS gates is limited by the process variations from circuit fabrication. This paper presents a novel delay-line scheme, which instead of mitigating the process variations from circuit fabrication, constructively leverages them to generate time signals of specific duration. Moreover, the resolution of the proposed delay-line method is reconfigurable, for which we propose a Machine Learning modeling method to assist such reconfiguration, i.e., to generate time duration of different scales. The performance of the proposed delay-line is validated with HSpice simulation and prototype on a Xilinx Virtex-6 FPGA evaluation kit. The experimental results demonstrate that the proposed delay-line method achieves an ultra-high resolution of sub-picosecond.**

***Index Terms*—Delay-line, sub-picosecond, machine learning.**

I. Introduction and Background

Alongside the rapid semiconductor industry development, more cutting-edge areas, such as quantum computing [1], remote sensing [2], and space science [3] now have increasing demand for high-quality timing signals. Besides, high-quality timing signals can also be used as clock signals for FPGA and ASIC evaluation. Therefore, new all-digital techniques are needed to allow fast on-chip or off-chip generations for timing signals with sub-picosecond resolution, such as delay-line. However, like other mixed-signal circuits, the resolution of current delay-line circuits in advanced CMOS technologies are exacerbated by the increased process variations from circuit fabrication [4] [5]. Such limits cannot be ignored when the time interval of interest goes below sub-10 picoseconds or even sub-picosecond. For example, the resolution of state-of-the-art: vernier delay-line [6], is limited by the minimum propagation delay of the gate-level component [7] [8] [9].

Delay-line circuits are mainly used in two application scenarios: 1) generating time delay for a circuit or system; 2) measuring the time delay of ultra-small interval. A delay-line circuit is usually used with other digital pre- or post-processing components, making all-digital delay-line a preferred solution. Various delay-line solutions based on analog-circuit have been proposed, such as using amplifiers [10], analog memories [11], and photons [12]. These methods, although can provide timing delay with relatively high-resolution, cannot be easily integrated with digital system or circuit. Marteau *et al.* [13] proposed a time-to-digital converter (TDC), with a large time generation range as $10ns$. However, this design could only achieve a resolution of $240ps$, limiting its applicability in cutting-edge areas. Morales *et al.* [14] proposed an all-digital reconfigurable delay-line based on current-starved and shunt-capacitor inverter, however, with a relatively low resolution of $340ps$. Zhang *et al.* [15] proposed to use coarse clock counters with a two-stage delay-line loop shrinking interpolator to increase the resolution of the TDC, which achieves a high resolution of $8.5ps$ However, the time generated by this method is not programmable and adaptable into different systems.

Aiming at extremely high-resolution delay-line with highly-refined precision, this work proposes an all-digital and reconfigurable delay-line. To the best of our knowledge, different from the existing designs, the proposed method is the first work that constructively leverages the process variation of CMOS for delay-line construction. The proposed scheme is validated with HSpice simulation and FPGA implementations. To assist the delay-line configuration, we propose a machine learning-based modeling method for the proposed delay-line circuit. This method could accurately model the delay-line circuit output by using no more than 1% outputs, which provides users an efficient way to reconfigure the proposed delay-line scheme for generating different timing signals. We validate the proposed design with both HSpice simulation and FPGA prototype. The experimental results show that our proposed delay-line could achieve reconfigurable resolution of sub-picosecond.

II. Process Variation-based Delay-line

A. Proposed delay-line scheme

The schematic of the proposed delay-line circuit is shown in Fig. 1, which consists of two configurable paths *top (t)* and *bottom (b)* and each path is composed of N 2–1 MUXs. The the *path selecting* signals of these MUXs are provided by a binary vector $\mathbf{S}$, where $\mathbf{S} = \mathbb{B}^{2N}$, $\mathbb{B} = \{0, 1\}$. The two paths (t and b) share the same input signal (i.e., pulse in), which propagates through the circuit via the paths chosen by $\mathbf{S} = \{s^0, s^1 \dots s^{N-1}\}$. The two outputs out_t and out_b are connected to a D Flip-flop (DFF) to generate a binary checking bit (cb) depending on whether out_t or out_b arrives earlier. If denoting the time interval between the two output signals: out_t and out_b with t_{out}, then for a given *select* vector S, the relationship between the digital checking bit cb, t_{out}, and S can be defined with Eq.1.

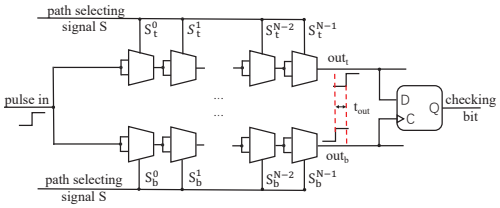


Fig. 1: Delay-line circuit based on process variations.

$$cb = \begin{cases} 1, & if\ t_{out} = f(S) > 0 \\ 0, & if\ t_{out} = f(S) < 0 \end{cases} \quad (1)$$

Assuming that one of the two output signals, e.g., out_t or out_b, is the original signal for which we need to generate a specific delay, then the other output (out_b or out_t) is the delayed version of the original signal. The time difference t_{out} between out_b and out_t will be the generated incremental delay.

B. Statistical characteristics of t_{out}

HSpice simulation with the 45 nm Predictive Technology Model (PTM) [16] is used to studies t_{out} and PS. The process variations are introduced as the physical size (e.g., effective channel length/width (L_{eff}/W_{eff})) and the threshold voltage of transistors (V_{th}) of CMOS transistors following [16] and [17]. For example, the standard deviation $\sigma_{V_{th}}$ of threshold voltage is calculated with Eq. 2, and $A_{V_{th}}$ is set as $1.8\ mV$ following [18]. The standard deviation of L_{eff} and W_{eff} are set as 10% of their nominal values following [19].

$$\sigma_{V_{th}} = \frac{A_{V_{th}}}{\sqrt{W_{eff} L_{eff}}} \quad (2)$$

We simulate the t_{out} and of a $64-$stage ($N = 64$) delay-line for $50,000$ random S vectors. The probability density function (PDF) of t_{out} and its curve fit are shown in Fig. 2a. The simulation results indicate that t_{out} follows approximately Gaussian distribution in $[-80\ ps, +80\ ps]$ with a mean value $\sim 0\ ps$. Thus, according to Eq. 1, the probability of $cb = 1$ is approximately equal with that of $cb = 0$ ($\sim 50\%$).

C. Machine learning-assisted modeling method

In this section, we introduce the machine learning-assisted modeling method, which do not need to measure total 2^N t_{out} configurable patterns, to fast modeling the proposed design. The propagation delay of two path can be derived below:

$$\alpha^i = \frac{(t^i_{t0} + t^i_{t1}) + \hat{s}^i_t (t^i_{t0} - t^i_{t1})}{2}$$
$$\beta^i = \frac{(t^i_{b0} + t^i_{b1}) + \hat{s}^i_b (t^i_{b0} - t^i_{b1})}{2} \quad (3)$$

where α^i and β^i represent the propagation delay of the i^{th} top and $bottom$ MUXs, respectively, and t^i_{t0}, t^i_{b0}, t^i_{t1}, and t^i_{b1} denote the propagation delay of channel 0 and 1 in the i^{th} top and $bottom$ MUXs. Besides, $\hat{s}^i$ is a linear conversion of s^i: $\hat{s}^i_{t/b} = 1 - 2 * s^i_{t/b}$, where t/b again stands for *top* or *bottom*.

Based on Eq. 3, the relationship between t_{out} and S can be rewritten as Eq. 4, where there indicates a linear additive delay model between the *select* vectors S and t_{out}. Therefore, a Support Vector Machine (SVM) is used to model the relationship between t_{out} and the corresponding S. As defined in Eq. 1, the digital checking bit cb, which is in accordance with the sign of t_{out}, can be used to quantify t_{out}. More specifically, we use some $cb - S$ pairs to train the SVM model and read out the parameters of the trained model to predict the cb values for rest unknown S. The vision is that if the trained SVM model can predict the correct cb value for any S with a high success rate, then $\mathbf{P}_{model}$ should also has high accuracy in formulating a numeric value mp_t_{out} (mp_t_{out}: a model predicted t_{out}) that has high linearity with the actual t_{out}.

$$t_{out} = f(S) = \sum_{0}^{N-1} \alpha_i - \sum_{0}^{N-1} \beta_i \quad (4)$$

As theoretical validation, we use a training set of $1,000$ random $cb-S$ pairs from HSpice simulation. The prediction rate of the trained model in Fig. 2b shows that the accuracy converges to 100% as more samples used (around 1%). Moreover, we extract the corresponding t_{out} for $50,000$ *select* vectors from HSpice simulation. The $\mathbf{P}_{model}$ predicted and simulated t_{out} are depicted in Fig. 2c, which proves the linear relationship between the two data sets: $mp\_t_{out} = coe \times$(simulated t_{out}), where coe denotes the correlation coefficient and its calculation is shown in Proc. 1. Note that Fig. 2c is only for visualizing mp_t_{out} and simulated t_{out} values. This shows that the $\mathbf{P}_{model}$ for a delay-line can predict the actual t_{out} for any S without enrolling it in advance. Note that this paper focuses on the generation of $\widehat{t_{out}}$ with high-resolution. As many other related works [15] [14] or a standard instrument calibration, we assume the timing output ($\widehat{t_{out}}$) of the delay-line circuit can be accurately measured or characterized by other circuits or instruments, which are out of the scope of this work.

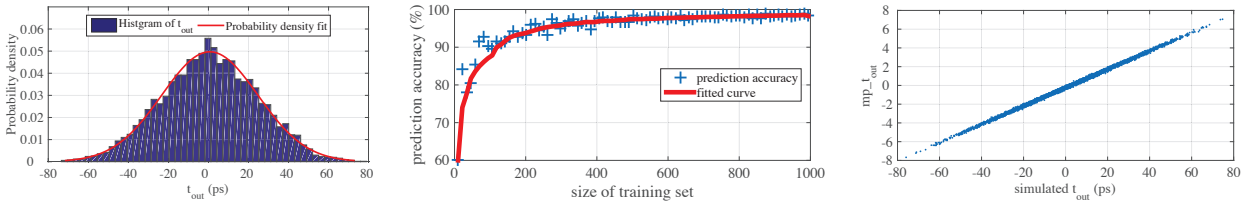
Procedure 1 Train a Machine Learning model for a delay-line and formulate the coefficient (coe) between t_{out} and model predicted t_{out}: mp_t_{out}.

Input: A set of random *select* vectors **S** and corresponding output values **cb** from a delay-line circuit.
Output: The $\mathbf{P}_{model}$ for the delay-line circuit.
Output: The coefficient coe between mp_t_{out} and actual t_{out}.
1: $\mathbf{P}_{model} \leftarrow$ **SVM(S, cb)** {train SVM model for delay-line}
2: $\widehat{t_{out}}$ = **Delay-Line**($\widehat{S}$) {apply a specific select vector S to the delay-line circuit and obtain timing output $\widehat{t_{out}}$.}
3: $\widehat{mp\_t_{out}}$ = $\mathbf{P}_{model}(\widehat{S})$ {apply a specific select vector S to the SVM model and obtain the modeled timing output $\widehat{mp\_t_{out}}$.}
4: **return** $coe = \frac{\widehat{t_{out}}}{\widehat{mp\_t_{out}}}$

III. Experimental Validation with FPGA

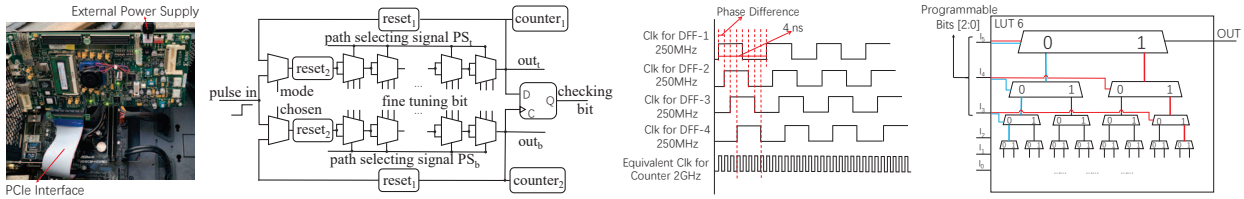
A. Experimental Setup

We implement the proposed delay-line structure with 16 stages ($N = 16$) on a Virtex-6 FPGA. The experimental setup is shown in Fig. 3a, in which the FPGA board is connected to a PC host through PCIe interface. To facilitate the implementation and measurement of the delay-line on FPGA, we slightly



(a) (b) (c)

Fig. 2: (a) shows the t_{out} follows approximately Gaussian distribution. (b) shows relationship between prediction rate and number of applied training sample. (c) shows linear relationship between simulated t_{out} and model predicted t_{out} by $\mathbf{P}_{model}$.



(a) (b) (c) (d)

Fig. 3: (a) FPGA (ML605) setup with PCIe as the interface. (b) Schematic of FPGA implementation. (c) Clocks used in the counter. (d) Programmable bits in LUT6 for the delay tuning purpose.

modify the schematic in Fig. 1 to a new structure shown in Fig. 3b. While keeping the core functional components, this new structure mainly has two parts: (1) two feedback loops for measurement purpose, and (2) the main delay-line circuit as presented in Sec. II-A. Each 2–1 MUX in the delay-line circuit is instantiated with an individual LUT.

We use Xilinx user constraint file (`UCF`) to instantiate each component as a `Hardmacro`, to avoid bias introduced by the automatic placement and routing of FPGA development tools. We then use the LUT inputs to further mitigate the bias effect from automatic placement and routing. Specifically, the inputs I3 $\sim$ I5 out of the 6 inputs of an LUT6 are used as *programmable bits*, as shown Fig. 3d. Due to the tiny path delay differences (the blue path 000 and the red path 111), reconfiguration of these inputs can be used for fine-tuning the propagation delay of each MUX. The top and bottom loops of the modified schematic in Fig. 3b are the feedback loops used for the delay-line measurement. Each feedback loop is composed of two reset gates, one counter, and one MUX. The counter is specially designed to work at high frequency, which consists of 4 independent flip-flops (DFFs) driven by both positive edge and negative edge of 4 clocks at the same frequency (250MHz), but with different phases (Fig. 3c). Based on the control signal from the MUX, there are two usable modes: *checking bit generation mode* (mode=1) and *delay measuring mode* (mode=0), as elaborated below. The reset gates are used to clean the residual pulse oscillation in the loop when the mode is switched.

1) Checking bit generation mode: This mode is used to generate checking bits (cb) for different path selecting (S) vectors, with which the machine learning model is trained. In this mode, the input pulse only propagates through the delay-line circuit. Note that all S inputs and fine-tuning bits are configured as needed before applying an pulse input.

TABLE I: Hardware footprint of the delay-line and auxiliary circuits.

		Delay-line circuit		Auxiliary circuit (Feedback-loop, counter, reset and cb DFF)	
	Total	Used	Used(%)	Used	Used(%)
LUT	150720	180	0.1194	27	0.0179
DFF	301440	124	0.0411	15	0.0046

2) Delay measuring mode: This mode is used to measure the generated t_{out} for each S. The basic idea of this mode is to accumulate the immeasurable tiny delay difference between the two paths with the feedback loop. In this mode, the input is the pulse signal, and the outputs are numbers from the two counters. Specifically, each time the two pulses pass through each delay-line, the delay difference t_{out} will be accumulated and the output numbers of the two counters will both add 1. With the two pulse signals oscillate/propagate in the feedback loop, the delay difference t_{out} between them keeps accumulating until becomes large enough, i.e., for quantifying t_{out}. More detailed, for a path selecting vector S, if denoting the measurement time as t_c, the count numbers of counter$_1$ and counter$_2$ as C_1 and C_2, then the t_{out} for S can be denoted with Eq. 5:

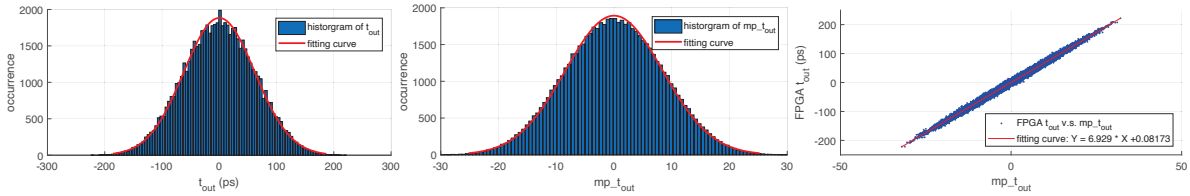
$$t_{out} = t_c * (\frac{1}{C_1} - \frac{1}{C_2}) \tag{5}$$

Tab. I presents the hardware footprint of the proposed delay-line implementation on a 16-bit (i.e., N=16) Virtex-6 FPGA. Note that the auxiliary circuits are only used for performance analysis in the FPGA implementation.

B. Experimental results

Two types of data are collected from the FPGA implementation: the binary output cb and the outputs (N_1 and N_2) of the two counters for all 2^{16} S input patterns.

1) Checking bits: After fine-tuning, the delay-line achieves a balanced percentile in generating 1 and 0. Specifically, in



(a) (b) (c)

Fig. 4: (a) Time delay t_{out} generated from FPGA implementation. (b) The predicted time delay mp_t_{out} of 2^{16} testing set using the SVM model. (c) The linear relationship between FPGA-generated t_{out} and SVM model predicted mp_t_{out}.

TABLE II: Comparison with other related works.

	Tech. node nm	Programmable	Modeling Method	Resolution ps
Marteau *et al.* [13]	Altera-130	✓	✗	240 (On Avg.)
Morales *et al.* [14]	ASIC-130	✓	✗	340 (On Avg.)
Zhang *et al.* [15]	Actel-130	✗	✗	8.5 (On Avg.)
This Work	Xilinx-40	✓	✓	0.102 (On Avg.)

our implementation, the percentage of $cd = 1$ and $cd = 0$ are $\frac{32178}{2^{16}} = 49.1\%$ 1 and $\frac{33358}{2^{16}} = 50.9\%$, respectively.

2) Measurement of t_{out}: Following Eq. 5, we set the working time length $t_c = 10\mu s$ and calculate the t_{out} for all 2^{16} S patterns.. The experimental results demonstrate that the smallest t_{out} is $0.102ps$ (on average). The range of t_{out} is $[-209.8ps, 215.7ps]$. In addition, we compare the performance of our design with other state-of-the-art delay-line method in Tab. II. Among these four delay-line designs, our design achieves the highest resolution with programmability, and is also the only design providing the fast modeling method for manufacturers and users.

To validate the SVM modeling method introduced in Sec. II-C, we plot all the calculated t_{out} for the 2^{16} S patterns in Fig. 4a. Similar to the HSpice simulation results in Fig. 2a, the t_{out} values generated by the FPGA prototype also follow approximate Gaussian distribution, which affirms the correctness of the delay-line implementation and t_{out} measurements. We utilize $10,000$ out of the 2^{16} collected cb values as the training set to train an SVM model, and $50,000$ out of the cb values are used as the testing set. The trained SVM model achieves a high success rate 99.02% in predicting cb for any unknown S. Following Eq. 4, the SVM model is then used to derive the model predicted delay mp_t_{out} for S patterns in the testing set, which are are shown in Fig. 4b. We plot the model predicted mp_t_{out} and the FPGA-generated t_{out} in Fig. 4c. Similar to Fig. 2c, we find there exists linear relationship between these two data sets, and the coefficient (*coe*) between FPGA t_{out} and mp_t_{out} is 6.929 following Proc. 1. These results demonstrate that the SVM model can be used to reconfigure the delay-line, i.e., using a S for specific t_{out} generation.

IV. Conclusion

In this paper, we propose a process variation-based, all-digital, and reconfigurable delay-line to generate time delay with ultra-high resolution of sub-picosecond. Unlike conventional delay-lines that suffer the inaccuracy and low precision caused by process-variations, the new design leverage them in a constructive way. A machine learning modeling method is proposed to assist the reconfiguration of the proposed delay-line scheme. The experimental results demonstrate that our design is fully compatible with modern ASIC and FPGA design flow, with generating timing delay of ultra-high-resolution ($\approx 0.1ps$).

References

[1] G. Leonardo *et al.*, "A 32× 32-pixel time-resolved single-photon image sensor with 44.64 μm pitch and 19.48% fill-factor with on-chip row/frame skipping features reaching 800khz observation rate for quantum physics applications," in *ISSCC*, 2018.

[2] R. K. Henderson *et al.*, "A 192 x 128 time correlated spad image sensor in 40-nm cmos technology," *JSSC*, 2019.

[3] H. Chung *et al.*, "Design-space exploration of backplane receivers with high-speed adcs and digital equalization," in *CICC*. IEEE.

[4] N. Pourmousavian *et al.*, "A 0.5-v 1.6-mw 2.4-ghz fractional-n all-digital pll for bluetooth le with pvt-insensitive tdc using switched-capacitor doubler in 28-nm cmos," *JSSC*, 2018.

[5] S. Li, X. Xu, and W. Burleson, "Pvtmc: An all-digital sub-picosecond timing measurement circuit based on process variations," in *2019 ISVLSI*.

[6] H. Wang *et al.*, "A reconfigurable vernier time-to-digital converter with 2-d spiral comparator array and second-order delta sigma linearization," *IEEE JSSC*, 2018.

[7] A. Stillmaker *et al.*, "Scaling equations for the accurate prediction of cmos device performance from 180 nm to 7 nm," 2017.

[8] X. Xu, S. Li, R. Kumar, and W. Burleson, "When the physical disorder of cmos meets machine learning," *Low Power Semiconductor Devices and Processes for Emerging Applications in Communications, Computing, and Sensing*, 2018.

[9] S. Li, X. Xu, and W. Burleson, "Ccatdc: A configurable compact algorithmic time-to-digital converter," in *2017 IEEE Computer Society Annual Symposium on VLSI (ISVLSI)*, pp. 501–506.

[10] G. Chen *et al.*, "An adaptive analog circuit for lvdt's nanometer measurement without losing sensitivity and range," *IEEE Sensors Journal*, 2014.

[11] Z. Yu *et al.*, "A programmable analog delay line for micro-beamforming in a transesophageal ultrasound probe," in *IEEE ICSICT*, 2010.

[12] D. M. Beggs *et al.*, "Ultrafast tunable optical delay line based on indirect photonic transitions," *Physical review letters*, vol. 108, no. 21.

[13] J. Marteau *et al.*, "Implementation of sub-nanosecond time-to-digital convertor in field-programmable gate array: applications to time-of-flight analysis in muon radiography," *Measurement Science and Technology*, 2014.

[14] J. I. Morales *et al.*, "Design and evaluation of an all-digital programmable delay line in 130-nm cmos," in *RPIC*, 2019.

[15] J. Zhang and D. Zhou, "An 8.5-ps two-stage vernier delay-line loop shrinking time-to-digital converter in 130-nm flash fpga," *IEEE Transactions on Instrumentation and Measurement*, 2018.

[16] W. Zhao *et al.*, "New generation of predictive technology model for sub-45 nm early design exploration," *T-ED*, 2006.

[17] A. Agarwal *et al.*, "Statistical timing analysis for intra-die process variations with spatial correlations," in *ICCAD*. IEEE, 2003, p. 900.

[18] K. J. Kuhn, "Reducing variation in advanced logic technologies: Approaches to process and design for manufacturability of nanoscale cmos," in *IEDM*.

[19] M. Anis and M. H. Aburahma, "Leakage current variability in nanometer technologies," in *System-on-Chip for Real-Time Applications, 2005.*