An Integrated Approach towards VLSI Implementation of SFQ Logic using Standard Cell Library and Commercial Tool Suite

Sukanya S Meher[1], M Eren Çelik[1], Jushya Ravi[2], Amol Inamdar[3] and Deepnarayan Gupta[4]

[1]Hypres Inc,175 Clearbrook Rd Ste 141, Elmsford, NY 10523, USA
[2]Formerly with Hypres Inc, now with Synopsys Inc, Hyderabad 500081, India
[3]Formerly with Hypres Inc, now with AMD, Fishkill, NY 12524, USA
[4]Formerly with Hypres Inc

E-mail: smeher@hypres.com

Abstract. The semiconductor industry seeks energy-efficient alternatives as Moore's law nears its limits. The Single Flux Quantum (SFQ) integrated circuits (ICs) using thousands of niobium Josephson junctions (JJs) and operating at 4 K show great promise for digital computing circuits at high speed (>20 GHz) and low power (a few nW per junction). The leading logic families are Rapid Single Flux Quantum (RSFQ), and its energy-efficient variant (ERSFQ). IARPA's SuperTools program aims to develop integrated design tools for superconductor electronics, targeting SFQ and Adiabatic Quantum-Flux-Parametron (AQFP) logic families. This paper presents a passive transmission line (PTL) based standard cell library for SFQ logic, designed with Synopsys Electronic Design Automation (EDA) software tools for MIT-LL 100µA/µm^2 SFQ5ee fab node. The dual RSFQ/ERSFQ standard cell library facilitates seamless integration of SFQ RTL-to-GDS design flow with Synopsys Fusion Compiler, an automated design tool. The SFQ RTL-to-GDS flow entails logic synthesis, checking, placement, clock synthesis, and routing. Row-based placement for library cells and H-tree clock tree structures are employed. Fusion Compiler's effectiveness is validated with Hypres designs such as finite impulse response (FIR) filters, scalable multiply-accumulate (MAC) units, and memory arrays, comparing single and dual clocking schemes. The synergy between Hypres and Synopsys achieves a milestone by demonstrating the design of a digital superconducting circuit with over 10 million JJs, facilitated by a fully automated design tool for the first time. Challenges in very large-scale SFQ scaling are also discussed.

1. Introduction

The demand for low-power and energy-efficient alternatives to traditional semiconductor circuits is escalating, particularly with the limitations of Moore's law [1]. A promising solution is the Single Flux Quantum (SFQ) superconducting logic family, encompassing Rapid Single Flux Quantum (RSFQ) [2] and its energy-efficient variant (ERSFQ) [3]. SFQ demonstrates ultra-low power consumption and high-speed operation, gaining increased attention in superconducting electronics [1][4]. Despite its advantages, scaling SFQ designs poses challenges [5], with one of the significant obstacles being the absence of robust industrial-strength design tools for SFQ logic [6]. While specialized tools like PSCAN [7], PSCAN2 [8], WRspice [9], JoSIM [10], and XicTools [11] have been available for superconductor electronics (SCE) design, there is a growing need to seamlessly integrate and analyze

SCE designs across platforms using robust electronic design automation (EDA) tools. Successful fabrication of smaller SCE chips has been achieved, but unlocking the full potential of SCE requires scaling to larger sizes and increasing complexity. It requires advancements in manufacturing techniques [12] and improvements in methods and tools for SCE circuit design.

As part of the IARPA SuperTools program [13], Synopsys collaborated with Hypres to adapt its EDA tools for superconductor electronics (SCE) operating at cryogenic temperatures. The collaboration involved modifying existing models and tools for SFQ circuits and improving design methodologies [14]. Under SuperTools, Synopsys transformed its semiconductor-based design infrastructure to cater to superconductor integrated circuits [15] for both SFQ and Adiabatic Quantum-Flux-Parametron (AQFP) [16][17] logic. Within the expansive domain of SuperTools, a parallel initiative was undertaken by the ColdFlux team in collaboration with diverse academic groups. Their endeavor was dedicated to the advancement of open-source S-EDA tools detailed in [18]-[20].

Standard cell libraries are an important component in designing very large-scale integrated circuits. While early cell libraries [21]-[23] were designed for manual or semi-manual SCE circuit design, modern SCE libraries focus on automated cell placement and interconnect routing tools, adhering to specific routing rules of EDA tools. During the initial years of the SuperTools program, Hypres developed a dual RSFQ/ERSFQ cell library [24][25] for the MIT-LL SFQ5ee 100μm/μm^2 fabrication process [26] that was used to design and experimentally validate circuits targeting high-speed operation [27][28]. The library supported abutment and signal routing primarily through Josephson transmission lines (JTLs) and passive transmission lines (PTLs) when needed. However, the use of JTLs are not suitable for large-scale circuits due to timing delays caused by an abundance of junctions. To solve this problem, the library was modified to include drivers and receivers for PTLs within cells, ensuring signal routing using PTLs exclusively. This led to reduced delay per unit length compared to JTLs [29]. The PTL-based library facilitated synthesis and automated placement and routing tools like Synopsys Fusion Compiler.

In the early stages of Fusion Compiler development for SCE circuits, it was found that SFQ logic cells which are clocked, required path-balancing D flip-flops (DFFs) for correct circuit operation. However, it resulted in increased gate count, and thus chip area, and power dissipation. To address this issue, Hypres collaborated with Synopsys to introduce additional cells based on non-destructive DFFs in the PTL-based cell library. The cells supported a dual-clocking scheme to reduce buffer requirements for path balancing in pipeline stages. Despite the array of EDA modules available, the paper focuses solely on the design of the PTL-based standard cell library and its integration into the SFQ RTL-to-GDS flow using the Synopsys tool suite.

The paper is organized as follows: Section 2 details the development of the PTL-based standard cell library, encompassing optimization methodology, layout architecture, timing model, and characterization via the Synopsys integrated platform. Section 3 outlines the SFQ RTL-to-GDS flow utilizing Synopsys EDA tools, validated with Hypres designs. Section 4 engages in a discussion and outlines future opportunities. Finally, Section 5 concludes the paper.

2. Development of PTL-based Standard Cell Library in Synopsys Environment

2.1. Design of PTL-based cell library

Figure 1 illustrates the design flow of the SFQ standard cell library using the Synopsys tool suite. The library cell schematics are made using Synopsys Custom Compiler—a robust editor seamlessly integrated with simulators such as PrimeSim HSPICE for spice simulation and FineSim VCS for digital HDL simulation. HSPICE has been enhanced to support SFQ circuit models and improve usability for SCE design. It incorporates features like a Gaussian pulse source, a quantum phase probe, and temperature effects as described in [30]. For RSFQ or ERSFQ logic, the simulation includes support for a configurable current source for appropriate biasing. Figure 2 presents a PTL-based D flip-flop with schematic, HSPICE simulated waveform, and digital HDL simulated waveform using Synopsys PrimeWave design environment.

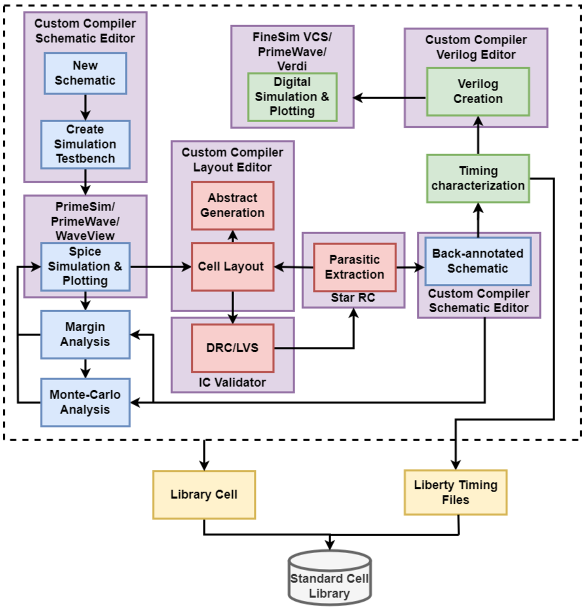

Custom Compiler Schematic Editor
New Schematic
Create Simulation Testbench
PrimeSim/ PrimeWave/ WaveView
Spice Simulation & Plotting
Margin Analysis
Monte-Carlo Analysis
FineSim VCS/ PrimeWave/ Verdi
Digital Simulation & Plotting
Custom Compiler Verilog Editor
Verilog Creation
Custom Compiler Layout Editor
Abstract Generation
Cell Layout
DRC/LVS
IC Validator
Parasitic Extraction
Star RC
Timing characterization
Back-annotated Schematic
Custom Compiler Schematic Editor
Library Cell
Liberty Timing Files
Standard Cell Library

Figure 1. Design flow of SFQ standard cell library using Synopsys tool suite.

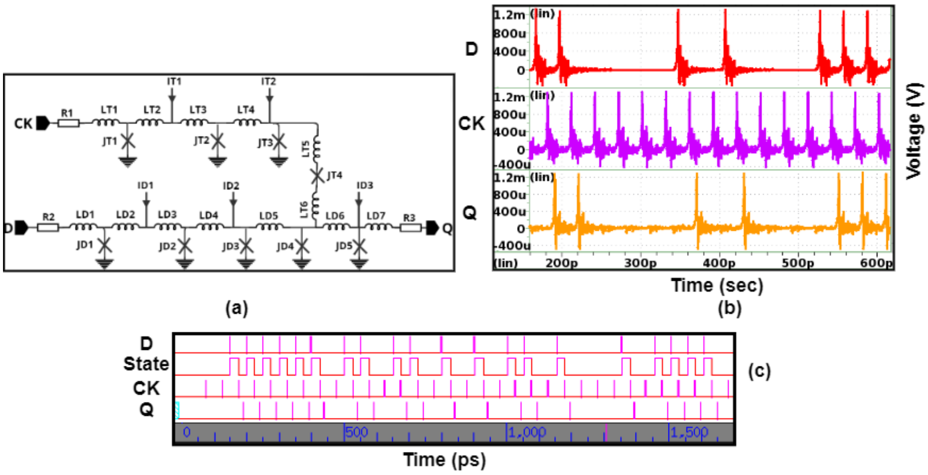


Figure 2. (a) Schematic of a PTL-based D flip-flop (DFF) cell (b) HSPICE waveform showing the data (D), clock (CK), and output (Q) of D flip-flop cell in Synopsys WaveView. (c) Verilog digital HDL waveform showing the data (D), internal state (State) showing the presence of '1' and '0', clock (CK), and output (Q) of D flip-flop cell in Synopsys Verdi.

Another factor to consider while designing SFQ circuits is margin analysis which is crucial for its stable operation. Global scaling parameters are optimized to achieve margins meeting or exceeding ±20% for critical current (XJ), ±30% for bias current (XI), and ±40% for inductances (XL) [24][25]. In the PTL-based cell library, overall margins depend on the integrated driver and receiver margins as well. While the XI margin of an independent receiver is ±30%, it decreases significantly at the PTL resonant frequency. To address this, a variant of the receiver with an additional damping resistor is included for the cell library (see [31] for details). The value of damping resistor is chosen to minimize margin degradation at the resonant frequency. However, it can deteriorate the receiver margins and the overall margin for PTL-based library cells, particularly for global bias current (XI). Thus, the optimization is performed for the best-case scenario. Table 1 presents an optimized D flip-flop as an example for global scaling parameter margins, extracted using HSPICE's ".measure tran_cont" commands for logic validation of SFQ circuits [30].

For further optimization of the circuit, a semiconductor industry-standard method is incorporated. It involves optimization across multiple process corners with Monte-Carlo (MC) statistical variations, considering local fabrication process variations [25]. For corner definition see Table 5 in Appendix. Gaussian distribution with a standard deviation (σ) of 2% and 3% is applied to the critical current (I_C) of Josephson junctions (JJs), bias current (I_b), inductors (L), and junction shunt resistance (R) for mismatch variations, as elaborated in [25]. Results are presented in Table 2 and satisfy more than 90% pass rate. During the development of the Synopsys SCE optimizer tool, HSPICE-AVA (Advanced Variability Analysis) [32], optimization was performed using a third-party tool with Hypres in-built infrastructure. The results were then compared with those from Synopsys tools showing a perfect match.

Table 1. Margin analysis for PTL-based D flip-flop cell.

Parameter	Minimum margin limit	Maximum margin limit
XI	-18.59%	31.71%
XJ	-25.15%	25.15%
XL	-44.84%	70.00%

Table 2. Monte-Carlo with process corners for PTL-based D flip-flop cell.

Corners	Pass rate with 2% variations	Pass rate with 3% variations
Nominal	95%	92%
Fast	96%	94%
FastFast	97%	98%
Slow	93%	97%
SlowSlow	94%	98%
Fast_dR_0.02	96%	96%
Fast_dR_0.04	96%	97%
Slow_dR_0.02	95%	98%
Slow_dR_0.04	99%	89%
XI_1.1	95%	95%
XI_0.9	93%	94%

2.2. Layout architecture of PTL-based cell library

The layout methodology adopts a dual RSFQ/ERSFQ cell library approach, supporting both logic families [24]. The layout is based on the Phase 1 SuperTools cell library [25], with core, bias, and subterranean sub-cells. The core sub-cell, containing the logic network between the ground plane (M4) and sky plane (M7), is common to both RSFQ and ERSFQ cells and includes an additional ground plane (M1 added for isolation). The bias sub-cell, detailed in [24] and implemented in [25], is specific to logic families. The subterranean sub-cell, common to all cells, uses metal layers M0-M4 for signal routing and power rails. Blockage layers in M3, M5, and M6 are strategically used to limit routing by automated tools in the active circuitry of cells. The MIT-LL SFQ5ee metal stack (Figure 3(a)) is used for the layout of the standard cell library.

A row-based placement methodology with channel routing is adopted for SFQ circuits designed using automated place and route tools. The cell width is function-dependent, maintaining a consistent 40 µm height with a 20 µm grid for standard rows (see Figure 3(b)). Dedicated moat slots along vertical edges double their size when adjacent or filler cells are placed during automated routing. The library cell includes two horizontal tracks for signal routing, accommodating PTLs with an 8 Ω impedance, and two horizontal bias lines. Input/output (I/O) pins are positioned along the x-axis at odd multiples of 5 µm, centered at the cell height (Figure 3(b)). The standardized height cell library, with

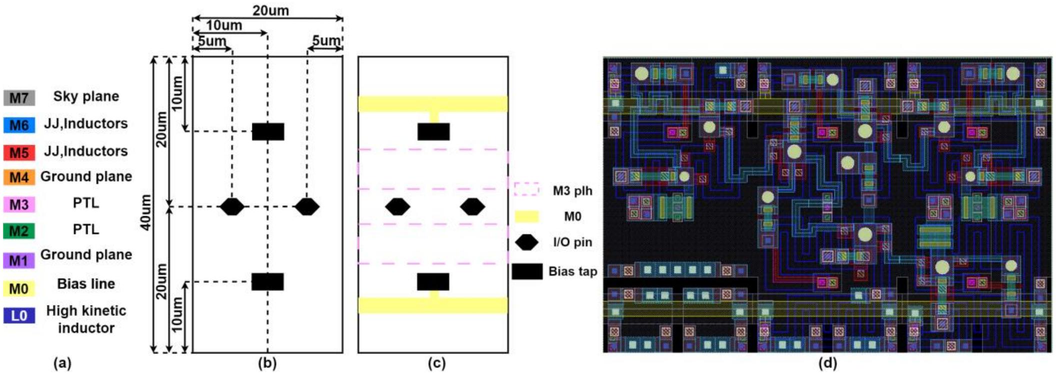


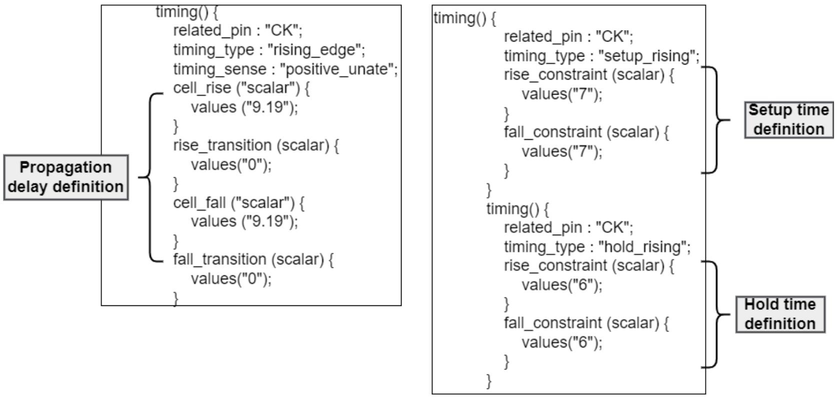
Figure 3. (a) Metal stack of MIT-LL SFQ5ee process showing layers as used in the design of library cell. (b) Layout dimensions and locations of I/O pins and bias taps in library cell. (c) Layout architecture of library cell with bias lines in M0 layer, PTL tracks in M3 placeholder layer, I/O pins, and bias taps highlighted. (d) Layout of a ERSFQ D flip-flop cell with dimension 60 µm × 40 µm as an example.

designated tracks for signals and power lines (Figure 3(c)), streamlines automated place and route processes.

After cell layout, specialized tools like InductEx [33] and Synopsys STAR-RC are used to extract inductor values and parasitics. Extracted values are back-annotated to the original schematic, and performance metrics are compared. Post-layout optimization is done if any degradation is observed. For layout correctness, Synopsys Custom Compiler integrates with the IC Validator (ICV) tool (highlighted in Figure 1), conducting design rule checks (DRC) and layout-vs-schematic (LVS) checks as per MIT-LL SFQ5ee rules. The inductor and resistance recognition layers in the process design kit (PDK) aid LVS. I/O pins, power source (VDD), and ground (GND) pins assist LVS checks and automated place and route tools. Once the cell layout is complete, an abstract view of the library cell is generated which contains the cell boundary, I/O pin information, and metal blockage layers for automatic placement and routing tools. The PTL-based ERSFQ DFF cell layout in the library is illustrated in Figure 3(d). The RSFQ and ERSFQ library includes 20 cells each as presented in Table 6 under the Appendix.

2.3. Timing model and characterization of PTL-based cell library

Each library cell includes a Verilog view to model its behavior for digital HDL simulation. Unlike CMOS, SFQ technology uses voltage pulses to represent logic, with timing details included in behavioral modeling. Typically, the propagation delay of SFQ gates depends on neighboring cell loads. To address this, we use automated scripts for the timing characterization of cells which are detailed in [34] and implemented in [35]. For the PTL-based cell library, we simplified the characterization since PTL connections are used as the signal routing method, thus eliminating delay


```
timing() {
    related_pin : "CK";
    timing_type : "rising_edge";
    timing_sense : "positive_unate";
    cell_rise ("scalar") {
        values ("9.19");
    }
    rise_transition (scalar) {
        values("0");
    }
    cell_fall ("scalar") {
        values ("9.19");
    }
    fall_transition (scalar) {
        values("0");
    }
```

```
timing() {
        related_pin : "CK";
        timing_type : "setup_rising";
        rise_constraint (scalar) {
            values("7");
        }
        fall_constraint (scalar) {
            values("7");
        }
    }
    timing() {
        related_pin : "CK";
        timing_type : "hold_rising";
        rise_constraint (scalar) {
            values("6");
        }
        fall_constraint (scalar) {
            values("6");
        }
    }
```

Figure 4. Section of Liberty file for D flip-flop cell showing the timing constraints definition.

dependence on neighboring cells. Now, every standard library cell has PTL drivers for outputs and PTL receivers for inputs, ensuring consistent delay irrespective of the load, and dependent only on the junction count of the cell.

We not only analyze propagation delay but also capture timing constraints concerning clock and data relationships such as clock-after-data (setup time) and data-after-clock (hold time). These constraints ensure data stability around clock transitions and are critical for reliable circuit operation. These specifications are encapsulated in the Liberty file, a standard timing model file frequently used in the semiconductor industry, detailing timing, power, area, and other logic cell performance characteristics The design flow is illustrated in Figure 1. We streamlined the Liberty format for SFQ library cells, tailoring it from [24] to satisfy Synopsys Library Compiler standards. Figure 4 exhibits a description of the PTL-based SFQ D flip-flop cell in Liberty format, showcasing constraints such as propagation delay, setup, and hold time. Using the behavioral-level HDL descriptions and static timing analysis via Liberty files, we can efficiently analyze the functionality of large SFQ digital circuits with over ten thousand junctions. This is verified with the digital simulation of a 64-bit arithmetic logic unit (ALU) consisting of more than 90K junctions as presented in [35].

3. SFQ RTL-to-GDS flow

3.1. Overview of Top-Down Approach

In the SFQ RTL-to-GDS flow, a master TCL script initiates subsidiary scripts for all the tasks. As highlighted in Figure 5, Synopsys Library Compiler compiles the SFQ cell library by taking physical information, logical/timing data in Liberty files (.lib) format, and tech files. Then Synopsys Fusion Compiler uses this compiled library object for synthesis. During synthesis, the tool checks the syntax of RTL circuit behavior, converts its boolean expressions, maps them to target library cells, and applies timing and physical constraints. The synthesis process results in a gate-level netlist after performing logic optimization. Formality ensures logic equivalence check between RTL and the gate-level netlist, producing outputs like quality of results (QoR), and timing reports.

The gate-level netlist generated during synthesis serves as an input for automated placement and routing (P&R). Figure 5 illustrates the steps involved in placement and routing. During floor planning, the design is partitioned, I/O pins are assigned, and blockages are set to manage congestion. The row-based placement method organizes standard cells in rows, and filler cells fill the gaps ensuring power rail continuity. Clock tree synthesis (CTS) evenly distributes the clock through an H-tree structure for reduced skew. Global routing, track assignment, and detailed routing follow, optimizing for timing, and DRC violations. The user specifies the floorplan's width and height and the number of clock taps for the digital design, similar to the CMOS domain. Asymmetric PTL configurations in M1-M3-M4

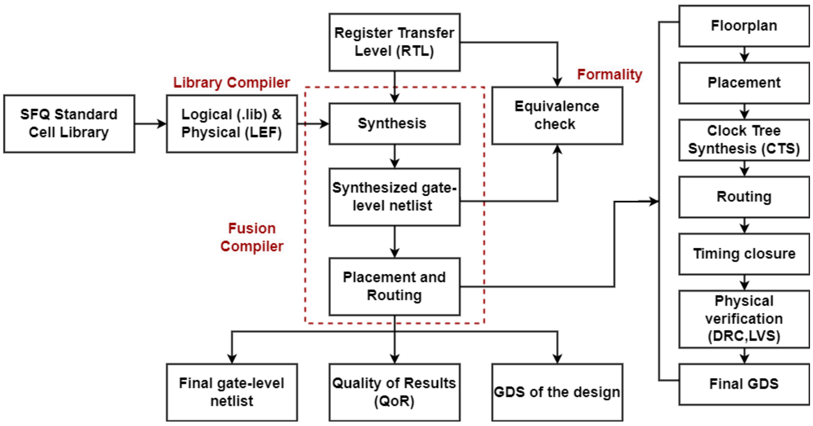


Figure 5. RTL-to-GDS flow for SFQ digital circuit design highlighting the tools used from Synopsys tool suite. The rightmost part highlights the placement and routing flow of Fusion Compiler.

and M1-M2-M4, are employed for horizontal and vertical routing respectively, with optional configurations in M4-M5/M6-M7 to reduce congestion and increase density. The tool performs physical verification (DRC, LVS) to identify design rule violations and shorts, and generates the final geometric layout of the SCE digital circuit in graphic data system (GDS) format. The post-synthesis Verilog netlist is generated, accompanied by reports on timing, cell usage, utilization, and wiring. A detailed algorithmic explanation of the Synopsys tool suite for the SFQ RTL-to-GDS flow is explained in [36].

3.2. Validation of Fusion Compiler using Hypres Designs

In the initial phase of the SuperTools program, Synopsys integrated a uniform global clock for all synchronous cells called a single-clocking scheme into the Fusion Compiler [36]. However, since SFQ gates are clocked, path-balancing DFFs are inserted to match the input signal transmission delay in terms of clock cycles. This method of path balancing caused several issues, such as an increase in power consumption, gate count, and chip area due to heavy DFF usage. This is illustrated with an example in Figure 6. To address these concerns, Synopsys introduced the dual clocking scheme, a method also used in CMOS involving two global clocks (Figure 7(a)), and implemented using a special non-destructive DFF called NDDFF2 for the SFQ domain. This synchronous cell, based on two set-reset flip-flops with non-destructive read-out (RSN) cells as depicted in Figure 7(b) significantly reduces the need for DFFs and mitigates associated challenges. NDDFF2 uses either a fast (CKM) or slow (CKS) clock for reading or resetting RSN cells. Data is stored in RSN1 via the Set (S) signal, moves to RSN2 at a slow clock rate when triggered from the Read (RN) port of RSN1, and is non-destructively read from RSN2 at a fast clock rate when triggered from the RN port. The output of the RSN1 is transmitted from the ZN port to RSN2 while satisfying the correct timing requirements. RSN2 is reset at a slow clock rate through the R signal. The information in RSN2 can be read multiple times within one system clock period, determined by the slow-to-fast clock ratio. This enables NDDFF2 to replace all pipelining DFFs with a single cell providing the same output at the same rate [36]. The functionality of the NDDFF2 cell is presented through the HSPICE waveform in Figure 7(c).

To evaluate the performance limitations of Synopsys Fusion Compiler, we used circuits of various complexities and scaling. The designs were an 8-bit MAC, an 8-bit 8-tap finite impulse response (FIR) filter, and 1Kb memory for mid-sized design, a 64-bit MAC for large-scale design, and a 128-bit MAC for very large-scale design. We presented the implementation outcomes under single and dual clocking schemes in Tables 3 and 4 respectively and reported key metrics for each design such as maximum clock frequency, floorplan area (with a focus on minimizing DRC violations), timing report data including worst negative slack (WNS) for setup time and hold time, total number of gates (excluding

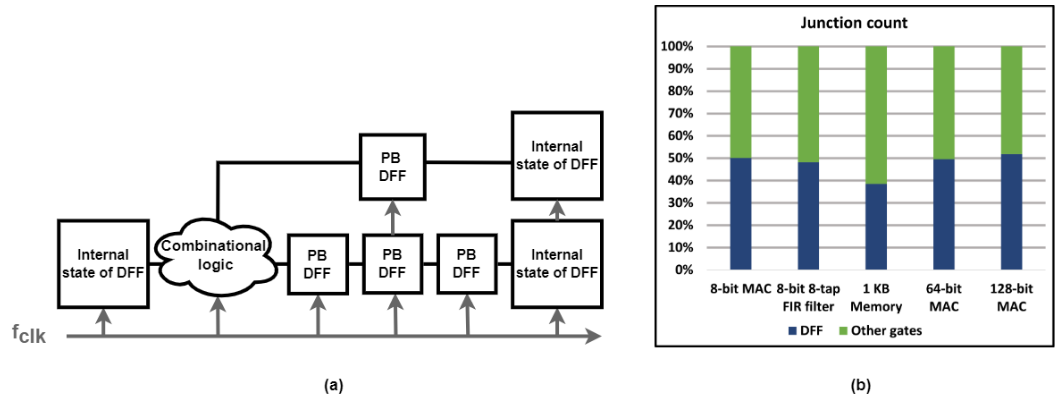


Figure 6. (a) Single clocking scheme with global clock for combinational logic, logical DFF and path-balancing (PB) DFF. As illustrated, lower path needs three PB DFFs and upper path need one PB DFF for correct circuit functionality. (b) Comparison of Josephson junction count from DFF gates (blue) and rest other gates (green) within each design when single clocking scheme is used. The junction counts from DFF account for ~50% of the total junction count of the design.

filler cells for empty spaces), and count of Josephson junctions. It is worth noting that the designs presented in these tables haven't been fabricated and tested. Instead, they serve as a demonstration of Fusion Compiler capabilities. In the single-clocking approach, the increased junction count due to path balancing leads to a larger area and higher power consumption. On the other hand, the dual-clocking scheme prioritizes power efficiency by reducing the gate and junction count but at the expense of lower clock frequency. The lower frequency of the dual-clocking scheme is due to the timing constraints imposed by two clocks, which negatively impact the timing characteristics of NDDFF2 cells. However, for larger circuits such as 64 and 128-bit MAC, the dual-clocking scheme minimally affects operating clock frequency compared to the single-clocking scheme. This is because the effect of latency on clock frequency in large-scale circuits is more significant than the timing characteristics of NDDFF2 cells. The junction count (Figure 8(a)) also shows a significant reduction by at least an order of magnitude with the dual-clocking scheme. Additionally, Figure 8(b) demonstrates a considerable decrease in the operational frequency of the dual clock for medium-complex circuits compared to large-scale circuits. With the current state of the Fusion Compiler, opting for single-clocking is favorable for low to medium-complexity circuits, whereas dual-clocking proves beneficial for larger circuits.

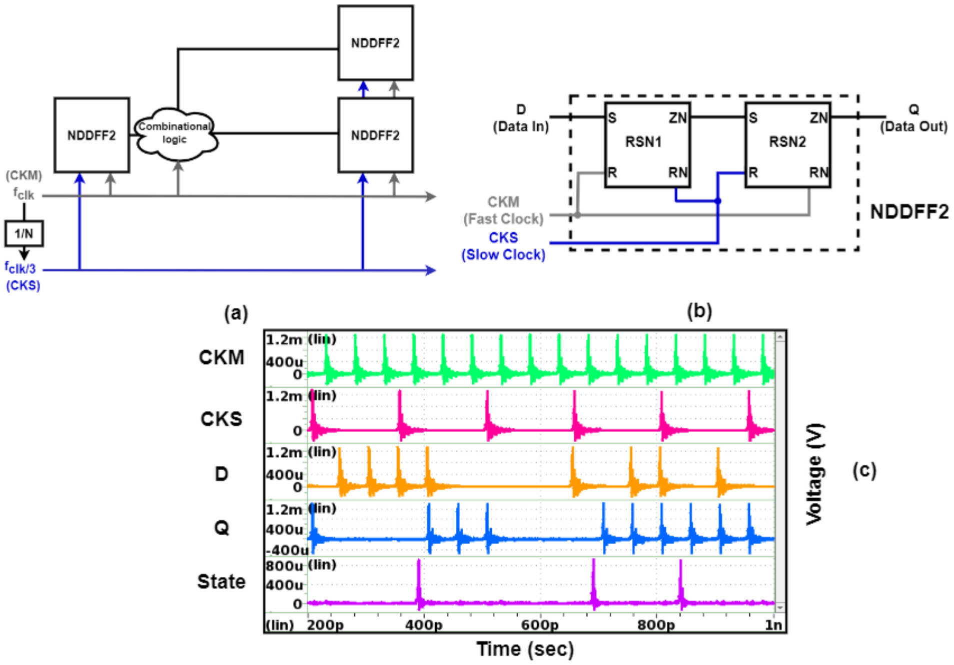


Figure 7. (a) Dual clocking scheme, with fast clock (CKM) and slow clock (CKS), eliminates the use of PB DFF. Based on the maximum pipeline depth of the design in Figure 6 (a) which is 3 clock cycles, the frequency of the slow clock, used to read the internal state of NDDFF2, is 1/3 of the fast clock frequency. (b) Block diagram of NDDFF2 cell. (c) HSPICE waveform showing the functionality of the NDDFF2 cell. In this example, the ratio between fast (CKM) and slow clock (CKS) is kept at 3, therefore the output (Q) reads the value of the internal state (State) 3 times.

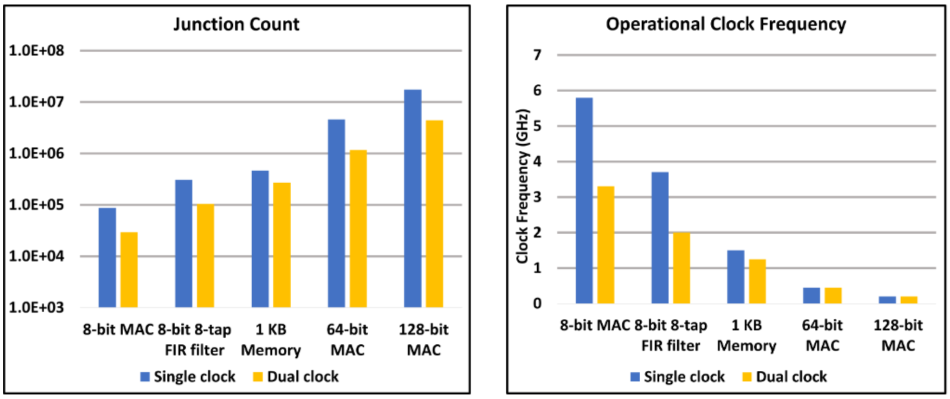
Table 3. Performance metrics of designs from Fusion Compiler using the single-clocking scheme

Design	Max. clock frequency (GHz)	Area (mm x mm)	Setup WNS (ps)	Hold WNS (ps)	No. of gates	No. of junctions
8-bit MAC	5.8	8 × 8	7.42	-0.03	8948	87,360
8-bit 8-tap FIR filter	3.7	14 × 14	2.19	-0.61	31,272	309,280

1 KB Memory	1.5	28 × 28	67.36	-0.41	45,199	463,227
64-bit MAC	0.45	150 ×150	160.27	-1.2	469,089	4,592,021
128-bit MAC	0.2	300 ×300	512.31	-1.37	1,857,969	18,150,632

Table 4. Performance metrics of designs from Fusion Compiler using the dual-clocking scheme

Design	Max. clock frequency (GHz)	Area (mm x mm)	Setup WNS (ps)	Hold WNS (ps)	No. of gates	No. of junctions
8-bit MAC	3.3	4.5 × 4.5	6.06	0.29	2196	29,283
8-bit 8-tap FIR filter	2	9 × 9	17.46	0.24	7237	104,333
1 KB Memory	1.25	17 × 17	154.37	-0.04	18,576	271,324
64-bit MAC	0.45	50 × 50	360.74	-0.04	85,764	1,173,229
128-bit MAC	0.2	100 × 100	649.63	-0.31	325,849	4,434,243



(a)	(b)

Figure 8. (a) Comparison of junction count of different designs using single (blue) and dual (yellow) clocking schemes. (b) Comparison of operational clock frequency of different designs using single (blue) and dual (yellow) clocking schemes.

4. Discussion

The collaboration between Synopsys and Hypres marks a pioneering effort to create a commercially available tool that seamlessly integrates the SFQ RTL-to-GDS design flow with the EDA tool suite. This sets a precedent for an efficient user experience and signifies the successful realization of designs surpassing 10 million Josephson junctions, a crucial milestone defined by the IARPA SuperTools project. However, challenges persist in fully enabling large-scale integration of SFQ circuits.

The superconductor RTL-to-GDS flow adapts semiconductor algorithms through TCL scripting for this unique domain. Table 3 reveals that despite clock tree distribution and buffer insertion, current optimization lacks efficiency, resulting in higher JJ count and increased power consumption. The dual-clocking scheme introduces new cells like NDDFF2 in the standard cell library, addressing these issues but presenting different challenges leading to reduced clock frequency, and added timing complexity. To fully utilize the unique properties of SFQ circuits, it is crucial to develop timing optimization algorithms tailored to superconductor circuits. This will help address the existing limitations and enhance their performance.

Currently, clocking support is limited to the zero-skew scheme for SFQ logic and the 4-phase clocking scheme for AQFP logic [16]. Extending the tool's functionality to accommodate other clocking methodologies, including concurrent-flow, counter-flow, and alternating current (AC) clocking schemes [37], would enable diverse digital architectures. Some of these approaches may reduce latency by optimizing clock tree synthesis and buffering. Furthermore, the current routing algorithm relies on long PTL connections as complexity increases, resulting in increased latency. Actively pursuing the optimization algorithm in logic implementation, while adjusting timing and area constraints to minimize the need for long PTL connections, is crucial.

As circuit design scales, the number of junctions within a chip also increases creating a high current that could disrupt circuit operation. To tackle this issue, serial biasing stands out as an effective technique, and research in this direction is worth considering. The technique involves partitioning a large RSFQ circuit into smaller blocks with uniform bias currents. The serial biasing technique using the dual RSFQ/ERSFQ standard cell library approach for the SuperTools program is presented in [38] which demonstrates reliable operation up to 50 GHz. As an alternative option to serial biasing, it would be worthwhile to explore the adaptation of the EDA tool to support the recently proposed novel AC biasing scheme in [39].

While this commercial tool represents a significant advancement in SCE design, it also highlights complexities and potential areas for improvement in achieving large-scale integration of SFQ circuits. Addressing these challenges and focusing on the advancement of different aspects of this technology is crucial for realizing the full potential of superconductor digital circuits.

5. Conclusion

In conclusion, the integration of SFQ technology into a mainstream EDA platform is a significant step towards commercialization and wider adoption of SFQ circuits. The implementation of very large-scale SFQ circuits is made possible with the support of RTL-to-GDS flow and a robust standard cell library. SFQ logic introduces a significant pipeline depth as the gates are synchronized with the clock. It is crucial to ensure correct data and clock arrival from different parts of the circuit for meaningful output. Historically, this has been done by adding flip-flops, which increase the junction overhead, area, and total power consumption. However, the dual-clocking scheme with NDDFF2 cells notably reduces gate count, enhancing power efficiency and shrinking chip area. This has been verified using Synopsys Fusion Compiler with Hypres designs, which further affirms its capabilities for VLSI SFQ circuits. Future research should focus on exploring diverse clocking schemes and optimizing the latency of routing networks to unlock the full potential of SFQ circuits for high-speed operation. Additionally, efforts should be directed toward addressing the current demand for complex designs.

Acknowledgments

The authors would like to thank A. Shukla, T. Filippov, D. Kirichenko, A. Kadin, and E. Track for their valuable contribution and helpful suggestions. The authors are grateful to various Synopsys groups for their contributions to EDA tool development, especially S. Whiteley, E. Mlinar, and A. Baker for sharing insights into PDK and tool usage.

This work is supported in part by the Office of the Director of National Intelligence, Intelligence Advanced Research Projects Activity (IARPA), via the U.S. Army Research Office under Contract W911NF-17-9-0001. 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 Office of the Director of National Intelligence, Intelligence Advanced Research Projects Activity, or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation herein.

Appendix

Table 5. Process corner definition

Corner	Bias current scaling parameter (XI)	Junction critical current scaling parameter (XJ)	Inductor scaling parameter (XL)	Junction shunt resistance scaling parameter (XR)	Parameter for missing junction radius (dR)
Nominal	1	1	1	1	0
Fast	1	0.9	0.95	1.05	0
FastFast	1.1	0.9	0.95	1.05	0
Slow	1	1.1	1.05	0.95	0
SlowSlow	0.9	1.1	1.05	0.95	0
Fast_dR_0.02	1	0.9	0.95	1.05	0.02
Fast_dR_0.04	1	0.9	0.95	1.05	0.04
Slow_dR_0.02	1	1.1	1.05	0.95	-0.02
Slow_dR_0.04	1	1.1	1.05	0.95	-0.04
XI_1.1	1.1	1	1	1	0
XI_0.9	0.9	1	1	1	0

Table 6. List of library cells in the PTL-based Standard Cell Library

Synchronous cells	**Asynchronous cells (Interconnect cells)**	**Interface cells**
OR	1-to-2 splitter	DC to SFQ
NOR	1-to-3 splitter	SFQ to DC
D flip-flop (DFF)	1-to-4 splitter	
NOT	RTX0	**Other cells**
XOR	RTX1	Filler
Set reset flip flop with non-destructive readout (RSN)	RTX2	
NDDFF2	RTX4	
NDDFF2P	RTXU2	
NDDFF2C		

References

[1] Holmes D S, Ripple A L and Manheimer M A 2013 *IEEE Trans. Appl. Supercond.* **23** 1701610-1701610

[2] Likharev K K and Semenov V K 1991 *IEEE Trans. Appl. Supercond.* **1** 3-28

[3] Kirichenko D E, Sarwana S and Kirichenko A F 2011 *IEEE Trans. Appl. Supercond.* **21** 776-779

[4] Holmes D S 2022 *IEEE International Roadmap for Devices and Systems*, Chapter 2 on Cryogenic Electronics and Quantum Information Processing 3-41 [Online] https://irds.ieee.org/editions/2022/irds%E2%84%A2-2022-cryogenic-electronics-and-quantum-information-processing

[5] Tolpygo S K 2016 *Low Temp. Phys.* **42** 361-379

[6] Fourie C J and Volkmann M H 2013 *IEEE Trans. Appl. Supercond.* **23** 1300205-1300205

[7] Polonsky S, Shevchenko P, Kirichenko A, Zinoviev D and Rylyakov A 1997 *IEEE Trans. Appl. Supercond.* **7** 2685-2689

[8] PSCAN2 superconductor circuit simulator [Online] http://pscan2sim.org

[9] WRspice circuit simulator [Online] http://www.wrcad.com/wrcad.com

[10] Delport J A, Jackman K, Roux P I and Fourie C J 2019 *IEEE Trans. Appl. Supercond.* **29** 1-5
[11] Xic Graphical Editor [Online] http://www.wrcad.com/xic.html
[12] Tolpygo S K *et al* 2019 *IEEE Trans. Appl. Supercond.***29** 1-13
[13] IARPA SuperTools Program [Online] https://www.iarpa.gov/research-programs/supertools
[14] Inamdar A, Ravi J, Miller S, Meher S S, Çelik M E and Gupta D 2021 *IEEE Trans. Appl. Supercond.* **31** 1-7
[15] Freeman R, Kawa J and Singhal K 2020 [Online] https://www.synopsys.com/content/dam/synopsys/solutions/documents/gomac-synopsys-supertools-paper.pdf
[16] Tanaka T *et al* 2023 *IEEE Trans. Appl. Supercond.* **33** 1-6
[17] Hironaka Y *et al* 2023 *IEEE Trans. Appl. Supercond.* **33** 1-5
[18] Shahsavani S N, Lin T R, Shafaei A, Fourie C J and Pedram M 2017 *IEEE Trans. Appl. Supercond.* **27** 1-8.
[19] Schindler L, Delport J A and Fourie C J 2022 IEEE Trans. Appl. Supercond. 32 1-7
[20] Fourie C J *et al* 2023 *IEEE Trans. Appl. Supercond.* **33** 1-26
[21] SUNY RSFQ Cell Library [Online] http://www.physics.sunysb.edu/Physics/RSFQ/Lib/contents.html
[22] Yorozo S, Kameda Y, Terai H, Fujimaki A, Yamada T and Tahara S 2002 *Phys. C, Supercond.* **378-381** 1471-14747
[23] Maezawa M, Ochiai M, Kimura H, Hirayama F and Suzuki M 2007 *IEEE Trans. Appl. Supercond.* **17** 500-504
[24] Inamdar A, Amparo D, Sahoo B, Ren J and Sahu A 2017 *IEEE Trans. Appl. Supercond.* **27** 1302109
[25] Meher S S, Ravi J, Çelik M E, Miller S, Sahu A, Talalaevskii A and Inamdar A 2021 *IEEE Trans. Appl. Supercond.* **31** 1-7
[26] Tolpygo S K, Bolkhovsky V, Weir T J, Wynn A, Oates D E, Johnson L M and Gouker M A 2016 *IEEE Trans. Appl. Supercond.* **26** 1-10
[27] Inamdar A, Meher S S, Chonigman B, Sahu A, Ravi J and Gupta D 2023 *IEEE Trans. Appl. Supercond.* **33** 1-8
[28] Inamdar A, Ravi J, Habib M, Meher S S, Sahu A and Gupta D 2023 *IEEE Trans. Appl. Supercond.* **33** 1-7
[29] Shukla A, Chonigman B, Sahu A, Kirichenko D, Inamdar A and Gupta D 2019 *IEEE Trans. Appl. Supercond.* **29** 1-7
[30] Lo S C *et al* 2022 *Inter. Sym. on Quality Electronic Design* 33-38
[31] Chonigman B *et al* 2021 *IEEE Trans. Appl. Supercond.* **31** 1-6
[32] Baker A J, Mlinar E, Lo S C and Singhal K 2023 *IEEE Trans. Appl. Supercond.* **33** 1-4
[33] InductEx [Online] https://sun-magnetics.com/
[34] Amparo D, Çelik, M E, Nath S, Cerqueira, J and Inamdar A 2019 *IEEE Trans. Appl. Supercond.* **29** 1-9
[35] Inamdar A, Ravi J, Miller S, Meher S S, Çelik M E and Gupta D 2021 *IEEE Trans. Appl. Supercond.* **31** 1-7
[36] Mlinar E *et al* 2023 *IEEE Trans. Appl. Supercond.* **33** 1-7
[37] Herr Q P, Herr A Y, Oberg O T, Ioannidis A G 2011 *J. Appl. Phys.* **109** 103903
[38] Shukla A, Fillipov T V, Kirichenko D E, Meher S S, Çelik M E, Seok M and Gupta D 2022 *IEEE Trans. Appl. Supercond.* **32** 1-12
[39] Semenov V K, Golden E B and Tolpygo S K 2021 *IEEE Trans. Appl. Supercond.* **31** 1-7