qSSTA: A Statistical Static Timing Analysis Tool for Superconducting Single-Flux-Quantum Circuits

Bo Zhang, Mingye Li, and Massoud Pedram

***Abstract*—Superconducting single flux quantum (SFQ) technology is an ultra-high performance and low power technology. The technology, however, lacks many of the design automation tools and capabilities that are commonplace in CMOS technology. This article describes methods to efficiently find the conditional probability density function (PDF) of the minimum workable clock period of SFQ circuits in view of manufacturing-induced process variations and presents qSSTA, a statistical static timing analysis tool targeting SFQ circuits. Following a grid-based correlation model, qSSTA represents spatial correlation of SFQ gates at different positions with respect to process parameters. By approximating timing characteristics of SFQ gates in a linear model, qSSTA is able to estimate the clock period as a normal random variable. Furthermore, process variations that generally result in extra delays in CMOS circuits can result in functional errors in SFQ circuits. qSSTA derives the closed form of the conditional PDF of the clock period under the scenario where all SFQ gates in the circuit work correctly. Compared to Monte Carlo simulations on look-up tables, experimental results show that the average percentage errors are 0.89% for the mean values, 8.04% for the standard deviation, and 0.61% for the 98-percentile point, whereas the runtime of qSSTA is 83% faster on average.**

***Index Terms*—Path pruning, single flux quantum (SFQ), statistical static timing analysis.**

I. Introduction

SUPERCONDUCTING single flux quantum (SFQ) technology is a promising beyond-CMOS technology, which is promising to achieve much higher performance and lower power consumption even compared to 5 nm complementary metal oxide semiconductor (CMOS) technology. Unfortunately, the state of electronic design automation tools for SFQ circuits far lacks that of CMOS. It is thus necessary to adopt and appropriately modify many of the analysis and design optimization tools from CMOS to SFQ domains.

Manuscript received December 24, 2019; revised June 22, 2020; accepted June 22, 2020. Date of publication June 25, 2020; date of current version August 4, 2020. This work was supported by the Office of the Director of National Intelligence (ODNI), Intelligence Advanced Research Projects Activity (IARPA), via the U.S. Army Research Office under Grant W911NF-17-1-0120. This article was recommended by Associate Editor V. Vernik. *(Corresponding author: Bo Zhang.)*

Bo Zhang is with Electrical and Computer Engineering, University of Southern California, Los Angeles, CA 90007 USA (e-mail: zhan254@usc.edu).

Mingye Li is with the Ming Hsieh Department of Electrical Engineering, University of Southern California, Los Angeles, CA 90007 USA (e-mail: mingyel@usc.edu).

Massoud Pedram is with the Electrical Engineering, University of Southern California, Los Angeles, CA 90007 USA (e-mail: pedram@usc.edu).

Color versions of one or more of the figures in this article are available online at https://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TASC.2020.3005082

Static timing analysis (STA) tools are necessary to enhance the circuit design and optimization. These tools help determine the timing parameters of a circuit based on static (closed-form or lookup table (LUT)-based) models of the gate and interconnect delays under different process corners. A process corner is a set of process parameter values that determine the circuit performance with respect to one figure of merit, e.g., processing speed, energy efficiency, etc. In this way, they eliminate the dependency of the circuit timing estimation on input patterns and thereby avoid expensive dynamic (input pattern-based) timing analysis done by circuit simulators such as Spice. STA has many advantages [1], [2], such as exhibiting linear time complexity with respect to the circuit size and producing deterministic timing analysis results. The nominal delay of an SFQ clocked gate is small (in the order of picoseconds) such that the delay deviation due to process variations can be a significant percentage of this nominal delay value. The result has been low accuracy (often hugely pessimistic) estimations done by corner-based STA tools, which is mainly due to worst-case estimations of low-probability scenarios.

Statistical static timing analysis (SSTA) has thus emerged as the standard industry-adopted solution to handle sources of variability (including manufacturing-induced process variations) by modeling them as random variables (RVs). The propagation delays of logic gates, which are typically nonlinear functions of physical parameters and the context in which they are instantiated, are also modeled as RVs. Using such a formalism, SSTA then attempts to estimate the PDFs of various circuit timing parameters like the clock period. However, very few studies [3] have focused on the development of SSTA for SFQ technology.

The basic gate behaviors in the SFQ technology [4] are different than those of the CMOS technology. Instead of using voltage levels, SFQ circuit technology represents logic one with the presence of a magnetic flux quantum, which is seen as a voltage pulse whose integral over time yields $2.07 * 10^{-15}$ mV $\cdot$ ps of magnetic flux). A logic zero is then represented by the absence of a magnetic flux quantum on a timing window of interest, T. Generic SFQ gates like AND, OR, XOR, and NOT are clocked gates, which process their inputs by changing the states of their internal interferometer and produce their outputs only in response to a clock pulse. Each SFQ clocked gate is thus equivalent to a combination of a logical gate and D flip-flop(s) in CMOS technology. Therefore, the direct application of existing SSTA to SFQ circuits is not possible. In CMOS circuits, input patterns that are held steady need not be reapplied, whereas in SFQ circuits the input patterns have to be reapplied because

the input patterns are "consumed" after the clock comes in and resets the internal state of the SFQ logic gate. While this is true, it is not an effect that we are exploiting in this article, and hence, our equivalence (which is simply meant to signify the clock-synchronized output behavior of SFQ gates) holds.

This article further considers that an SFQ clocked gate could have a functional failure under process variations although it can work with the nominal design values. The distribution of the clock period thus becomes a (conditional) RV conditioned on the case that all SFQ gates in the circuit are working correctly. This new condition makes the computation of the clock period distribution for SFQ circuits more complex. On the other hand, the gate-level deep pipeline nature of SFQ circuits allows us to use path-oriented timing analysis, which yields higher accuracy with lower computation time compared to the block-oriented analysis that is common in CMOS-based circuits.

In this article, we propose a new SSTA method to compute the clock period distribution of SFQ circuits efficiently. The main contribution of this article is thus to implement a SSTA tool, called qSSTA, which have characteristics as follows.

1) Consider functional failure of an SFQ gate under process variations.
2) Extract linear models for propagation delay calculation and functional success determination of an SFQ gate.
3) Account for spatial and topological correlations to make the clock period distribution less pessimistic.
4) Formulate and solve the problem to find the distribution of the clock period.
5) Efficiently derive the closed form of the PDF of the clock period for an SFQ circuit.
6) Prune noncritical paths to speed up the computation.

II. Background

A. Circuit Topology

In our design framework, an SFQ circuit is a netlist that typically connects all SFQ gates by passive transmission lines (PTLs). Another way to design an SFQ circuit is to connect all SFQ gates by Josephson transmission lines (JTLs). Straightforwardly, a JTL-wiring based SFQ circuit is a particular case when PTL lengths are 0 μm. This article assumes PTL is ideal with a constant propagation velocity of 100 μm/ps. To ensure the functionality of the SFQ circuit, we must guarantee that an SFQ clocked gate can successfully capture the signals at its input ports that are generated by its upstream SFQ clocked gates. In addition, each SFQ clocked gate must also generate its output upon arrival of a clock pulse at its clock port. Therefore, an SFQ circuit can be decomposed into (data) signal paths and clock paths for timing analysis. A signal path refers to an interconnection from a source clocked gate to a sink clocked gate, whereas a clock path refers to an interconnection from the clock source to an SFQ clocked gate. Because each signal path has two clocked gates, we call the clock path associated with the source clocked gate the *launch path* and the clock path associated with the sink clocked gate the *capture path*. A clock path can serve as the launch path for one signal path and the capture path for another signal path.

SFQ clocked gates have a fan-out capability of one. An SFQ splitter, a combinational gate with a fan-out capability of two, is thus needed to provide the multiple fan-out capabilities for SFQ clocked gates. Thus, (data) signal paths and clock paths may comprise one or more splitters and PTLs. The SFQ technology follows many of the timing definitions that are commonplace in the CMOS technology [5] (e.g., setup and hold times, clock-to-Q propagation delays, clock skew) albeit with some differences. So although it is possible to develop SSTA for SFQ circuits starting from the existing SSTA tools for CMOS circuits, a number of changes and augmentations are required. Because of the gate-level deep pipeline nature of SFQ circuits, the number of signal paths in an SFQ circuit is linear in the number of SFQ clocked gates in the circuit.

Depending on internal methodology to check timing constraints, SSTA falls into two categories of SSTA: block-based SSTA and path-based SSTA. Block-based SSTA computes arrival time of a gate output based on the latest arrival times of its input signals and delay through the gate, and subsequently uses this output arrival time for downstream gates, whereas path-based SSTA computes the arrival times of all signal paths first and determines the latest arrival time among all such paths in the end. Compared with block-based SSTA, path-based SSTA is more accurate because it avoids arrival time approximation of gate inputs. Because of the linearity of the number of paths in an SFQ circuit, path-based SSTA is thus the right choice for SFQ circuits.

B. Setup Time Checks

Path-based SSTA needs to check the setup time constraints for all signal paths to determine a workable clock period for the target SFQ circuit. Inequality (1) is the setup time check for the *i*th signal path. DP_i denotes delay of the data signal from the source clocked gate to the sink clocked gate of the *i*-th path, including the clock-to-output propagation delay of the source clocked gate. LP_i and CP_i denote delays of the clock signal from the clock source to the source clocked gate and to the sink clocked gate of the *i*th path, respectively. Moreover, Setup_i denotes the setup time of the sink clocked gate. Then, we define $\mathrm{SetupCheck}_i = \mathrm{LP}_i + \mathrm{DP}_i - \mathrm{CP}_i + \mathrm{Setup}_i$. Clearly,

$$\mathrm{SetupCheck}_i \leqslant T. \tag{1}$$

The optimum clock period T for a circuit is defined as the minimum value of the clock period so that setup time checks for all *N* signal paths in the circuit are successful. Equivalently, the optimum clock period T is equal to the maximum of all $\mathrm{SetupCheck}_i$.

$$T = \max_{1 \leqslant i \leqslant N} \mathrm{SetupCheck}_i. \tag{2}$$

III. Problem Formulation

A. Spatial Correlations

Spatial correlations are typically used to capture small changes of circuit parameters that arise from imprecision and uncertainties associated with the chip manufacturing processes.

Chips in both SFQ and CMOS technology are manufactured layer by layer. Circuits in both SFQ and CMOS technology are gates connected by metal connections. Those similarities support the presumption that the spatial correlation model developed and used successfully in CMOS circuits can also be used in SFQ circuits. In the SFQ technology, resistances (*R*), inductances (*L*), and areas of Josephson junction (*B*) are three critical physical parameters affecting propagation delays. All of these parameters are subject to manufacturing induced process variations [6], [7] and are, therefore, modeled as normal RVs based on the central limit theorem. Therefore, as functions of *R*, *L*, and *B*, variables DP_i, LP_i, CP_i, Setup_i, and even T are all RVs.

Generally, SSTA is a technology-independent technique designed to model parameter variations. Therefore, as a spatial correlation model, reference [8], which provides a multilevel quad-tree model, also applies to the SFQ technology. A multilevel quad-tree model can be extended to partition the die into nrow rows and ncol columns, resulting in nrow $*$ ncol $= m$ grid cells. The physical parameter variation within each grid cell (specified as a percentage change around the nominal value of the physical parameter) is the same, but this variation value is possibly different for different grid cells. There are specified correlation parameters that describe the relationship between variation sources of any pair of grid cells. Strictly speaking, the correlation between two physical parameters depends on the distance of their corresponding logic gates. However, this distance-based spatial correlation model is computationally expensive for a large-scale circuit, since we need to consider the distances of each pair of gates, regardless of whether they are logically connected or not. The grid-based model is a coarsely grained version of the distance-based model where two physical parameters are correlated if their corresponding gates are located in nearby grid cells. Obviously, the accuracy of the grid-based spatial correlation model approaches that of its distance-based counterpart as the number of grid cells approaches the number of logic gates. The grid-based model is, however, much more scalable and useful than the distance-based one because of its bounded complexity (regardless of the number of logic gates in a circuit which may in tens or even hundreds of thousands of gates, the 2-D grids that define the spatial correlations may be limited in size). This bounded model size enables us to statistically analyze a circuit without running into computational complexity issues.

The mathematical representation of RVs for *R*, *L*, and *B* can be explained by an example of four gird cells shown in Fig. 1. ΔR_j, ΔL_j, and ΔB_j denote the normalized process variations of *R*, *L*, and *B* in the grid cell *j* (they are specified as a percentage of the corresponding nominal parameter values). Considering a resistance R_1^u in grid cell 1, its RV is decomposed into a nominal base value $R_{\mathrm{nom},1}^u$ and a normalized process variation component ΔR_1 in (3). By applying the central limit theorem, ΔR_1 can be assumed to be a normal RV with a mean 0 and a standard deviation σ_R. Note that any other resistance R_1^v in grid cell 1 can also be modeled similarly with its nominal value $R_{\mathrm{nom},1}^v$ and spread ΔR_1. In this article, we assume the process variation will not change during the program execution once a chip is manufactured.

Grid 1 ΔR_1, ΔL_1, ΔB_1	Grid 2 ΔR_2, ΔL_2, ΔB_2
Grid 3 ΔR_3, ΔL_3, ΔB_3	Grid 4 ΔR_4, ΔL_4, ΔB_4

Fig. 1. Example of spatial correlation.

$$\begin{aligned} R_1^u &= R_{\mathrm{nom},1}^u(1+\Delta R_1) \\ R_1^v &= R_{\mathrm{nom},1}^v(1+\Delta R_1) \\ E(\Delta R_1) &= 0, \quad \mathrm{Var}(\Delta R_1) = \sigma_R^2 \\ E(R_1^u) &= R_{\mathrm{nom},1}^u, \quad E(R_1^v) = R_{\mathrm{nom},1}^v. \end{aligned} \tag{3}$$

Considering another resistance $R_2^w = R_{\mathrm{nom},2}^w(1+\Delta R_2)$ in grid cell 2, to model the correlation between ΔR_1 and ΔR_2, they are further decomposed into two parts [1], [2]. For example, ΔR_1 is decomposed into a local variation component $\sqrt{p}\,\sigma_R Z_1$ and a global variation component $\sqrt{1-p}\,\sigma_R Z$, where Z_1, Z_2, and Z are independent normal RVs in (4). The parameter p is used to specify the ratio of the local variance over the total variance. Hence, the covariance and correlation between ΔR_1 and ΔR_2 can be easily derived

$$\begin{aligned} \Delta R_1 &= \sqrt{p}\,\sigma_R Z_1 + \sqrt{1-p}\,\sigma_R Z \\ \Delta R_2 &= \sqrt{p}\,\sigma_R Z_2 + \sqrt{1-p}\,\sigma_R Z \\ \mathrm{Cov}(\Delta R_1, \Delta R_2) &= [1-p+p\,\mathrm{Cov}(Z_1, Z_2)]\,\sigma_R^2 \\ \mathrm{Cor}(\Delta R_1, \Delta R_2) &= 1-p+p\,\mathrm{Cor}(Z_1, Z_2). \end{aligned} \tag{4}$$

All process variations can be collected to form a correlated multivariate normal random vector $\mathbf{X}$ with mean value of 0 and a covariance matrix of Σ as shown in (5). X_j is the *j*th RV, and the dimension of $\mathbf{X}$ is 3 m.

$$\begin{aligned} \mathbf{X} &= [\Delta R_1, \ldots, \Delta R_m, \Delta L_1, \ldots, \Delta L_m, \Delta B_1, \ldots, \Delta B_m]^T \\ E(X) &= 0 \\ \Sigma_{i,j} &= \begin{cases} \mathrm{Cov}(X_i, X_j) & \text{if } i \neq j \\ \mathrm{Var}(X_i) & \text{if } i = j. \end{cases} \end{aligned} \tag{5}$$

B. First-Order Model

Although process variations of physical parameters are normal RVs, the propagation delay of an SFQ gate is not a normal RV because it is a nonlinear function of process parameters. The method to deal with the nonnormality of propagation delay

is to rely on a linear model based on first-order Taylor series expansion of the nonlinear delay function [9]. The propagation delay of an SFQ gate becomes a linear combination of two or more normal RVs. Recall that the summation of two correlated normal RVs is also a normal RV. The propagation delay is also modeled as a normal RV. Similarly, clock-to-output delay, setup time, and hold time are also modeled as normal RVs.

The propagation delay of an SFQ gate is dependent not only on process variations but also on the context where the corresponding gate is. Considering gate *i* in the grid cell *j*, its propagation delay pd_i is a linear combination of nominal value nom and process variations ΔR_j, ΔL_j, and ΔB_j in (6). pd_i can be further written as a linear combination of $\mathbf{X}$.

$$\begin{aligned}\text{pd}_i &= \text{nom} + a_R\Delta R_j + a_L\Delta L_j + a_B\Delta B_j \\ &= \text{nom} + \mathbf{b}^T\mathbf{X} \\ b_k &= \begin{cases} a_R & \text{if } X_k = \Delta R_j \\ a_L & \text{if } X_k = \Delta L_j \\ a_B & \text{if } X_k = \Delta B_j \\ 0 & \text{otherwise.} \end{cases}\end{aligned} \tag{6}$$

Consequently, delays of signal paths and clock paths, which are calculated as the summation of the propagation delays of SFQ gates and PTLs on the corresponding paths, are also written as a linear combination of $\mathbf{X}$ in (7). nom_i and each element of the column vector $\mathbf{c_i}$ are results of summation or subtraction of linear combinations for all gates in LP_i, DP_i, CP_i, and Setup_i.

$$\begin{aligned}\text{SetupCheck}_i &= \text{LP}_i + \text{DP}_i - \text{CP}_i + \text{Setup}_i \\ &= \text{nom}_i + \mathbf{c_i}^T\mathbf{X} \\ T &= \max_{1\leqslant i\leqslant N} \text{SetupCheck}_i \\ &= \max_{1\leqslant i\leqslant N} (\text{nom}_i + \mathbf{c_i}^T\mathbf{X}).\end{aligned} \tag{7}$$

SSTA based on the first-order model is efficient since the complexity is linear in the circuit size and number of the process parameters. The primary source of inaccuracies (errors), of course, is the approximation of nonlinear delay by a linear equation. Many recent studies have improved the quality of SSTA by using more accurate (but also computationally more expensive) models. References [10] and [11] extend the above-said first-order model to include quadratic components of the Taylor series expansion. References [12], [13] assume that propagation delays follow a skew-normal distribution. These techniques are, however, greatly increase the computational complexity of SSTA.

C. Functional Success

Generic SFQ gates like AND, OR, XOR, and NOT are clocked logic gates. They are in charge of two tasks: compute the combinational logic function and generate the corresponding output in synchrony with a clock pulse. Therefore, process variations have an impact not only on circuit timing (propagation delay, setup time, and hold time) but also on logic functionality. Margin analysis [14] is a such research field to study functional success. The functional failure of an SFQ clocked gate can be formalized as follows.

1) When an input pattern arrives, an SFQ clocked gate generates an output pulse before the arrival of the clock pulse.
2) If the correct output of an SFQ gate in response to an input pattern is logic one, the gate does not generate an output pulse after the arrival of the clock pulse. Alternatively, if correct output is logic zero, the gate generates an (erroneous) output pulse after the arrival of the clock pulse.

If either of the above cases occurs, the SFQ gate has functionally failed. Examples of functional failures are shown in Section IV.

The functional success (or failure) is a complex, nonlinear function of process parameters and the context where the SFQ clocked gate is situated. Equation (8) is provided to express the functional success of the OR gate in the *j*th grid cell. We define $\text{FS}_{\text{OR},j} = 1$ when the OR gate works correctly and $\text{FS}_{\text{OR},j} = 0$ when it does not. Note that $\text{FS}_{\text{OR},j}$ is either 0 or 1 as a function of process variations only. Logistic classification [15] enables us to use a linear combination to approximate $\text{FS}_{\text{OR},j}$ as the result of a classification problem. If the result is nonnegative, $\text{FS}_{\text{OR},j} = 1$. If the result is negative, $\text{FS}_{\text{OR},j} = 0$. The *k*th value of the column vector $\mathbf{c}_{\text{OR,j}}$ is $a_{R,\text{OR}}$, $a_{L,\text{OR}}$, or $a_{B,\text{OR}}$ if X_k is ΔR_j, ΔL_j, or ΔB_j, and 0 otherwise. The indicator function $\mathbf{1}(x)$ is 1 when the condition x is true, and 0 otherwise.

$$\begin{aligned}\text{FS}_{\text{OR},j} &= \mathbf{1}(\text{nom}_{\text{OR}} + a_{R,\text{OR}}\Delta R_j \\ &\quad + a_{L,\text{OR}}\Delta L_j + a_{B,\text{OR}}\Delta B_j \geqslant 0) \\ &= \mathbf{1}(\text{nom}_{\text{OR}} + \mathbf{c}_{\text{OR,j}}{}^T\mathbf{X} \geqslant 0).\end{aligned} \tag{8}$$

The clock period of an SFQ circuit is valid only when the circuit works, which is equivalent to all gates functioning correctly. Instead of checking the functional success of all SFQ gate instances one at a time, we can only check the functional success of a small number of gate instances. All OR gates in the same grid cell share the same process variations, such that they succeed or fail simultaneously. The problem thus reduces to only check the functionality of all gate types in all grid cells in (9). If a gate type k in grid cell *j* can work, then $\text{nom}_k + \mathbf{c_{k,j}}^T\mathbf{X}$ must be nonnegative. If a circuit can work, then all gate types in all grid cells must have nonnegative results. To simplify the following discussions and proofs, let us define S as the minimum of all $\text{nom}_k + \mathbf{c_{k,j}}^T\mathbf{X}$. As shown in (9), if S is nonnegative, then the circuit can work.

$$\begin{aligned}\text{FS}_{\text{circuit}} &= \prod_{\text{gate type } k \text{ in grid cell } j} \mathbf{1}(\text{nom}_k + \mathbf{c_{k,j}}^T\mathbf{X} \geqslant 0) \\ &= \mathbf{1}(\text{nom}_k + \mathbf{c_{k,j}}^T\mathbf{X} \geqslant 0, \forall k, j) \\ &= \mathbf{1}(\min_{k,j}(\text{nom}_k + \mathbf{c_{k,j}}^T\mathbf{X}) \geqslant 0) \\ &= \mathbf{1}(S \geqslant 0) \\ S &= \min_{k,j}(\text{nom}_k + \mathbf{c_{k,j}}^T\mathbf{X})\end{aligned} \tag{9}$$

D. Goal

Based on above discussions, T is a RV conditioned on the fact that the SFQ circuit is working correctly. The cumulative density function (CDF) of T can be formed as in (10) by applying the Bayes' theorem

$$P(T \leqslant t|\text{FS}_{\text{circuit}} = 1) = \frac{P(T \leqslant t, \text{FS}_{\text{circuit}} = 1)}{P(\text{FS}_{\text{circuit}} = 1)} = \frac{P(T \leqslant t, S \geqslant 0)}{P(S \geqslant 0)}. \tag{10}$$

The corresponding PDF of T is

$$f(T = t|\text{FS}_{\text{circuit}} = 1) = \frac{f(T = t, S \geqslant 0)}{P(S \geqslant 0)}. \tag{11}$$

The goal of SSTA is to find the PDF efficiently and accurately.

IV. Statistical Static Timing Analysis

A. Principal Component Analysis

Because two RVs X_i and X_j are in general correlated, the operations performed on them (e.g., the maximum operation) can be complicated. Therefore, it is difficult to find the PDFs of T and S. The problem can be simplified by applying principal component analysis (PCA) [16]. PCA is a statistical procedure that uses an orthogonal linear transformation to project correlated normal RVs into independent normal RVs. $\mathbf{X}$ can be transformed into a multivariate standard normal random vector $\mathbf{Z}$ as shown in (12)

$$\mathbf{Z} = [Z_1, \ldots, Z_{3m}]^T$$

$$D = \begin{bmatrix} \sqrt{\lambda_1} & 0 & \ldots & 0 \\ 0 & \sqrt{\lambda_2} & 0 & \ldots \\ \ldots & \ldots & \ldots & \ldots \\ 0 & \ldots & 0 & \sqrt{\lambda_{3m}} \end{bmatrix}$$

$$\mathbf{X} = A \cdot D \cdot \mathbf{Z} + \mathbf{u} = A \cdot D \cdot \mathbf{Z}. \tag{12}$$

The *i*th column of matrix A is the *i*th eigenvector corresponding to the *i*th eigenvalue λ_i of the covariance matrix Σ. In this article, the mean value vector $\mathbf{u}$ of $\mathbf{X}$ is $\mathbf{u} = E(\mathbf{X}) = \mathbf{0}$. As a result, each RV X_i is a linear combination of at most 3 m mutually independent standard normal distributions. Consequently, T and S can be rewritten as a function of $\mathbf{Z}$ in (13).

$$T = \max_{1 \leqslant i \leqslant N} (\text{nom}_i + \mathbf{c_i}^T \mathbf{X}) = \max_{1 \leqslant i \leqslant N} (\text{nom}_i + \mathbf{p_i}^T \mathbf{Z})$$

$$S = \min_{k,j} (\text{nom}_k + \mathbf{c_{k,j}}^T \mathbf{X}) = \min_{k,j} (\text{nom}_k + \mathbf{p_{k,j}}^T \mathbf{Z}). \tag{13}$$

The expression after applying PCA has some desirable properties [9]. For any SetupCheck$_i$, the mean is the nominal value nom$_i$, and the variance is the squared sum of coefficients. The covariance between SetupCheck$_i$ and Z_l is the coefficient $p_{i,l}$.

$$\text{SetupCheck}_i = \text{nom}_i + p_{i,1} Z_1 + \cdots + p_{i,3m} Z_{3m}$$

$$E(\text{SetupCheck}_i) = \text{nom}_i$$

$$\text{Var}(\text{SetupCheck}_i) = \sum_{l=1}^{3m} p_{i,l}^2 = \sigma_i^2$$

$$\text{Cov}(\text{SetupCheck}_i, Z_l) = p_{i,l}$$

$$\text{Cor}(\text{SetupCheck}_i, Z_l) = p_{i,l}/\sigma_i. \tag{14}$$

Further, considering another SetupCheck$_j$ ($i \neq j$), it is also easy to get the covariance and correlation between SetupCheck$_i$ and SetupCheck$_j$.

$$\text{SetupCheck}_i = \text{nom}_i + p_{i,1} Z_1 + \cdots + p_{i,3m} Z_{3m}$$

$$\text{SetupCheck}_j = \text{nom}_j + p_{j,1} Z_1 + \cdots + p_{j,3m} Z_{3m}$$

$$\text{Cov}(\text{SetupCheck}_i, \text{SetupCheck}_j) = \sum_{l=1}^{3m} p_{i,l} p_{j,l}$$

$$\text{Cor}(\text{SetupCheck}_i, \text{SetupCheck}_j) = \frac{\sum_{l=1}^{3m} p_{i,l} p_{j,l}}{\sigma_i \sigma_j}. \tag{15}$$

B. Approximation of MAX Operation

Although the maximum of two normal RVs is not a normal RV, Clark[17] provides a way to approximate it into a normal RV. Considering two normal RVs $a \sim N(\mu_a, \sigma_a^2)$ and $b \sim N(\mu_b, \sigma_b^2)$ with a correlation $\rho_{a,b} = Cor(a, b)$, the mean μ_g and the variance σ_g^2 of $g = \max(a, b)$ can be approximated by the followings:

$$\mu_g = \mu_a F_Z(\beta) + \mu_b F_Z(-\beta) + \alpha f_Z(\beta)$$

$$\sigma_g^2 = (\mu_a^2 + \sigma_a^2) F_Z(\beta) + (\mu_b^2 + \sigma_b^2) F_Z(-\beta) + (\mu_a + \mu_b) \alpha f_Z(\beta) - \mu_g^2 \tag{16}$$

where

$$\alpha = \sqrt{\sigma_a^2 + \sigma_b^2 - 2\sigma_a \sigma_b \rho_{a,b}}$$

$$\beta = (\mu_a - \mu_b)/\alpha$$

$$f_Z(x) = \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{x^2}{2}\right)$$

$$F_Z(x) = \int_{-\infty}^{x} \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{x^2}{2}\right) \mathrm{d}x. \tag{17}$$

F_Z and f_Z are the cumulative and PDF of a standard normal distribution. Further, considering another normal RV d with $\rho_{a,d} = Cor(a, d)$ and $\rho_{b,d} = Cor(b, d)$, the correlation $\rho_{g,d} = Cor(g, d)$ between g and d can be obtained by

$$\rho_{g,d} = \frac{\sigma_a \rho_{a,d} F_Z(\beta) + \sigma_b \rho_{b,d} F_Z(-\beta)}{\sigma_g}. \tag{18}$$

Therefore, the normal approximation of two normal RV enables us to represent $g = \max(\text{SetupCheck}_i, \text{SetupCheck}_j)$ as a linear model. SetupCheck$_i$ and SetupCheck$_j$ can be represented

as follows:

$$\text{SetupCheck}_i = \text{nom}_i + \mathbf{p_i}^T\mathbf{Z}$$
$$\text{SetupCheck}_j = \text{nom}_j + \mathbf{p_j}^T\mathbf{Z}. \quad (19)$$

Derived from (14) and (15), their variance and covariance can shown in (20) such that we can compute $\alpha = \sqrt{(\mathbf{p_i}-\mathbf{p_j})^T(\mathbf{p_i}-\mathbf{p_j})}$ in (17)

$$\sigma_i^2 = \sum_{l=1}^{3m} p_{i,l}^2 = \mathbf{p_i}^T\mathbf{p_i}$$
$$\sigma_j^2 = \sum_{l=1}^{3m} p_{j,l}^2 = \mathbf{p_j}^T\mathbf{p_j}$$
$$\sigma_i\sigma_j\rho_{i,j} = \sum_{l=1}^{3m} p_{i,l}p_{j,l} = \mathbf{p_i}^T\mathbf{p_j}. \quad (20)$$

Because $\mathbf{Z}$ is a standard normal vector, we have (21) such that we can compute β in (17)

$$\mu_i = E(\text{SetupCheck}_i) = \text{nom}_i$$
$$\mu_j = E(\text{SetupCheck}_j) = \text{nom}_j. \quad (21)$$

The computation of μ_g and σ_g^2 in (16) is straightforward. If we consider the *l*th element Z_l of $\mathbf{Z}$, (22) rewrites the covariance of SetupCheck_i and Z_l, and the covariance of SetupCheck_j and Z_l in (14)

$$\text{Cov}(\text{SetupCheck}_i, Z_l) = p_{i,l}$$
$$\text{Cov}(\text{SetupCheck}_j, Z_l) = p_{j,l}. \quad (22)$$

If we also define the covariance of g and Z_l as $p_{g,l}$, following (18), we can write (23). Note that the variance of the standard normal distribution Z_l is 1.

$$p_{g,l} = p_{i,l}F_Z(\beta) + p_{j,l}F_Z(-\beta)$$
$$\mathbf{p_g} = \mathbf{p_i}F_Z(\beta) + \mathbf{p_j}F_Z(-\beta). \quad (23)$$

Consequently, we can represent T as a linear model

$$T = \max_{1\leqslant i\leqslant N}(\text{nom}_i + \mathbf{p_i}^T\mathbf{Z})$$
$$= \text{nom}_T + p_{T,1}Z_1 + \cdots + p_{T,3m}Z_{3m}. \quad (24)$$

An algorithm to compute the normal approximation of $T \sim N(\mu_T, \sigma_T^2)$ is presented below (this is similar to the algorithm first presented in [9]). Line 22 is used to eliminate the mismatch between σ_g^2 and s^2. Similarly, this algorithm is also able to compute the normal approximation of $S \sim N(\mu_S, \sigma_S^2)$ since $S = \min_{k,j}(\text{nom}_k + \mathbf{p_{k,j}}^T\mathbf{Z}) = -\max_{k,j}(-\text{nom}_k - \mathbf{p_{k,j}}^T\mathbf{Z})$.

Algorithm 1: Approximate $T = \max_{1\leqslant i\leqslant N}(nom_i + \mathbf{p_i}^T\mathbf{Z})$.

1: $nom_T = nom_1$
2: $\mathbf{p_T} = \mathbf{p_1}$
3: **for** $j = 2, \ldots, N$ **do**
4: $\quad\alpha = \sqrt{(\mathbf{p_T}-\mathbf{p_j})^T(\mathbf{p_T}-\mathbf{p_j})}$
5: $\quad\mu_a = nom_T$
6: $\quad\mu_b = nom_j$
7: $\quad$**if** $\alpha = 0$ **then**
8: $\quad\quad$**if** $\mu_a \geq \mu_b$ **then**
9: $\quad\quad\quad$continue
10: $\quad\quad$**else**
11: $\quad\quad\quad nom_T = nom_j$
12: $\quad\quad\quad \mathbf{p_T} = \mathbf{p_j}$
13: $\quad\quad$**end if**
14: $\quad$**else**
15: $\quad\quad\sigma_a^2 = \mathbf{p_T}^T\mathbf{p_T}$
16: $\quad\quad\sigma_b^2 = \mathbf{p_j}^T\mathbf{p_j}$
17: $\quad\quad\beta = (\mu_a - \mu_b)/\alpha$
18: $\quad\quad\mathbf{p_g} = \mathbf{p_T}F_Z(\beta) + \mathbf{p_j}F_Z(-\beta)$
19: $\quad\quad\mu_g = \mu_a F_Z(\beta) + \mu_b F_Z(-\beta) + \alpha f_Z(\beta)$
20: $\quad\quad\sigma_g^2 = (\mu_a^2+\sigma_a^2)F_Z(\beta) + (\mu_b^2+\sigma_b^2)F_Z(-\beta) + (\mu_a+\mu_b)\alpha f_Z(\beta) - \mu_g^2$
21: $\quad\quad s^2 = \mathbf{p_g}^T\mathbf{p_g}$
22: $\quad\quad\mathbf{p_T} = \mathbf{p_g} * \sqrt{\sigma_g^2/s^2}$
23: $\quad\quad nom_T = \mu_g$
24: $\quad$**end if**
25: **end for**
26: $\mu_T = nom_T$
27: $\sigma_T^2 = \mathbf{p_T}^T\mathbf{p_T}$

C. Path Pruning

The optimum clock period T is the maximum of all SetupCheck_i. Although we can compute the normal approximation of *T*, the path-based SSTA has to traverse all paths, which is not required in reality. Considering two paths with RVs SetupCheck_i and SetupCheck_j, to guarantee that $\text{SetupCheck}_i \geqslant \text{SetupCheck}_j$ holds under process variations, we simply enforce equation (25), thereby, eliminating SetupCheck_j [18].

$$\text{SetupCheck}_i = \max(\text{SetupCheck}_i, \text{SetupCheck}_j). \quad (25)$$

Another interpretation of (25) is that it checks whether the inequality (26) is true or not. If it is true, we can deduce that $\text{SetupCheck}_i \geqslant \text{SetupCheck}_j$ and can be confident to drop SetupCheck_j. However, checking this inequality is not trivial

$$\min(\text{SetupCheck}_i - \text{SetupCheck}_j) \geq 0. \quad (26)$$

1) Basic Technique: Following the definition of SetupCheck_i in Section II, the straightforward and most pessimistic lower bound of inequality (26) is inequality (27). Once the right-hand-side (RHS) value becomes nonnegative, we can guarantee that the left-hand-side (LHS) value is also nonnegative, and therefore, SetupCheck_j can be safely dropped

from consideration.

$$\begin{aligned}\min(&\text{SetupCheck}_i - \text{SetupCheck}_j)\\ &= \min((\text{LP}_i - \text{CP}_i + \text{DP}_i + \text{Setup}_i)\\ &\quad - (\text{LP}_j - \text{CP}_j + \text{DP}_j + \text{Setup}_j))\\ &\geqslant \min(\text{LP}_i) - \max(\text{CP}_i)\\ &\quad - \max(\text{LP}_j) + \min(\text{CP}_j)\\ &\quad + \min(\text{DP}_i) - \max(\text{DP}_j)\\ &\quad + \min(\text{Setup}_i) - \max(\text{Setup}_j). \end{aligned} \tag{27}$$

Notice that $\min(\text{LP}_i)$ is the sum of the minimum propagation delays of all SFQ gates in path LP_i. Moreover, the minimum propagation delay of an SFQ gate can be found under specified process variations (this observation is widely used in STA). Other components in inequality (27) are calculated in the same way.

2) Uni-Data-Path Common Path Pessimism Removal (CPPR) Technique: The difference between LHS and RHS values determines how many noncritical paths are pessimistically included. The launch and capture paths are two clock paths feeding the clock to different clocked SFQ gates on a signal path of interest. Due to the limited fan-out capability of SFQ gates, a splitter-based clock tree is used. A path from the clock source to a clocked SFQ gate is a clock path in this tree. Therefore, any two clock paths certainly have a common path between them. To reduce the difference, a common path pruning approach is typically used to remove the common path between the launch path and the capture path related to a single signal path.

$$\begin{aligned}\min(&\text{SetupCheck}_i - \text{SetupCheck}_j)\\ &\geqslant \min(\text{LP}_i - \text{CP}_i)\\ &\quad - \max(\text{LP}_j - \text{CP}_j)\\ &\quad + \min(\text{DP}_i) - \max(\text{DP}_j)\\ &\quad + \min(\text{Setup}_i) - \max(\text{Setup}_j). \end{aligned} \tag{28}$$

The technique is called CPPR [19]. Because it only involves one data path and associated clock paths, we will call it uni-data-path CPPR or UDP-CPPR. Note that UDP-CPPR is a highly popular technique used in commercial STA tools in order to make worst-case timing estimates less pessimistic. However, the UDP-CPPR is not widely used in SSTA. The reason is that the number of paths in a CMOS circuit is in the worst-case exponential in the number of CMOS gates in the circuit. Consequently, the block-based SSTA, which does not rely on the estimation of paths, has emerged as the dominant strategy for CMOS circuits,. As we discussed earlier, path-based SSTA is more suitable and quite practical for SFQ circuits. We thus adapt basic and UDP-CPPR techniques to prune noncritical paths in SFQ circuits.

3) Multi-Data-Path CPPR: The computation of $\min(\text{SetupCheck}_i - \text{SetupCheck}_j)$ involves two data paths and four clock paths. Following the idea to remove common paths from consideration before doing the computations, we can further reduce pessimism by removing common paths between LP_i and LP_j and between CP_i and CP_j as shown as follows:

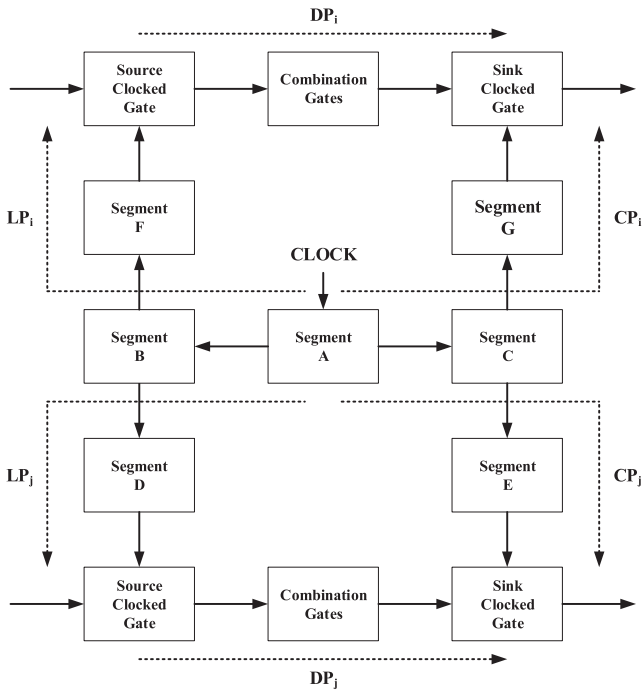


Fig. 2. Model of UDP-CPPR and MDP-CPPR.

$$\begin{aligned}\min(&\text{SetupCheck}_i - \text{SetupCheck}_j)\\ &\geqslant \min((\text{LP}_i - \text{CP}_i) - (\text{LP}_j - \text{CP}_j))\\ &\quad + \min(\text{DP}_i) - \max(\text{DP}_j)\\ &\quad + \min(\text{Setup}_i) - \max(\text{Setup}_j) \end{aligned} \tag{29}$$

A new algorithm to prune noncritical paths, which we call multidata-path CPPR or MDP-CPPR for short, is thus presented. MDP-CPPR needs to compare a pair of data paths. An intuitive way is to compare all pairs to prune noncritical paths, which is in fact unnecessary. A path M with the largest SetupCheck_Max among all SetupCheck_i must exist after path pruning since it determines the worst value of the clock period. Instead of comparing all paths with the path M, we only apply MDP-CPPR on those paths which survive after application of UDP-CPPR ($\text{SetupCheck\_Min} < \max(\text{SetupCheck}_\text{i})$). By comparing $\min(\text{SetupCheck}_M - \text{SetupCheck}_i)$, we can prune all noncritical paths. The complexity of the path pruning algorithm is the same as that of running STA twice. In conclusion, the model-free path pruning approach is efficient and accurate. We can find the normal approximation of T by examining a small number of paths.

Fig. 2 the model of this new and more powerful pruning technique. Equation (28) of UDP-CPPR can remove the common segment A from LP_i and CP_i and from LP_j and CP_j. Equation (29) of MDP-CPPR can further remove the common segment A and B from LP_i and LP_j and the common segment A and C from CP_i and CP_j. Note that a segment could include zero or more nets and splitters.

Algorithm 2: Prune Non-Critical Paths.

$SetupCheck\_Max = -\infty$
$SetupCheck\_Min = +\infty$
$M = 0$
for $i = 1, \ldots, N$ **do**
  $a = \max(LP_i - CP_i)$
  $b = \max(DP_i) + \max(Setup_i)$
  $c = \min(LP_i - CP_i)$
  $d = \min(DP_i) + \min(Setup_i)$
  $\max(SetupCheck_i) = a + b$
  $\min(SetupCheck_i) = c + d$
  if $SetupCheck\_Max < \max(SetupCheck_i)$ **then**
    $SetupCheck\_Max = \max(SetupCheck_i)$
    $SetupCheck\_Min = \min(SetupCheck_i)$
    $M = i$
  end if
end for
Initialize the set $\phi = \{M\}$
for $i = 1, \ldots, N$ **do**
  if $SetupCheck\_Min < \max(SetupCheck_i)$ **then**
    $a = \min((LP_M - CP_M) - (LP_i - CP_i))$
    $b = \min(DP_M) - \max(DP_i)$
    $c = \min(Setup_M) - \max(Setup_i)$
    $\min(SetupCheck_M - SetupCheck_i) = a + b + c$
    if $\min(SetupCheck_M - SetupCheck_i) < 0$ **then**
      add i into ϕ
    end if
  end if
end for

D. Probability Density Function

With the normal approximation of *T* and *S*, the conditional PDF of *T* can be derived. *T* and *S* are correlated since they are both affected by process variations. The correlation $\rho_{T,S}$ can be computed in (15). The joint PDF $f_{T,S}(t, s)$ of two correlated normal distribution *T* and *S* is a well-known concept [20]

$$f_{T,S}(t,s) = h \cdot \exp\left(-\frac{z_T^2 + z_S^2 - 2\rho_{T,S} z_T z_S}{2(1-\rho_{T,S}^2)}\right)$$
$$h = \frac{1}{2\pi\sigma_T\sigma_S\sqrt{1-\rho_{T,S}^2}}, z_T = \frac{t-\mu_T}{\sigma_T}, z_S = \frac{s-\mu_S}{\sigma_S}. \quad (30)$$

Therefore, we can compute $f(T = t, S \geqslant 0)$ by integrating $f_{T,S}(t, s)$.

$$f(T = t, S \geqslant 0) = \int_0^{+\infty} f_{T,S}(t,s)\mathrm{d}s$$
$$= h \cdot \exp(-\frac{z_T^2}{2}) \times \int_0^{+\infty} \exp\left(-\frac{(z_S - \rho_{T,S} z_T)^2}{2(1-\rho_{T,S}^2)}\right)\mathrm{d}s. \quad (31)$$

By converting *T* and *S* into standard normal RVs, we can simplify the representation in (32) so that $f(T = t, S \geqslant 0)$ is purely a function of *t*

$$f(T = t, S \geqslant 0) = \frac{1}{\sigma_T} f_Z\left(\frac{t-\mu_T}{\sigma_T}\right)(1 - F_Z(u))$$
$$u = -\frac{1}{\sqrt{1-\rho_{T,S}^2}}\left(\frac{\mu_S}{\sigma_S} + \rho_{T,S}\frac{t-\mu_T}{\sigma_T}\right). \quad (32)$$

Similarly, we can compute $P(S \geqslant 0)$.

$$P(S \geqslant 0) = 1 - F_Z(v), v = -\frac{\mu_S}{\sigma_S}. \quad (33)$$

In conclusion, we can compute the PDF of clock period *T* as follows:

$$f(T = t|\mathrm{FS}_{\mathrm{circuit}} = 1) = \frac{f(T = t, S \geqslant 0)}{P(S \geqslant 0)}$$
$$= \frac{1}{\sigma_T} f_Z\left(\frac{t-\mu_T}{\sigma_T}\right)\frac{1 - F_Z(u)}{1 - F_Z(v)}. \quad (34)$$

TABLE I
CORRELATION MATRIX Ψ

	Grid 1	Grid 2	Grid 3	Grid 4
Grid 1	1	0.75	0.75	0.5
Grid 2	0.75	1	0.5	0.75
Grid 3	0.75	0.5	1	0.75
Grid 4	0.5	0.75	0.75	1

V. EXPERIMENTAL RESULTS

We used the SFQ gates developed by Stellenbosch University [21]. SFQ gates are wrapped with a receiver for each input and a transmitter for each output and connected by PTLs. References [22] and [23] describe key aspects, and an analysis of the process control monitor (PCM) data for the 350-nm fabrication process SFQ5ee developed at MIT Lincoln Laboratory (MIT LL).

Based on the available PCM data for the SFQ5ee fabrication process, we set $\sigma_R = 1\%$, $\sigma_L = 8\%$, and $\sigma_B = 3\%$ and consider *R*, *L*, and *B* to be mutually independent RVs. To represent spatial correlations Ψ for *R*, *L*, and *B*, respectively, we follow the grid-based model shown in [8] and divide the die area into four equal-size grid cells (with two rows and two columns) as shown in Table I. The ratio p of the local variance to the total variance is set to 0.3 based on data presented in [24].

The covariance matrix for *R*, *L*, and *B*, respectively, can be easily derived as follows:

$$\Sigma_R = (p\Psi + 1 - p)\sigma_R^2$$
$$\Sigma_L = (p\Psi + 1 - p)\sigma_L^2$$
$$\Sigma_B = (p\Psi + 1 - p)\sigma_B^2$$
$$\Sigma = \begin{bmatrix} \Sigma_R & 0 & 0 \\ 0 & \Sigma_L & 0 \\ 0 & 0 & \Sigma_B \end{bmatrix}. \quad (35)$$

The proposed qSSTA is implemented in Python3 and tested on a desktop computer with Intel(R) Core(TM) i7-8700 CPU @

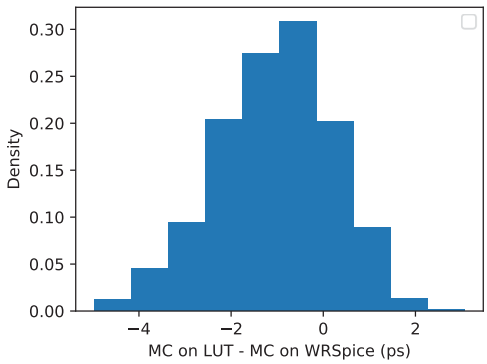


Fig. 3. Histogram of difference between MC on LUTs and MC on WRSpice.

TABLE II
MC COMPARISON ON LUTS AND WRSPICE

	Mean	Std	98% point
MC on LUTs (ps)	25.61	1.68	29.45
MC on WRSpice (ps)	26.76	1.81	30.90
Absolute Error (ps)	-1.15	-0.13	-1.45
Relative Error (%)	-4.30	-7.18	-4.69

TABLE III
REAMAINING PATHS AFTER PATH PRUNING

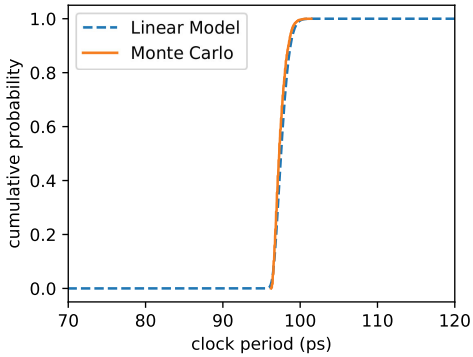
Circuits	# Paths	Basic	UDP-CPPR	MDP-CPPR
MULT4	238	238	238	238
ID4	538	538	538	538
KSA16	568	568	568	568
C499	888	888	666	664
C1355	944	944	856	854
C432	1199	1199	974	970
MULT8	1350	1350	1350	1350
KSA32	1435	1435	1077	1065
C1908	1634	1634	1634	1633
ID8	3184	3184	777	718
C3540	3703	19	12	12
I2C Control*	4740	3083	1555	1421
C7552*	5352	5352	2343	2243
MULT16*	6166	6166	2713	2671
Barrel Shifter*	6620	1926	1055	965
C6288*	7154	7154	3140	3092
Priority Encoder*	13495	3533	602	560
OC Datetime*	16603	684	485	391
ID16*	19136	18898	1314	1209
Sine*	27232	457	266	229
ALU64*	29106	306	223	128

*These circuits are routed by Cadence Innovus™.

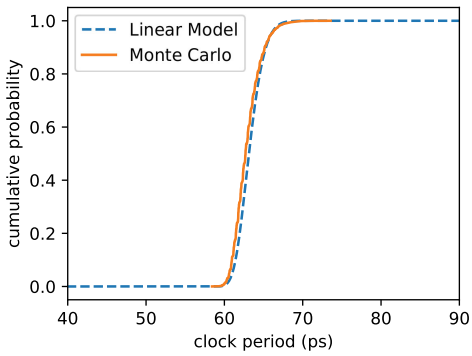
3.20 GHz. The SFQ tool suite [25]–[27] provides a collection of postrouting SFQ circuits, such as the Kogge-Stone adder (KSA), multiplier (MULT), integer divider (ID), some ISCAS benchmark circuits, and a few of the EPFL benchmark circuits [28].

A. Monte Carlo Simulation

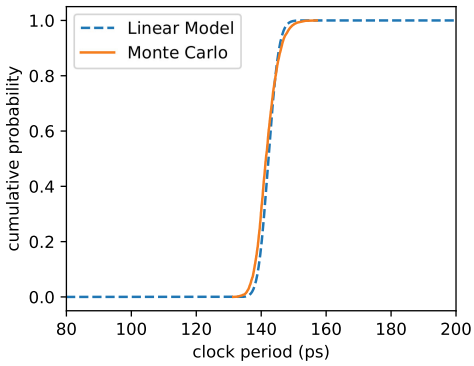
To verify the efficiency of our tool and the quality of our timing results, we used the Monte Carlo (MC) simulation to set the gold standard for comparison. However, Spice simulation for an SFQ circuit through JSIM [29] or WRSpice [30] is time-consuming. To manage the runtime, we extracted the timing characteristics [31], [32](e.g., functional success, propagation



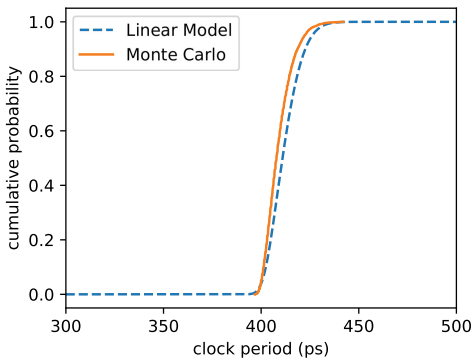
(a) KSA32



(b) MULT8



(c) ID8



(d) C3540

Fig. 4. CDFs of circuits KSA32, MULT8, ID8, and C3540.

TABLE IV
SSTA OF CIRCUITS

Circuits	# of Gates	qSTA (ps)	Monte-Carlo (MC)				qSSTA				|qSSTA-MC|			$\frac{\|qSSTA-MC\|}{MC} \times 100\%$		
			Mean (ps)	Std (ps)	98% Point (ps)	Run time (s)	Mean (ps)	Std (ps)	98% Point (ps)	Run time (s)	Mean (ps)	Std (ps)	98% Point (ps)	Mean	Std	98% Point
KSA16	1,212	107.1	51.19	1.86	55.98	5.17	51.76	1.66	55.57	0.95	0.57	0.20	0.41	1.11	10.75	0.73
KSA32	3,738	144.7	97.43	0.75	99.3	15.38	97.67	0.81	99.56	2.22	0.24	0.06	0.26	0.25	8.00	0.26
MULT4	526	99.4	47.78	2.32	53.28	2.13	48.32	2.20	53.23	0.39	0.54	0.12	0.05	1.13	5.17	0.09
MULT8	3,454	131.4	62.94	1.71	67.15	13.58	63.32	1.59	66.90	2.18	0.38	0.12	0.25	0.60	7.02	0.37
ID4	1,081	117.9	57.35	2.06	62.5	4.53	58.33	1.91	62.92	0.91	0.98	0.15	0.42	1.71	7.28	0.67
ID8	7,337	217.9	141.82	3.33	149.55	28.04	142.33	2.96	148.55	5.02	0.51	0.37	1.00	0.36	11.11	0.67
C432	2,283	138.1	74.84	2.61	81.21	10.11	75.72	2.68	82.16	1.89	0.88	0.07	0.95	1.18	2.68	1.17
C499	2,088	184.8	107.44	3.89	116.96	9.36	108.56	3.65	116.87	1.47	1.12	0.24	0.09	1.04	6.17	0.08
C1355	2,143	164.9	88.90	3.30	96.92	9.76	89.82	3.73	98.28	1.51	0.92	0.43	1.36	1.03	13.03	1.40
C1908	3,801	169.1	100.80	2.14	106.39	16.80	101.53	2.24	106.78	2.58	0.73	0.10	0.39	0.72	4.67	0.37
C3540	7,960	511.2	409.14	6.98	426.48	40.20	411.9	7.86	430.19	5.56	2.76	0.88	3.71	0.67	12.61	0.87

delays, setup time, and hold time) of all SFQ gates as a function of *R*, *L*, and *B* and stored them in 3-D LUTs.

The next thing is to verify the reliability of LUTs. We spent almost ten days finding the minimum workable clock period of an 2-b multiplier (MULT2) with 44 gates for 4000 random samples of physical parameters by using WRSpice on a computer with Intel(R) Xeon(R) CPU E5-2630 v4 @ 2.20 GHz. Fig. 3 shows the histogram of difference between MC on LUTs and MC on WRSpice. Most difference lies in the range $[-4, 2]$.

Table II shows the statistical analysis, where std stands for standard deviation. Absolute error and relative error are defined as $(\text{LUTs} - \text{WRSpice})$ and $(\text{LUTs} - \text{WRSpice})/\text{WRSpice}$ for mean, standard deviation, and the 98% point. The Monte Carlo results on LUTs are close to those on WRSpice and thus reliable for large-scale circuits. Subsequently, instead of using spice-simulation, we could rely on gate-level Monte Carlo simulation utilizing the said LUTs. qSSTA further extract coefficients of the linear models through those LUTs.

B. Path Pruning

Path pruning results are reported in Table III, where the number of pruned paths varies from one circuit to next. Two design steps that greatly affect the efficacy of the path pruning techniques are logic synthesis and clock tree synthesis.

Logic synthesis determines how many splitters are placed on a signal path. In the inequality (29), $\min(\text{DP}_i) - \max(\text{DP}_j)$ is also a factor, affecting whether we can prune a path. However, some SFQ clocked gates need to drive a lot of sink gates, especially in a large-scale circuit. Due the fan-out limitation of SFQ gates (limit is just one), a splitter tree is needed such that a signal path (say DP_i) may include many splitters. Evidently, there also exist many signal paths (say DP_j) that generally include zero or only one splitter. Subsequently, $\min(\text{DP}_i)$ can be much larger than $\max(\text{DP}_j)$, in which case the remaining terms in (29) become negligible. Note also that signal paths, which are not in the above splitter tree, are not critical for determining the clock period (this is the case for example in circuit C3540). A lot of paths can be pruned by just using the basic path pruning.

Clock tree synthesis determines the common path that two clock paths share. If the clock tree of a circuit is well designed, meaning that two clock paths share as much as possible, we can prune a lot of paths by applying UDP-CPPR and MDP-CPPR (like the circuit ID8). When a circuit only has signal paths with small delays, and the clock tree is not well designed, we cannot prune any path (this is the case for example in circuit C1908). Since path pruning is an efficient algorithm, we can always implement it and try to reduce the number of paths for further timing analysis.

When SFQ circuitsare small, the clock trees are also small so that no paths can be pruned. With the increase in the size of SFQ circuits, the common path (between the launch path and the capture path) for a data path becomes important. We can thus prune many noncritical paths by applying the UDP-CPPR technique. When a circuit becomes very large, the common paths involved with two data paths also become prominent, and we can prune more noncritical paths by applying the MDP-CPPR technique on any survived paths after the application of the UDP-CPPR technique. For example, the remaining path count of the 64 b ALU after the MDP-CPPR technique is 42.6% lower than when the UDP-CPPR technique alone is used. For circuits with more than 10 000 paths, the MDP-CPPR technique results in an average of 18.2% fewer remaining paths compared to the path count when using the UDP-CPPR technique.

C. SSTA

The CDFs of MC simulation and qSSTA for KSA32, MULT8, ID8, and C3540 are shown in Fig. 4. The SSTA results (in dashed line) are close to the MC results (in solid line).

Detailed results of SFQ circuits are given in Table IV. The 98% point is the value of t such that $P(T \leqslant t) = 0.98$ and is a safe upper bound for the clock period. On average, the obtained percentage errors are 0.89% for the mean values, 8.04% for the standard deviation, and 0.61% for the 98% point. The error of the standard deviation is acceptable since the largest percentage error is 13.0%. Moreover, the absolute difference of the standard deviation is at most 0.88 ps and on average 0.25 ps.

Experimental results in Section V-A show the reliability of LUTs because Monte-Carlo results done using WRSpice and Monte-Carlo results generated with LUTs are quite close. In conclusion, the SSTA results (which are generated much more efficiently than MC simulation results using WRSpice) are in fact very accurate when compared to the MC simulation results

using WRSpice (with one level of indirection through use of LUTs).

The linear model is the main reason why the performance of the standard deviation is worse than the mean value and the 98% point. The second-order components of Taylor series expansion contain more information about standard deviation, which are not included in the linear model. However, it is quite difficult to solve for the conditional RV of the clock period based on the quadratic model. It can be seen that compared to the 98% point of SSTA result, the included STA results [31] are very pessimistic. The runtime column shows the efficiency of qSSTA.

VI. Conclusion

This article presented an efficient SSTA tool, qSSTA, for SFQ circuits. The correlation model is introduced to represent the intra/inter-die spatial correlations of physical parameters like resistance, inductance, and the area of Josephson junction. Although timing parameters like clock period are nonlinear functions of physical parameters, we could focus on the linear model based on the first-order Taylor expansion. The PCA enables us to map correlated normal distributions into uncorrelated standard normal distributions. Therefore, we can further approximate the distribution of the clock period into a normal distribution through Clark theorem.

Due to the clocked nature of SFQ gates, process variations affect not only timing but also logic function. The distribution of the clock period is valid only when a circuit can work so that it is a conditional RV. With some simplifications, we can achieve the close form of the PDF of the clock period.

Experimental results demonstrate the efficiency and quality of qSSTA. The average percentage errors are 0.89% for the mean values, 8.04% for the standard deviation, and 0.61% for the 98-percentile point, whereas the runtime of qSSTA is 83% faster on average than Monte-Carlo simulations on LUTs. Further research may include the quadratic model to reduce the error of standard deviation.

Acknowledgment

The authors would like to thank Qisheng Fu for source codes used in qSSTA and Naveen Katam, Ting-Ru Lin, Soheil Nazar Shahsavani, and Ghasem Pasandi for the SFQ circuits used in the article. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the ODNI, IARPA, or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation herein.

References

[1] D. Blaauw, K. Chopra, A. Srivastava, and L. Scheffer, "Statistical timing analysis: From basic principles to state of the art," *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, vol. 27, no. 4, pp. 589–607, Apr. 2008.

[2] C. Forzan and D. Pandini, "Statistical static timing analysis: A survey," *Integration*, vol. 42, no. 3, pp. 409–435, 2009.

[3] M. E. Çelik and A. Bozbey, "Statistical timing analysis tool for SFQ cells (STATS)," in *Proc. IEEE 14th Int. Supercond. Electron. Conf.*, Jul. 2013, pp. 1–3.

[4] K. K. Likharev and V. K. Semenov, "RSFQ logic/memory family: A new Josephson-junction technology for sub-terahertz-clock-frequency digital systems," *IEEE Trans. Appl. Supercond.*, vol. 1, no. 1, pp. 3–28, Mar. 1991.

[5] K. Gaj, E. G. Friedman, and M. J. Feldman, "Timing of multi-gigahertz rapid single flux quantum digital circuits," *J. VLSI Signal Process. Syst. Signal, Image Video Technol.*, vol. 16, no. 2–3, pp. 247–276, 1997.

[6] C. J. Fourie, W. J. Perold, and H. R. Gerber, "Complete Monte Carlo model description of lumped-element RSFQ logic circuits," *IEEE Trans. Appl. Supercond.*, vol. 15, no. 2, pp. 384–387, Jun. 2005.

[7] M. Jeffery, W. J. Perold, Z. Wang, and T. Van Duzer, "Monte Carlo optimization of superconducting complementary output switching logic circuits," *IEEE Trans. Appl. Supercond.*, vol. 8, no. 3, pp. 104–119, Sep. 1998.

[8] A. Agarwal, D. Blaauw, and V. Zolotov, "Statistical timing analysis for intra-die process variations with spatial correlations," in *Proc. IEEE/ACM Int. Conf. Comput.-Aided Des.*, 2003, pp. 900–907.

[9] H. Chang and S. S. Sapatnekar, "Statistical timing analysis under spatial correlations," *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, vol. 24, no. 9, pp. 1467–1482, Sep. 2005.

[10] Y. Zhan, A. J. Strojwas, X. Li, L. T. Pileggi, D. Newmark, and M. Sharma, "Correlation-aware statistical timing analysis with non-Gaussian delay distributions," in *Proc. 42nd Annu. Des. Autom. Conf.*, 2005, pp. 77–82.

[11] Z. Feng, P. Li, and Y. Zhan, "Fast second-order statistical static timing analysis using parameter dimension reduction," in *Proc. 44th ACM/IEEE Des. Autom. Conf.*, 2007, pp. 244–249.

[12] S. Ramprasath, M. Vijaykumar, and V. Vasudevan, "A skew-normal canonical model for statistical static timing analysis," *IEEE Trans. Very Large Scale Integration Syst.*, vol. 24, no. 6, pp. 2359–2368, Jun. 2016.

[13] K. Chopra, B. Zhai, D. Blaauw, and D. Sylvester, "A new statistical max operation for propagating skewness in statistical timing analysis," in *Proc. IEEE/ACM Int. Conf. Comput.-Aided Des.*, 2006, pp. 237–243.

[14] A. Inamdar, J. Ren, and D. Amparo, "Improved model-to-hardware correlation for superconductor integrated circuits," *IEEE Trans. Appl. Supercond.*, vol. 25, no. 3, Jun. 2015, Art. no. 1300308.

[15] J. A. Nelder and R. W. Wedderburn, "Generalized linear models," *J. Roy. Stat. Soc.: Ser. A (General)*, vol. 135, no. 3, pp. 370–384, 1972.

[16] D. F. Morrison, L. C. Marshall, and H. L. Sahlin, *Multivariate Statistical Methods*. New York, NY, USA: McGraw-Hill, 1976.

[17] C. E. Clark, "The greatest of a finite set of random variables," *Oper. Res.*, vol. 9, no. 2, pp. 145–162, 1961.

[18] B. Zhang, F. Wang, S. Gupta, and M. Pedram, "A statistical static timing analysis tool for superconducting single-flux-quantum circuits," in *Proc. IEEE Int. Supercond. Electron. Conf.*, 2019, pp. 1–5.

[19] Y.-M. Yang, Y.-W. Chang, and I. H.-R. Jiang, "iTimerC: Common path pessimism removal using effective reduction methods," in *Proc. IEEE/ACM Int. Conf. Comput.-Aided Des.*, 2014, pp. 600–605.

[20] G. Hamedani and M. Tata, "On the determination of the bivariate normal distribution from distributions of linear combinations of the variables," *Amer. Math. Monthly*, vol. 82, no. 9, pp. 913–915, 1975.

[21] SUN Magnetics, "RSFQlib," 2019. [Online]. Available: https://github.com/sunmagnetics/RSFQlib.

[22] S. K. Tolpygo *et al.*, "Superconductor electronics fabrication process with MoNx kinetic inductors and self-shunted Josephson junctions," *IEEE Trans. Appl. Supercond.*, vol. 28, no. 4, Jun. 2018, Art. no. 1100212.

[23] S. K. Tolpygo, V. Bolkhovsky, T. J. Weir, L. M. Johnson, M. A. Gouker, and W. D. Oliver, "Fabrication process and properties of fully-planarized deep-submicron Nb/Al– AlO_x/Nb Josephson junctions for VLSI circuits," *IEEE Trans. Appl. Supercond.*, vol. 25, no. 3, Jun. 2015, Art. no. 1101312.

[24] R. N. Tadros and P. A. Beerel, "A robust and self-adaptive clocking technique for SFQ circuits," *IEEE Trans. Appl. Supercond.*, vol. 28, no. 7, Oct. 2018, Art. no. 1301211.

[25] S. N. Shahsavani, T.-R. Lin, A. Shafaei, C. J. Fourie, and M. Pedram, "An integrated row-based cell placement and interconnect synthesis tool for large SFQ logic circuits," *IEEE Trans. Appl. Supercond.*, vol. 27, no. 4, Jun. 2017, Art. no. 1302008.

[26] G. Pasandi and M. Pedram, "PBMap: A path balancing technology mapping algorithm for single flux quantum logic circuits," *IEEE Trans. Appl. Supercond.*, vol. 29, no. 4, Jun. 2019, Art. no. 1300114.

[27] T. Lin and M. Pedram, "qGDR: A via minimization oriented routing tool for large-scale superconductive single flux quantum circuits," *IEEE Trans. Appl. Supercond.*, vol. 29, no. 7, Oct. 2019, Art. no. 1303412.

[28] L. Amarú, P.-E. Gaillardon, and G. De Micheli, "The EPFL combinational benchmark suite," in *Proc. 24th Int. Workshop Log. Synthesis*, 2015.

[29] E. S. Fang, "A Josephson integrated circuit simulator (JSIM) for superconductive electronics application," in *Extend. Abst. 1989 Int. Supercond. Electron. Conf. (The Jpn. Society Appl. Phys., Tokyo, 1989)*, 1989, pp. 407–410.
[30] Whiteley Research, Inc., 2018. [Online]. Available: http://www.wrcad.com/wrspice.html.
[31] B. Zhang and M. Pedram, "qSTA: A static timing analysis tool for superconducting single-flux-quantum circuits," *IEEE Trans. Appl. Supercond.*, vol. 30, no. 5, Aug. 2020, Art. no. 1700309.
[32] C. J. Fourie, "Extraction of DC-biased SFQ circuit verilog models," *IEEE Trans. Appl. Supercond.*, vol. 28, no. 6, Sep. 2018, Art. no. 1300811.

Bo Zhang received the B.S. degree in electrical engineering from the Nanjing University of Aeronautics and Astronautics, Nanjing, China, in 2014, and the M.S. degree in electrical engineering, in 2017, from the University of Southern California, Los Angeles, CA, USA, where he is currently working toward the Ph.D. degree in electrical engineering with the Ming Hsieh Department of Electrical Engineering.

His research interests include the computer-aided design of digital systems, statistical and static timing analysis, superconducting single-flux-quantum technology, machine learning and VLSI implementation of digital systems.

Mingye Li received the B.S. degree in electrical engineering from the University of Electronic Science and Technology of China, Chengdu, China, in 2017, and the M.S. degree in electrical engineering in 2019 from the University of Southern California, Los Angeles, CA, USA, where he is currently working toward the Ph.D. degree in electrical engineering with the Ming Hsieh Department of Electrical Engineering.

His research interests include the computer-aided design, testing, and VLSI implementation of digital systems and single-flux-quantum technology.

Massoud Pedram received the B.S. degree in electrical engineering from the California Institute of Technology, Pasadena, CA, USA, in 1986 and the M.S. and Ph.D. degrees in electrical engineering and computer sciences from the University of California, Berkeley, Berkeley, CA, USA, in 1989 and 1991, respectively.

In September 1991, he joined the Ming Hsieh Department of Electrical Engineering, University of Southern California, where he currently is the Charles Lee Powell Professor of electrical engineering and computer science with the USC Viterbi School of Engineering.

Dr. Pedram is a recipient of the IEEE Circuits and Systems Society Charles A. Desoer Technical Achievement Award (2015), the Presidential Early Career Award for Scientists and Engineers (1996), and the National Science Foundation's Young Investigator Award (1994). His research has received a number of other awards including two Design Automation Conference Best Paper Awards, a Distinguished Paper Citation from the International Conference on Computer Aided Design, one Best Paper Award of the ACM/IEEE International Symposium on Low Power Design and Electronics, three Best Paper Awards from the International Conference on Computer Design, one Best Paper Award of the IEEE Computer Society Annual Symposium on VLSI, an IEEE TRANSACTIONS ON VLSI SYSTEMS Best Paper Award, and an IEEE Circuits and Systems Society Guillemin-Cauer Award. He was recognized as one of the four DAC Prolific Authors (with 50+ papers) and the DAC Bronze Cited Author at the 50th anniversary of the Design Automation Conference, Austin, TX (2013), received a Frequent Author Award (Top Three Author Award) at the 20th Anniversary Asia and South Pacific Design Automation Conference, Chiba/Tokyo, Japan (2015), and listed as the Second Most Prolific and Second Most Cited Author at the 20th Anniversary International Symposium on Low Power Electronics and Design, Rome, Italy (2015).