Static Timing Analysis (STA) with Timing Bleed:
Certifying Much Higher Performance for Rapid
Single Flux Quantum (RSFQ) Logic

Fangzhou Wang, Bo Zhang, Massoud Pedram and Sandeep Gupta
Department of Electrical Engineering, University of Southern California, Los Angeles, CA, 90089, USA
E-mail: {fangzhow, zhan254, pedram, sandeep}@usc.edu

Abstract. Josephson-junction based technologies, such as RSFQ, are receiving growing investments due to their high speeds and low power.

As the time interval between the data input and the clock becomes smaller, clock-to-Q delay of a flip-flop increases. In conventional timing analysis, setup time is defined as the interval where clock-to-Q delay increases by 10%. This allows the pipeline to be divided into separate stages and STA to be performed independently for each stage.

However, since RSFQ is pipelined at gate-level, setup time is a significant portion of the clock period and makes this timing constraint extremely conservative. Instead, we define setup time based on the probability of causing a logic error and develop a new STA method that allows larger increases in clock-to-Q delay, i.e., timing bleed, whenever the data input arrives late. We present results of simulations for benchmark circuits with process variations to demonstrate that our new method certifies much higher speeds for RSFQ logic.

1. Introduction
There is a revival of interest in superconducting electronics (SCE) as the CMOS technology scaling starts to slow down. SCE logic families, such as Rapid Single-Flux-Quantum (RSFQ), offer high performance at low power consumption [1] [2].

In order to model the timing behavior of the RSFQ cells, researchers have adopted the methods used in CMOS to characterize the timing parameters of the logic cells [3] [4]. Based on the cell characterization results, researchers also developed static timing analysis tools to certify the performance of a RSFQ circuit [5] [6].

However, due to the fundamental differences between CMOS technology and RSFQ technology, adopting the approach used in CMOS may not enable us to fully harness the potential of RSFQ technology. For example, consider the concept of setup time is adopted from CMOS and used in the cell characterization process [4]. The conventional setup time simplifies the timing analysis by allowing the STA to divide a circuit into individual pipeline stages separated by flip-flops, and performing timing analysis on those stages independently. In fact, as we will show ahead, imposing the guard-band needed for this approach will dramatically reduce the performance of the circuit and lose one of the major benefits provided by RSFQ technology.

In addition, due to quantized operation in RSFQ, non-idealities such as process variations are more likely to affect the timing of a cell rather than cause a logic error at the cell [7].
most cases, the effect of process variation is not large enough to cause immediate logic or timing failure at a single cell but the additional delay can propagate from one cell to the next and get accumulated along the path. The accumulated delay effect is typically not small enough to be ignored or efficiently guard-banded.

Therefore, in this paper, we propose a new static timing analysis (STA) approach which relaxes the conventional setup time constraint and performs timing analysis along multi-cycle paths. In the next section, we provide quantitative justification of the performance overhead of the conventional STA approach. Then in Section 3, we describe our proposed new STA approach and present the unique properties of RSFQ cells identified and used by our STA. Experimental results are shown in Section 4 to demonstrate the effectiveness of our proposed STA approach.

2. Motivation

Logic operations in RSFQ technology are based on quantized SFQ pulses. In almost every logic cell, a clock signal is introduced to synchronize the SFQ data pulses. Therefore, each logic cell is a pipeline stage and as a result, circuits in RSFQ technology are highly pipelined. (In many RSFQ cell libraries, e.g., [8] [9], the only combinational cells are interconnects e.g., splitters.)

The delay of the cell, namely the clock-to-Q delay not only depends on the type of the cell, but also depends on the timing interval between data input and clock input. Figure 1 shows the clock-to-Q delay of a D-flip-flop (DFF) under different input delay conditions. On the left side is when the data input arrives much earlier than the clock input (i.e., at least 7.4 ps before the clock), while on the right side is when the data input arrives after the clock. As shown in Figure 1, input can arrive much earlier than the clock, or close to the clock or even a little after the clock, the DFF works correctly in terms of logic value. However, as the input arrives closer to the clock, the time at which the output pulse will appear at the output of the cell will increase, namely the clock-to-Q delay will increase as shown in Figure 1.

A similar behavior is also observed in CMOS technology. In CMOS, when the data arrives sufficiently before the clock, the corresponding clock-to-Q delay is defined as the nominal delay. As the timing interval $\Delta DC$ between the data input and the clock input reduces, the clock-to-Q delay starts to increase. When $\Delta DC$ reduces to $\Delta_{\text{setup}}$, the clock-to-Q delay increases by a predetermined amount (i.e., 10%) compared to the nominal delay, then $\Delta_{\text{setup}}$ is defined as the setup time. In conventional timing analysis approaches, the setup time is the latest time when
Figure 2. The performance overhead of conventional setup time constraint for pipelines with different numbers of levels of combinational logic cells between consecutive flip-flops.

the data input is required to arrive. During the timing analysis, for a given clock period, if the data input is found to arrive after the setup time of even one cell in the entire circuit, then that clock period is determined as insufficient. In this case, the clock period needs to be increased and the whole circuit is required to run at a lower clock frequency.

However, as shown in Figure 1, even if the data input arrives late which causes the clock-to-Q delay to increase by 30%, the cell can still produce the correct output logic value. We say that the additional delay at the input of the cell bleed through the pipeline stage to the next cell. If the next cell can capture the pulse correctly and produce the right value at its output, then the bleeding can continue along the path, and we call this phenomenon timing bleed. If no cell in the circuit creates a logic error based on the timing analysis, then the circuit can operate at the given clock period. The fact that the conventional setup time of some cells are violated does not lead to the failure of the circuit. Therefore, the conventional setup time constraint can be relaxed to improve the performance of the circuit.

In conventional CMOS, there are typically more than ten levels of combinational logic between pipeline stages [10]. The setup time and the clock-to-Q delay of the DFF only constitute a small portion of the overall clock period in conventional CMOS circuits. As shown in Figure 2, as the number of combinational logic between pipeline stages increases, the overhead of imposing the conventional setup time constraint decreases.

However, in RSFQ technology, every logic cell has an internal DFF, the setup time of each cell becomes a major part of the clock period. Therefore, imposing the conventional setup time constraint will dramatically impact the performance of the circuit. In addition to the huge performance overhead, if we use the conventional setup time approach, during post-fabrication testing we need to scan all the pipeline DFFs within the logic cells to verify that timing bleed does not occur at any internal cell. This will also cause an astronomical area overhead due to the required full-scan structure for testing.

In summary, in order to avoid the area overhead and to dramatically increase the yield at desired speed, we propose a static timing analysis approach which allows timing bleed.
3. Proposed approach

3.1. Cell delay characterization with timing bleed

As described in Section 2, when the data input of a cell is delayed, the cell may create a logic error or have a larger than normal clock-to-Q delay (i.e., timing bleed). As shown in Figure 1, we define the soft setup time as the point where the cell starts to experience a larger than normal clock-to-Q delay. We also define the hard setup time as the point where the cell generates a logic error and the circuit would fail. Between the soft setup time and the hard setup time, the cell produces logically correct response but the timing is affected. In order to capture the newly defined setup times as well as the timing behavior of the cell under nominal input conditions and delayed input conditions, we need to characterize the cells in a given cell library under various input conditions.

The cell library used in this paper is a generic RSFQ cell library [11] designed using a standard simulator, i.e., WRspice [12]. All the cells in this library have built-in Passive Transmission Line (PTL) drivers and receivers to achieve maximum isolation between cells when connected together to build logic circuits. Figure 3 shows the simulation environment of the cell characterization process we use [4]. The artificial input signal is converted to realistic SFQ pulse using a DCSFQ convertor, a resistive load is connected to the output of the cell under characterization. The arrival time of the input signal is controlled by adjusting the arrival time of the artificial input signal directly and is measured at the input of the integrated PTL receiver (PTLRX). Similarly, the arrival time of the output signal is measured at the output of the integrated PTL transmitter (PTLTX).

Because process variation has a major impact on timing of the cell [14], we not only characterize the timing behavior of the cells under nominal conditions but also perform the characterization under process variations. For a given cell and a certain amount of variation, we perform a comprehensive set of simulations to capture the timing characteristic of the cell such as the one shown in Figure 1. For cells with two inputs, we not only characterize signal-to-clock timing behavior, but also capture signal-to-signal timing behavior such as the one shown in Figure 4. In order to create a conservative static timing library for a two-input cell, we perform simulations using all possible input patterns and capture the worst case delay among all those patterns.

Our cell characterization results show that the characteristics of timing bleed are different for different types of cells. Since in RSFQ logic-1 is represented by the existence of an SFQ pulse while the absence of an SFQ pulse indicates logic-0, delay is only associated with logic-1. For non-inverting cells, such as AND2, logic-1 at the inputs of the cell can potentially cause logic-1 at the output, therefore, timing bleed can propagate via non-inverting cells. However, for inverting cells (i.e., INV), logic-1 at the input leads to logic-0 at the output, since logic-0 does not cause delay, the propagation of timing bleed is blocked at inverting cells. Detailed description of this phenomenon can be found in [13].
3.2. Circuit model
As shown in Figure 5, RSFQ circuit modules (e.g., a multiplier) have architectural registers at their primary inputs and outputs. Due to the gate-level pipelined nature of RSFQ technology, the logic cells within the circuit module also have internal registers and the functional unit within the module is fully path-balanced [15]. Further, all feedback loops pass via architectural registers, therefore the functional unit is an acyclic logic. In order to avoid the astronomical area overhead described in Section 2, in this paper, we assume that the architectural registers are scanned but the internal registers are not. Therefore, timing bleed is only allowed at internal registers but no timing bleed is allowed at any architectural register. This also allows us to divide a large circuit into smaller functional units (i.e., modules) based on architectural registers and perform timing analysis on each functional unit independently.

3.3. Static timing analysis with timing bleed

3.3.1. Levelization. For a certain cell, if the soft setup time is violated, then the clock-to-Q delay of that cell will increase. Due to the potential increase of clock-to-Q delay, we cannot perform the timing analysis for the second stage in a functional unit unless we complete the analysis for the first stage, therefore, we need to analyze multi-cycle paths. Since in our circuit model, we assume that no timing bleed is allowed at architectural registers, typically primary inputs and primary outputs of functional modules, a multi-cycle path starts at a primary input of a functional module and ends at one of its primary outputs. Therefore, the initial step of our timing analysis is to levelize a given circuit using the algorithm shown in Algorithm 1.

3.3.2. Timing analysis with a given clock period. After levelizing the circuit, the goal of our STA approach is to identify whether a given clock period will cause timing failures in the circuit or not.
Algorithm 1: An outline of levelization algorithm

1. **Phase - Initialization**
   2. Initialize level=unknown to every cell in the given circuit
   3. **Pass - 0**
      4. Assign level=0 to for every cell that contains a primary input
   5. **Pass - i**
      6. repeat
         7. foreach cell c that is assigned a new level j in the previous pass do
            8. foreach fanout f of cell c do
               9. if f is a combinational logic then
                  10. Assign level j to f
               11. end
               12. else if f is a sequential logic then
                  13. Assign level j+1 to f
               14. end
            15. end
         16. until no change is made during the iteration or all the cells that contain a primary output have been levelized;
   7. **Phase - Check**
      8. If all cells in the circuit have been assigned a level and the maximum level does not exceed the total number of cells in the circuit, then the levelization is completed successfully, otherwise, the levelization fails.

All the primary input signals are assumed to be on time (i.e., arrive before the soft setup time of the cells). Then, based on the given clock period, for each cell, we compute the arrival time of the clock signal and calculate the timing interval between the data input and the clock input. For cells with more than one input, we take the worst case among all the inputs.

As described in Section 3.1, the cells are divided into two categories based on their timing characteristics, namely inverting and non-inverting.

For non-inverting cells: Depending on the timing interval between data and clock, the cell (1) may produce the correct logic value without having an increased clock-to-Q delay; (2) may have an increased clock-to-Q delay; or (3) may have a logic error (see Figure 1). If the timing

![Figure 5. The schematic of a RSFQ Full-Adder.](image-url)
interval is larger than the soft setup time, then the cell produces correct output logic value with a nominal clock-to-Q delay (i.e., there is no timing bleed at the output of the cell). If the timing interval is smaller than the hard setup time, then a timing failure is identified at the cell and the timing analysis terminates for the given clock period. If the timing interval is smaller than the soft setup time but larger than the hard setup time, then timing bleed occurs at the cell. In this case, we adjust the clock-to-Q delay of the cell based on our cell characterization results (using a curve similar to Figure 1 for the given cell) and use the adjusted clock-to-Q delay to continue our timing analysis.

For Inverting cells: If the logic-1 at the input is captured correctly by the clock (i.e., arrives before the hard setup time), then a logic-0 is generated at the output of the cell. Since there is no delay associated with logic-0, the signal is always on time. If the logic-1 misses the clock, then a hard setup time violation is detected and the timing failure will cause a logic error at the output of the cell. If the input is a logic-0 then the signal is always on time, hence the output clock-to-Q delay does not need to be adjusted. In summary, due to the unique behavior of inverting cells, the propagation of timing bleed is blocked.

Our timing analysis proceeds level by level, if a timing failure is found at any cell, then the given clock period is not large enough to operate the circuit. If our timing analysis finishes without identifying any timing failure at any cell, then the circuit is guaranteed to provide correct logic response when operated at the given clock period.

3.3.3. Identifying minimum clock period In our timing analysis, we assume that the RSFQ circuits and the cells to have the following properties. (1) As the timing interval between the data input and the clock input decreases, the clock-to-Q delay increases monotonically as shown in Figure 1 and Figure 4. In cases where the monotonicity does not hold due to a slight decrease in clock-to-Q delay, we use a slightly increased clock-to-Q delay for the corresponding timing interval to preserve monotonicity. (2) The clock skew is arbitrary, fixed, and independent of the data values applied to any cell.

For a particular circuit and a given clock period, our STA approach is able to identify if the clock period can cause any timing failures in the circuit. Therefore, by conducting a search within a range of clock periods, our approach is able to identify the minimum clock period for the given circuit. If the current clock period causes a timing failure, then we increase the clock period and perform the timing analysis again. If the current clock period does not lead to any timing failures, then we decrease the clock period to find a smaller one for the given circuit.

Once a certain clock period $c$ is identified as insufficient for the given circuit, then all the clock periods that are smaller than $c$ can also be shown to be insufficient for that circuit. Because of the monotonicity, faster search methods such as binary search can be used to reduce the time complexity of our STA approach.

4. Experimental results
To evaluate our approach, we developed a prototype of our STA tool which computes the minimum clock period with the knowledge of timing bleed. We then compared our new STA prototype with two other STA approaches. One approach is that no timing bleed is allowed at any cell, the other approach uses the conventional setup time method where timing bleed is only allowed up to 10% increase in the clock-to-Q delay. Table 1 shows the minimum clock period (ps) identified by all three timing analysis methods for 19 RSFQ benchmark circuits [16].

The result shows that the conventional STA approach indeed improves the performance relative to the approach based on no timing bleed (i.e., based on the soft setup time), but our new STA tool can certify circuits to operate at an even higher frequency.

The result also shows that our approach offers lower benefits when there is a deep multi-stage combinational logic in the worst case path between pipeline stages. However, the depth of the
combinational logic can be optimized during the synthesis process. For example, in Table 1, C499 and C499_imp_sp (i.e., improved by redesigning splitter tree) are functionally identical but synthesized in different ways. The results show that when the maximum depth of combinational logic reduces then the benefit of our approach increases.

| Circuit name | Maximum number of combinational logic between pipeline stages | Allowing timing bleed | Strick setup time (10% timing bleed) | Strick setup time (0% timing bleed) | Improvement of allowing timing bleed compared to conventional setup time |
|--------------|---------------------------------------------------------------|-----------------------|-------------------------------------|-------------------------------------|---------------------------------------------------------------------|
| FA           | 2                                                             | 14.6                  | 17.3                                | 23                                  | 18.49%                                                              |
| KSA4         | 2                                                             | 20                    | 23                                  | 28.6                                | 15.00%                                                              |
| KSA8         | 3                                                             | 23.8                  | 27.2                                | 33.9                                | 14.29%                                                              |
| KSA16        | 3                                                             | 23.8                  | 27.2                                | 33.9                                | 14.29%                                                              |
| KSA32        | 3                                                             | 25.3                  | 28.3                                | 33.9                                | 11.86%                                                              |
| MULT4        | 3                                                             | 24.3                  | 27.5                                | 34.2                                | 13.17%                                                              |
| MULT8        | 2                                                             | 18.8                  | 22                                  | 28.6                                | 17.02%                                                              |
| MULT16       | 2                                                             | 18.8                  | 22                                  | 28.6                                | 17.02%                                                              |
| DIV4         | 4                                                             | 32.9                  | 37                                  | 43.4                                | 12.46%                                                              |
| DIV8         | 8                                                             | 53.4                  | 57.3                                | 63.8                                | 7.30%                                                               |
| DIV16        | 15                                                            | 95.6                  | 99.7                                | 106                                 | 4.29%                                                               |
| C432         | 14                                                            | 86.6                  | 90                                  | 96.7                                | 3.93%                                                               |
| C499         | 15                                                            | 93.3                  | 97.2                                | 103.7                               | 4.18%                                                               |
| C499_imp_sp  | 7                                                             | 41.1                  | 43.9                                | 51.2                                | 6.81%                                                               |
| C880         | 7                                                             | 46.6                  | 50                                  | 56.7                                | 7.30%                                                               |
| C1355        | 15                                                            | 93.4                  | 97.3                                | 103.8                               | 4.18%                                                               |
| C1908        | 13                                                            | 80.8                  | 84.2                                | 90.9                                | 4.21%                                                               |
| C3540        | 25                                                            | 149.6                 | 153                                 | 159.7                               | 2.27%                                                               |
| C6288        | 12                                                            | 69.6                  | 72.4                                | 79.7                                | 4.02%                                                               |

We then compared the above three different approaches using Monte Carlo simulations on these benchmark circuits. For each circuit, we created 500 Monte Carlo instances under process variations and computed the minimum clock periods for each instance using the no timing bleed STA tool, conventional CMOS STA tool, and our new prototype STA tool.

Figure 6 shows the distribution of the minimum clock periods for the 500 instances for each benchmark circuit. Due to the space limit, only 12 benchmark circuits are shown here. The red bars show the minimum clock period for various instances produced by the no timing bleed STA approach. The black bars show the clock periods computed using the conventional CMOS setup time approach. The blue bars show that much lower clock periods are identified by our new prototype STA tool which allows timing bleed at internal cells.

For each benchmark circuit, the comparison of the three distributions clearly shows that by allowing timing bleed at internal cells, we can greatly improve the performance of the circuit.

5. Conclusion
In this paper, we propose a new STA approach along with a new method for characterizing timing behaviors of RSFQ logic cells. By relaxing the setup time constraints on internal registers of the logic cells and properly accounting for their effects of timing at the subsequent cell using the concept of timing bleed, our STA is able to certify much higher performance for RSFQ logic,
Figure 6. Distribution of minimum clock periods for different benchmark circuits.

and hence preserve the key advantage of RSFO, namely high speed. Experimental simulation results show that for a given circuit, our STA prototype identifies a minimum clock period that
can dramatically improve the performance of the circuit.

Acknowledgement
The research is based upon work supported by the Office of the Director of National Intelligence (ODNI), Intelligence Advanced Research Projects Activity (IARPA), via the U.S. Army Research Office grant W911NF-17-1-0120. 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] Likharev, K. K. and Semenov, V. K. [1991], ‘Rsfq logic/memory family: A new josephson-junction technology for sub-terahertz-clock-frequency digital systems’, *IEEE Transactions on Applied Superconductivity*.
[2] Nakajima, K., Onodera, Y. and Ogawa, Y. [1976], ‘Logic design of josephson network’, *Journal of Applied Physics* 47(4), 1620–1627.
[3] Lin, J.-C. and Semenov, V. [1995], ‘Timing circuits for rsfq digital systems’, *IEEE transactions on applied superconductivity* 5(3), 3472–3477.
[4] Fourie, C. J. [2018], ‘Extraction of dc-biased sfq circuit verilog models’, *IEEE Transactions on Applied Superconductivity* 28(6), 1–11.
[5] Delport, J. A. and Fourie, C. J. [2018], ‘A static timing analysis tool for rsfq and ersfq superconducting digital circuit applications’, *IEEE Transactions on Applied Superconductivity* 28(5), 1–5.
[6] Zhang, B., Wang, F., Gupta, S. and Pedram, M. [2019], A statistical static timing analysis tool for superconducting single-flux-quantum circuits, in ‘2019 IEEE International Superconductive Electronics Conference (ISEC)’, IEEE, pp. 1–5.
[7] Wang, F. and Gupta, S. [2019], Timing verification for rapid single-flux-quantum (rsfq) logic: New paradigm and models, in ‘2019 IEEE International Superconductive Electronics Conference (ISEC)’, IEEE, pp. 1–3.
[8] Yorozu, S., Kameda, Y., Terai, H., Fujimaki, A., Yamada, T. and Tahara, S. [2002], ‘A single flux quantum standard logic cell library’, *Physica C: Superconductivity* 378, 1471–1474.
[9] Maezawa, M., Hirayama, F. and Suzuki, M. [2004], ‘Design and fabrication of rsfq cell library for middle-scale applications’, *Physica C: Superconductivity* 412, 1591–1596.
[10] Hrishikesh, M., Jouppi, N. P., Farkas, K. I., Burger, D., Keckler, S. W. and Shivakumar, P. [2002], The optimal logic depth per pipeline stage is 6 to 8 fo4 inverter delays, in ‘Proceedings 29th Annual International Symposium on Computer Architecture’, IEEE, pp. 14–24.
[11] Coenrad Fourie, L. S. [2018], ‘Generic parameterized cell library for logic synthesis with dc-biased rsfq logic’. URL: [https://github.com/sunmagnetics/RSFQLib](https://github.com/sunmagnetics/RSFQLib)
[12] Whiteley, S. R. [1996], ‘WRspice circuit simulator’. URL: [http://www.wrcad.com/wrspice.html](http://www.wrcad.com/wrspice.html)
[13] Wang, F. and Gupta, S. K. [2020], ‘An effective and efficient automatic test pattern generation (atpg) paradigm for certifying performance of rsfq circuits’, *IEEE Transactions on Applied Superconductivity* 30(5), 1–11.
[14] Wang, F. and Gupta, S. [2019], Automatic test pattern generation for timing verification and delay testing of rsfq circuits, in ‘2019 IEEE 37th VLSI Test Symposium (VTS)’, IEEE, pp. 1–6.

[15] Pasandi, G. and Pedram, M. [2018], ‘Pbmap: A path balancing technology mapping algorithm for single flux quantum logic circuits’, *IEEE Transactions on Applied Superconductivity* **29**(4), 1–14.

[16] Katam, N., Shahsavani, S. N., Lin, T.-R., Pasandi, G., Shafaei, A. and Pedram, M. [2017], ‘Sport lab sfq logic circuit benchmark suite’, *Univ. Southern California, Los Angeles, CA, USA, Tech. Rep*. }