Superconducting Shuttle-Flux Shift Register for Race Logic and its Applications

Meriam Gay Bautista, *Member, IEEE*, Patricia Gonzalez-Guerrero, *Member, IEEE*, Darren Lyles, *Member, IEEE*, and George Michelogiannakis, *Senior Member, IEEE*

***Abstract*—This paper presents a superconducting, magnetically-coupled, shuttle-flux shift register (SF-SR) that stores single flux quantum (SFQ) pulses. This shift register has a DC bias operating margin of $\pm 34\%$ at 10 GHz, with a power dissipation of $3.6\mu W$ and 38% fewer Josephson junctions (JJs) when scaled up to multiple stages compared to a data flip-flop (DFF) based shift register. The clock input is inductively coupled and is independent from the data input. We then present three applications for our SF-SR. In the first application, we add two non-destructive readout (NDRO) cells to construct a buffer that temporarily stores the temporal information of a series of race logic (RL) pulses. The second application is a pseudo-random number generator based on a linear function shift register (LFSR). The third application is N parallel SF-SRs that can act similar to a deserializer or instead can emulate a single SF-SR of N times higher clock frequency. These three applications motivate deep shift registers with many shifting intervals, which our SF-SR can implement with fewer JJs and lower power consumption compared to DFF-based shift registers.**

***Index Terms*—Buffer, Pseudo-Random Number Generator, Race Logic, Shift Register, Shuttle Flux, Superconducting Circuit.**

I. Introduction

SUPERCONDUCTING digital computing is a promising alternative to preserve performance scaling after traditional complementary metal oxide semiconductor (CMOS) scaling ends [2], [3]. Superconductors exhibit 0Ω resistance when cooled below their critical temperature (T_c), which is usually a few degrees Kelvin. Therefore, energy dissipation of superconductors is drastically lower compared to traditional CMOS technology even after accounting for cooling [3], [4]. Superconducting circuits were shown to reach up to seven orders of magnitude higher energy efficiency [5], [6].

Superconducting digital circuits are typically based on the single flux quantum (SFQ) logic family [7], [8], where the presence of a picosecond-duration pulse encodes the value of 1 and the absence of a pulse encodes the value 0. The Josephson junction (JJ) is the fundamental building block for superconducting digital computing. A JJ is made up of two superconductors sandwiching a thin non-superconducting barrier such that electrons can tunnel through the barrier. SFQ is a current-based technology. The JJ switches from a superconducting state to a resistive state when the current flowing through the JJ exceeds its critical current (I_c). A superconducting loop containing a pair of JJs is known as a superconducting quantum interference device (SQUID). A SQUID is the basic SFQ element that stores a magnetic flux quantum. When the input current plus the bias current exceeds the JJ critical current ($|I_{in} + I_{bias}| > I_c$), the JJ produces an SFQ pulse.

Circuits based on the rapid single flux quantum (RSFQ) logic family, currently the most popular SFQ variant, have become popular due to their ability to transfer SFQ pulses with low latency and energy [9]. RSFQ circuits are a potential game-changer compared to traditional CMOS due to their high-speed operation and low dynamic power consumption. Several SFQ logic cells have been demonstrated such as set-reset (SR) latches, toggle flip-flops (TFFs), D flip-flops (DFFs), multipliers, multiplexers, demultiplexers [7], and analog-to-digital converters [10]. In addition, shift registers (SRs) have been used for temporary data storage, data transfer, data manipulation, time delay, synchronization between clock domains, and conversion between serial and parallel data representations (e.g., demultiplexers for communication lines) [11]–[13].

Recent work proposed race logic (RL) in RSFQ to mitigate the tight area constraints of modern RSFQ chips [14], because superconducting device technology is three orders of magnitude less dense than state-of-the-art CMOS [3]. RL assigns values to each pulse based on the pulse's delay from a reference pulse. Time is divided into what we refer to as "time slots" to assign discrete values. For example, as illustrated in Fig. 1, suppose a time slot duration is set to 10ps. In this scenario, a pulse arriving at 25ps after the reference signal is in the third time slot and thus it is interpreted as the value 2 (assuming the first time slot encodes the value 0). A pre-defined number of time slots defines an epoch, after which the reference pulse arrives again in order to reset the values. The number of time slots in an epoch defines the range of possible values each pulse can encode. RL in RSFQ has demonstrated higher performance [14]–[16] and thus is a promising computational model for future RSFQ circuits.

Storing or buffering RL pulses is challenging because doing so must preserve each pulse's delay relative to the reference signal. In other words, storing an RL-encoded pulse requires releasing it in the same time slot of a future epoch. Using a single DFF would discard the temporal information. Instead, we can use a shift register with as many stages as time

Manuscript received February 25, 2022. Revised June 22, 2022. The first version of this paper has been submitted and presented in MWSCAS2021 and published in IEEE Explore with DOI: 10.1109/MWSCAS47672.2021.9531899, [1].

The authors are with the Computer Architecture Group, Applied Mathematics and Computational Research Division, Lawrence Berkeley National Laboratory, Berkeley, California, USA., Email: [mgbautista,lg4er,dlyles,mihelog]@lbl.gov

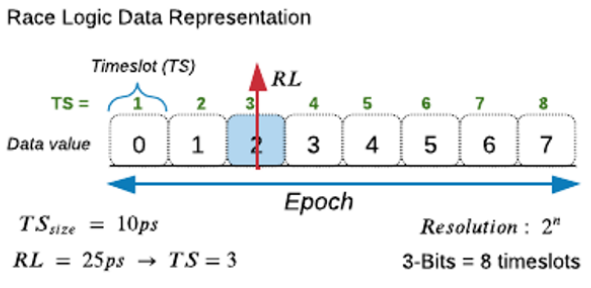

Fig. 1. Race logic (RL) data representation: In the example above, we have a data range of zero to seven (equivalent to 3 bits in binary), which is equivalent to eight time slots in RL. A pulse arriving during time slot number 2 encodes the value of 2.

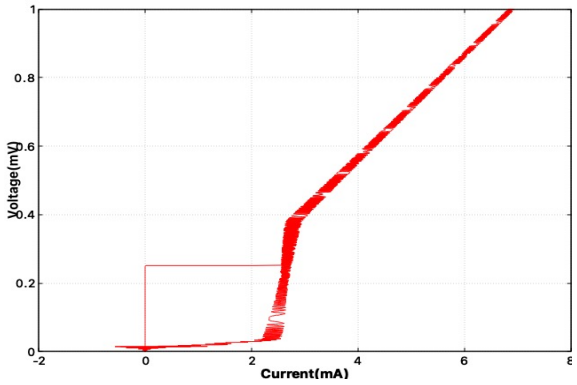

Fig. 2. Transient current-voltage (IV) curve of a Josephson junction (JJ) with $I_c = 250\mu A$. The gap voltage of the junction is 2.5mV.

slots in an epoch. However, this requires many JJs if we use conventional shift registers that use one DFF that consists of six JJs per stage [7], [17].

In addition to RL, other computational models such as spiking neural networks that interpret values by the number of incoming pulses [18], digital signal processing (DSP) [19], and NoCs (Networks-on-Chip) [15], [20], also motivate shift registers that are used as storage or buffers. In those applications, scaling up computation or the range of data values that each operand can represent requires deep shift registers, up to hundreds or thousands of stages; this would quickly become impractical with DFF-based shift registers.

To that end, we consider various shift register implementations based on RSFQ [11], [12], [21], quantum flux parametric (QFP) [22], and shuttle flux [23]–[26]. Among those three designs, shuttle flux-based designs have demonstrated a larger operating margin for each shifting operation [27]. Based on this exploration, we propose a shuttle-flux shift register (SF-SR) design based on scalable magnetically-coupled shuttle flux SQUIDs. The master cell that is the first stage of the SF-SR has only four JJs; each extended slave cell uses only one JJ per shifting interval (stage). The delays between different shifting intervals are configurable by three clock inputs. One advantage of the SF-SR is that the input clock pulse train can be applied to a long shuttle via the three series of clock control lines. Also, our SF-SR does not need an additional bias current compared to other shift registers that require a clock tree bias, which would require an additional four JJs [12].

Based on our SF-SR, we then propose a buffer (storage) for RL-encoded pulses that includes control inputs to determine when to store and release pulses, implemented by adding non-destructive read-out (NDRO) cells. Our buffer can store RL pulses for any number of epochs while preserving their value (i.e., releasing them in the same time slot during a future epoch) [14]. This can enable practical area-efficient storage for computing accelerators or networks that use RL. A second practical application is a pseudo-random number generator. In this application, our SF-SR replaces the typical chain of DFFs used to implement the linear feedback shift register (LFSR). Finally, the third application is a single-input delay element that consists of a set of parallel SF-SRs that can either produce multiple outputs in parallel and thus act as a deserializer or instead produce a single output and behave similarly to a single SF-SR that is clocked at an N higher frequency than each of the N parallel SF-SRs can support. The latter is useful in cases where we want to emulate a single SF-SR that is clocked at a frequency that violates a single SF-SR's timing constraints. In all three applications, our SF-SR has the lowest area overhead per stage and low power consumption compared to the state-of-the-art.

II. Background

A. Josephson Junction

A Josephson junction (JJ) is a device that consists of two superconducting materials weakly coupled by an insulating barrier [28]. Two equations characterize the JJ: the current across the JJ $I = I_c sin(\delta)$ and the voltage across the JJ $V = (\Phi_0/2\pi)\delta$; δ is the superconducting phase difference across the JJ, I_c is the critical current, and the flux quantum is $\Phi = 2x10^{-15}$ Weber. To evaluate the transport properties of a JJ, we plot the current-voltage (IV) curve of a JJ with critical current $I_c = 250\mu A$, as shown in Fig. 2.

The JJ is at a superconducting state when $I = 0$ or $I < I_c$. As we increase the critical current I_c, the JJ is trapped at a fixed value of superconducting gap δ (few millivolts) without changing the voltage potential across the JJ. As we further increase the bias current, the JJ voltage slowly increases. As the current flowing through the JJ exceeds the critical current value I_c, the tunneling junction switches from a superconducting state to a resistive state. As we increase the bias current beyond that point, the JJ voltage increases linearly. It has been shown that the damping of the JJ and the sweep rate of bias current influence the shape of the IV curve [29].

B. Superconducting Circuits

An SFQ pulse is produced when the magnetic flux ϕ in a superconducting loop of inductance L that contains a JJ changes by one flux quantum as a result of the JJ switching. A SQUID is a basic SFQ cell and consists of a superconducting loop with two JJs. The physical representation of information is based on the quantization of magnetic flux. The magnetic flux inside the SQUID can be expressed as $\phi = h/2e$, where e is energy approximately equal to $2x10^{-19}$ J at $T = 4K$.

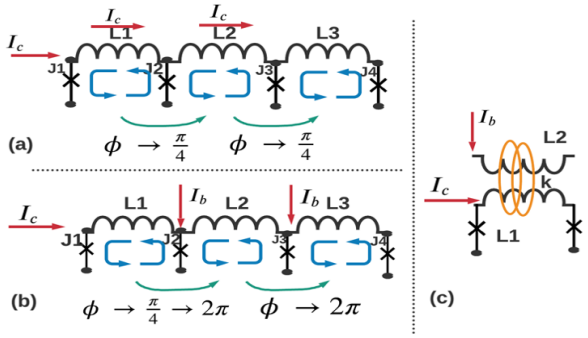

Fig. 3. Shuttle flux line. (a) Current I_c (data pulse) is applied at J1. The current flows sequentially from left to right to change the phase by $\pi/4$. The blue arrows represent the superconducting current loop (vortex). (b) Direct bias current I_b is applied at J2 to shift the flux by 2π to the next vortex. (c) An interferometer is a series structure of SQUIDs inductively coupled to one of the control lines.

C. SFQ Switching Energy

Based on thermodynamics entropy and the Shannon-Von Neumann-Landauer limit, the energy spent to switch one bit from a 0 to a 1 is $E_{bit} = 4\ x\ 10^{-21}\ J$ at $T = 300°\ K$ [30]. In CMOS, the energy required for reliable data transmission is about $E_{CMOS} = 10^6 \times E_{bit}$, while in superconducting SFQ technology, the energy required to transfer SFQ pulses is approximately $E_{SFQ} = 10^3 \times E_{bit}$ J at $T = 4\ K$. This makes SFQ an attractive alternative to conventional CMOS technology [30], [31] even after considering the power required for cryo-cooling that can result in a power penalty of $300\times$ to $400\times$ [32], [33].

D. SFQ Shuttle Flux

A shuttle flux JJ shift register was initially proposed by Anderson [24]. His work shows that the superconducting pairs of conductors through an insulating barrier could have a tunnel effect, also known as the *Josephson effect* [34]. The superconducting current equation is $I_s = I_i\ sin\ \phi$, where I_i is the input current and ϕ is the difference between the phases of the electron pair's "wave function" of the two isolated superconductors. The phase difference is related to the energy $E = -E_t\ cos\ \phi$. It was demonstrated that if the phase is time-independent, the supercurrent flows have no voltage drop across the superconducting JJ [24] [35]. However, an applied external current can alter the circuit's phase to be unequal.

As illustrated in Fig. 3, a guided motion of the magnetic flux quanta (blue circulating arrow) is also known as a vortex, where a meta-stable persistent current loop is present between two JJs coupled by an inductor. The vortex consists of a localized loop of K_x fed by JJ_x, which will induce a magnetic-self flux $\Phi = \Phi_0$ in a thin surface layer of thickness λ, the penetration depth. The position of the vortex can be shifted by applying current or a magnetic field. The potential energy stored in the vortex is given by equation [23]:

$$E_0 = \frac{4}{\pi}\ \Phi_0\ \lambda_j\ J_c \tag{1}$$

The energy required to move the vortex is [23]:

$$E = E_0 \sqrt{(a - \frac{V^2}{I_c^2})} \tag{2}$$

To transfer the vortex to the next superconducting loop, current I_c is applied to J1, as shown in Fig. 3(a). A new vortex would then be formed repeatedly, where the JJ will undergo a transition from 0 (state 0) to V^+ (state 1). The equivalent of the trapped vortex is a current loop concentrated primarily between the two adjacent JJs. Subsequently, the current applied sequentially moves the vortices over a JJ in the direction of the current I. In addition, the phase ϕ moves by about $\frac{\pi}{4}$. To change the phase from $\frac{\pi}{4}$ to 2π, a bias current or magnetic field is applied alternately at every other JJ, as shown in Fig. 3(b). At this stage, the magnetic self-flux is $\Phi = \Phi_0$ and the phase increases by 2π. The input voltage pulse will decrease the flux of inductor L1 and increase that of inductor L2, allowing the vortex to transfer to the next loop. The current pulse I_c that induces the shift must last a time Δt enough to accomplish the shifting.

One bit of information is represented as a flux quantum vortex [23]. In this manner, an ordered array of vortices represents a binary number. This operation can carry a shifting process simultaneously in parallel for all vortices, making it attractive for a shift register (SR) application.

A shuttle-flux shift register (SF-SR) was demonstrated in [27]. This circuit consists of three interferometers with three loops per bit. Another implementation of an SF-SR was described in [36], shown in Fig. 6(a). This circuit comprises two loops with a master and a slave JJ. A shuttle flux circuit has the potential advantage of high-speed operation and low energy dissipation of the switching operation over conventional shift registers based on data flip-flops (DFFs) [23].

III. Circuit Design Description

A. Implementation of Single-Stage Shuttle Flux Shift Register

Our implementation of a single-stage SF-SR extends prior art [23], [25], [27], [36] and is illustrated in Fig. 4. For our implementation, we use MIT-LL SFQ5ee [37] technology library and simulate using WRSPICE. As illustrated in Fig. 4, our SF-SR consists of the following four stages:

Input stage: The input DC pulse that carries incoming data is converted into an SFQ pulse using a DC-to-SFQ converter circuit adapted from [7] and illustrated in Fig. 4(a). The purpose of this circuit is to produce a controlled number of SFQ pulses by changing the pulsating DC current applied at the input.

N shifting stages: The master cell consists of three magnetically-coupled interferometers, as shown in Fig. 4(b), where R1, R2, and R3 are the bias resistors connected to the three-phase input clock pulses [27]. An interferometer is a series structure of SQUIDs inductively coupled to one of the control lines [25]. This structure has three three-phase input clock currents (P1, P2, and P3) that determine the shifting direction. The magnetic flux quanta from the DC-SFQ input stage will circulate at the first SQUID storage loop at J1, L1, and J2, respectively. The clock pulse P1 will induce current

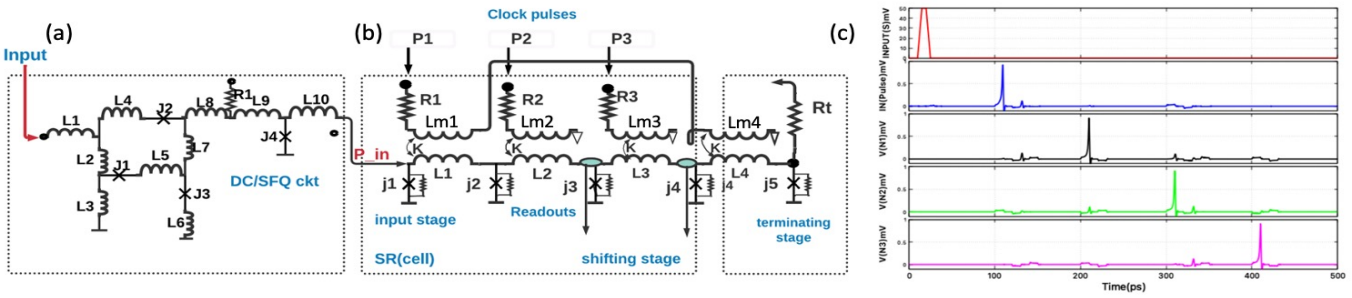

Fig. 4. (a) DC-to-SFQ conversion circuit. (b) A three-stage shuttle flux shift register cell: Reading the content of each stage (readouts) can be obtained from JJs using a JTL or a coupling inductor. (c) Simulation waveform. Inputs are shown in red. In this case, the only input is the DC input pulse signal. P_{in} is shown in blue and is the SFQ pulse generated from the DC−SFQ circuit. Pulses at $V(N1, N2, N3)$ are the shifted pulses for each of the three shifting stages. In this example, the shifting interval (delay per stage) is set to 200ps, (full shift at node J4) and a 100ps mid−interval at J3.

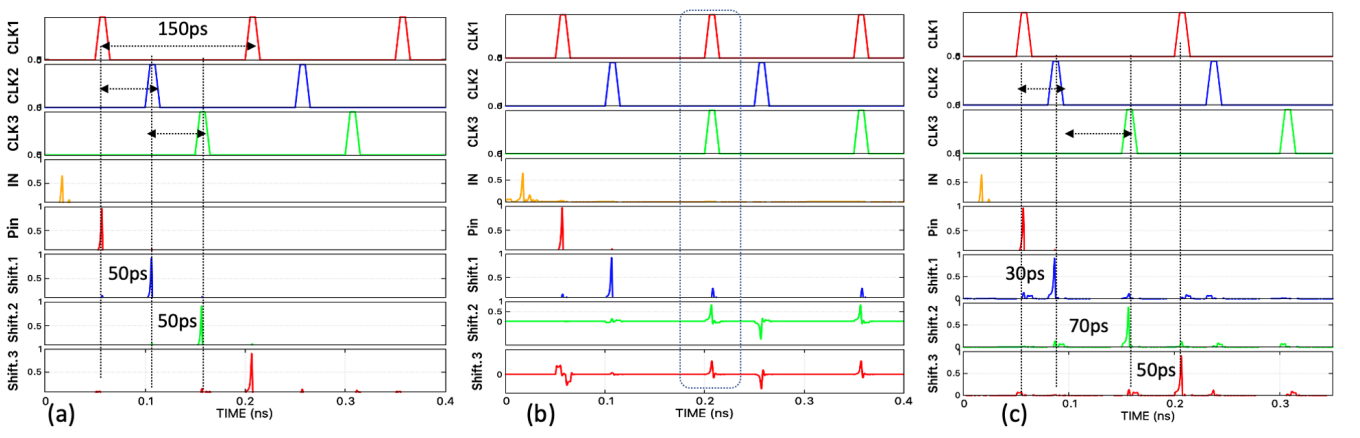

Fig. 5. Clock tuning method. (a) Uniform interval: the three clocks have a uniform delay between them. The delays between clocks are set to 50ps and the period is set to 150ps. (b) Overlapping clocks: clock pulses overlap, resulting in a distorted output signal. Overlapping clocks is an invalid configuration. (c) Un-even clock delays result in uneven shifting. In this example, CLK2 is delayed by 30ps and CLK3 is delayed by 100ps in reference to CLK1.

(I_{bias}) across R1. The inductively-coupled inductors L1 and Lm1 have a coupling coefficient of K. The sum of the current from the first storage loop and I_c will exceed the critical or threshold current, which switches J2 and shifts J2's phase to $\phi = 2\pi$. The flux quanta will continue to move to the subsequent storage loop L2 or L3 by applying current in P2 or P3, respectively. The shifting stage easily scales to multiple stages, as shown in Fig. 6 and discussed in detail later. We refer to one cell with three JJs as a full shift, whereas a pair of JJs and two coupled inductors is referred to as a half shift.

Readout stage: The shifted pulses can be read at the common node connection between the vortices, the coupled inductors, and the JJ (for instance, J3 and J4), as shown in Fig. 4(b). The shift interval is set by tuning the phase of the three input clock pulses. In our example, we use the common node at J2 and J3 for our readouts, R01 at the first full shift, R02 at the mid-shift, and R03 at the second stage, as shown in Fig. 4(c). For our implementation, we have two readouts for every stage, though each readout is optional. An inductor or a Josephson transmission line (JTL) can be used for signal amplification at the readouts.

Terminating stage: The terminating stage consists of the last storage loop and a coupled inductor ($JJ_{lastterm}$). The pulse is terminated using a load damping resistor at the last JJ (R_{tf}).

B. Clock Tuning Method

Our SF-SR has three clock inputs, P_1, P_2, and P_3. In this work, we use external bias clock inputs. The delays of the three clock pulses set the time interval between shifting (i.e., the delay of each shifting stage).

Fig. 5 shows three scenarios for tuning the input clocks. CLK1, CLK2, and CLK3 are the input clocks, respectively. IN is the data input signal to the DC-SFQ and produces an SFQ pulse P_in, which is then the input to the SF-SR. Shifts 1-to-3 are shifted versions of input pulses after each SF-SR stage. Fig. 5(a) shows that the three clocks have a constant delay relative to each other. Here we have a 50ps delay (refer to $\Delta_t(CLK1 - CLK2)$, $\Delta_t(CLK2 - CLK3)$ and therefore a full shift of 150ps. The period of each clock is what we refer to as full shifting. Fig. 5(b) shows that when the clock pulses overlap, the result is distorted and interrupts the shifting of the pulse. Therefore, overlapping clocks is an invalid configuration. Finally, Fig. 5(c) shows an uneven delay interval between clocks with mixed frequency clocks. By tuning the clock delays, the SF-SR can have a non-uniform shift interval. Here, we can use CLK2 and CLK3 to adjust the mid-intervals (i.e. 20ps and 80ps mid-shift at 150ps full-shift). This interval remains throughout the shuttle. This scenario can be useful to applications that require uneven shifting.

TABLE I
TIME DELAY EVALUATION OF OUR SINGLE-STAGE SHUTTLE FLUX SHIFT REGISTER.

Period	Freq	At node j2	At node j3	At node j4
50 ps	20 GHz	3 ps	3 ps	3 ps
200 ps	5 GHZ	5 ps	2 ps	3 ps
1000 ps	1 GHz	5 ps	3 ps	3 ps

TABLE II
TIME DELAY EVALUATION OF OUR MULTI-STAGE SHUTTLE FLUX SHIFT REGISTER. WE EVALUATE A 7−STAGE−SR COMPOSED OF ONE MASTER CELL AND SIX SLAVE CELLS.

Shifting stages (delay in ps)								
Time	Freq	1	2	3	4	5	6	7
500ps	20 GHZ	2	2.5	2.6	2.6	2.8	2.8	2.9
5ns	200 MHz	1	2.5	2.6	2.6	3	3	3.1

We refer to mid-interval as the delay difference from CLK2 and CLK3 in reference to CLK1. Setting a mid-interval is one of the advantages of using three clock inputs because we can set up more fine-grain delays than just for one stage. For instance, for a two-stage SF-SR with 8 JJs, we can configure two 300ps shift intervals or six 100ps using the same number of JJs. This is how we can double the number of RL time slots (SF-SR stages) without doubling the number of JJs, thus reducing the number of JJs per stage. This reduces the area overhead of storing long pulse trains or long collections of RL pulses.

C. Implementation of a Multiple-Stage Shift Register

To allow our shift register to scale to an arbitrary number of shifting stages, we adapt the master-slave design of [27], shown in Fig. 6. We do not need an extra current bias connection to the slave cell because the clock bias is connected in series. To implement multiple stages, a set of interferometers (slave cells) is connected in series to the last JJ of the master shift register cell. Also, the coupled clock inductors are connected in series to the clock inductors that are also magnetically coupled with mutual inductance m.

We compare the JJ count of a three-stage shift register (shown in Fig. 9) based on our SF-SR design and a conventional DFF-based shift register (shown in Fig. 10) in Table IV. Our SF-SR has 38% fewer JJs. Even though the DFF-based shift register only requires one clock input, a higher number of stages would require more splitters in the clock line, as projected in Fig. 11. In RL, one DFF corresponds to a one-time slot. However, our SF-SR cell can be configured to contain two or three time slots (as long as there is no more than one pulse per cell), which provides more favorable scaling characteristics.

TABLE III
INPUT−TO−OUTPUT DELAY VARIABILITY (DELAY DIFFERENCE COMPARED TO EXPECTED DELAY) OF OUR SHUTTLE FLUX SHIFT REGISTER (SF−SR).
Clk_w: CLOCK PULSE WIDTH, SD: STANDARD DEVIATION.

Order of arrival of pulse in sequence								
Clk_w	1st	5th	10th	15th	20th	25th	30th	SD
5ps	0.6ps	0.4ps	0.5ps	0.9ps	0.7ps	1ps	0.5ps	0.22
10ps	1ps	6ps	2ps	2.2ps	1.7ps	1.5ps	2.5ps	1.65

TABLE IV
JJ COUNT BREAKDOWN FOR THREE-STAGE SHIFT REGISTERS (SRS).

Our design		DFF-based	
SR (4 JJs per stage × 3)	12	SR (5 JJs per DFF × 3)	15
DC-SFQ	4	DC-SFQ	4
Terminating stage	1	Clk line splitters (3x)	9
Total count	**17**		**28**

Table V compares our SF-SR against other shuttle flux-based shift registers from literature, implemented in RSFQ, QFP, and using shuttle flux.

D. Delay Evaluation

We evaluate input-to-output delays using WRSPICE for our single-stage shift register (i.e., a single SF-SR cell) with clock periods of 50ps, 200ps, and 1000ps, as shown in Table I. In reference to Fig. 4, the next shifted pulse at node J3 is evaluated without connecting to a JTL at the readout stage. However, for node J4, we use a JTL at the read-out at node j4 to evaluate the effect of having a load connection. We also evaluate the delay per stage with respect to the clock for a seven-stage SF-SR (composed of one master cell and six slave cells), as shown in Table II. Similarly, we evaluate the variability of our SF-SR with respect to the delay between the input and output nodes. For this experiment, we run multiple input pulses and record the difference between expected and measured input-to-output shift delays for 5ps and 10ps clock pulse widths, as shown in Table III. The numbers shown are not the absolute delay values but rather the variability from the expected delay based on the SF-SR's configuration. Fine-tuning the clock pulse width to 5ps results in lower variability and higher accuracy as compared to a 10ps pulse width.

IV. APPLICATION I: STORAGE BUFFER FOR RACE LOGIC

To store pulses encoded in RL, the shift register must have as many stages as time slots in an epoch. While this is feasible with a shift register implemented by DFFs in series that are clocked with a period equal to a time slot duration, the result is a large per-stage number of JJs. Therefore, as the first application, we use our proposed SF-SR to implement an area-efficient storage buffer capable of preserving the temporal information of RL pulses it contains, thus preserving their value. This requires a shift interval equal to the number of time slots in an epoch, as shown in Fig. 13. For example, an epoch with eight time slots needs eight shift stages. However, as we scale the number of bits of the equivalent binary representation, the number of time slots and thus shift stages scales at a rate equal to 2^N where N is the number of bits. With a DFF-based shift register, the JJ count may quickly grow to impractical numbers.

In order to allow pulses to loop back from the shift register's output back to its input, we implement a simple ring counter based on a shift register, as illustrated in Fig. 12(a). Here, the output of the last readout node is connected back to the input

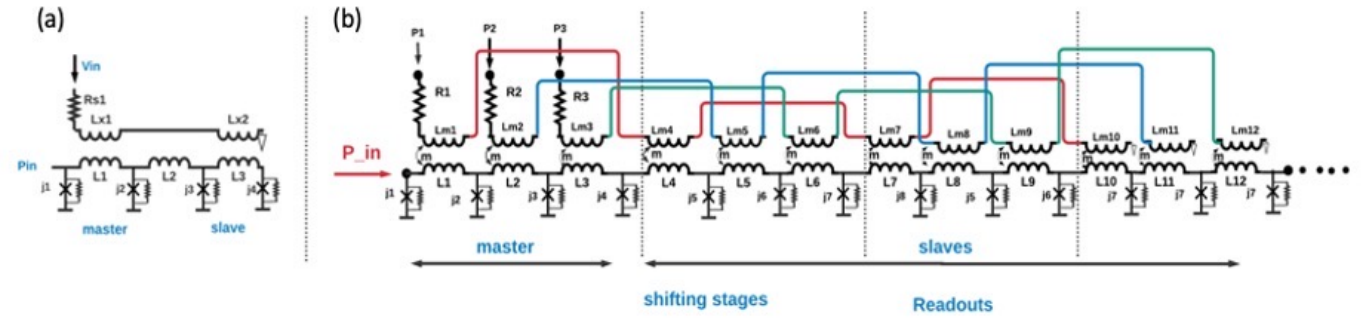

Fig. 6. (a) Master-slave based shift register [27]. (b) We adopt the master-slave design to implement multiple stages in our shuttle flux shift register.

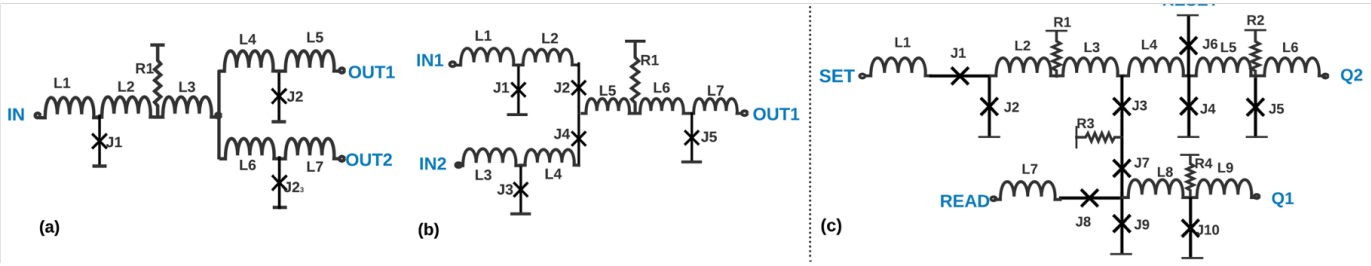

Fig. 7. Circuit schematic: (a) A splitter produces a pulse at both outputs for each input pulse (implements fan-out). (b) A merger produces a pulse at the output for a pulse at either input (implements fan-in). (c) Non-destructive read-out (NDRO) cell: Input pulses arriving to the READ input propagate to output Q1 if NDRO is in a "set" state.

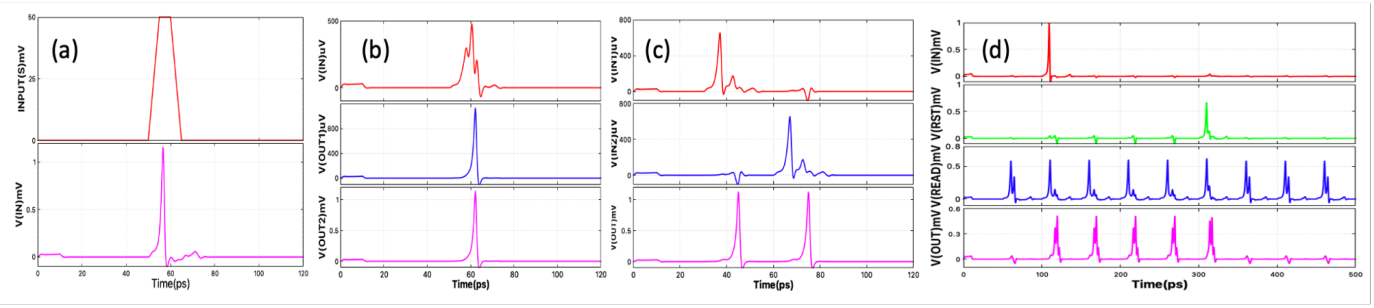

Fig. 8. Simulation waveform: (a) DC-SFQ: The incoming DC pulse is converted to an SFQ pulse with a picosecond pulse width, (b) Splitter: An incoming pulse produces a pulse at both outputs. (c) Merger: An incoming pulse at either input produces a pulse at the output. (d) NDRO: the arrival of the epoch pulse to the SET input activates the "set" state and allows the propagation of subsequent READ pulses. A pulse at the RESET input clears the state. Thus, the NDRO acts as a filter. It returns to the "set" state at the arrival of the next epoch.

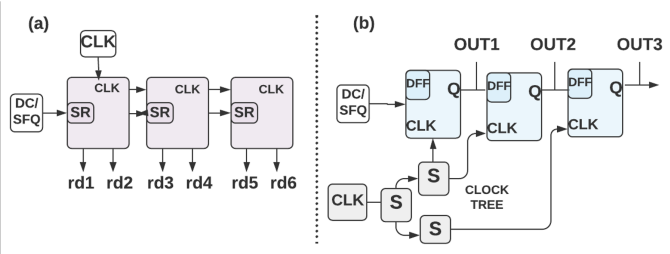

Fig. 9. Three-stage shift register: (a) Shuttle-flux based. (b) DFF-based.

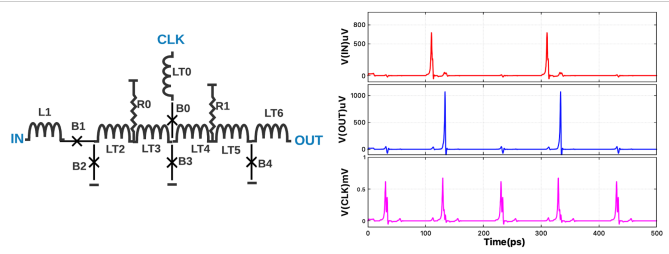

Fig. 10. Data flip-flop (DFF) circuit schematic and simulation waveform.

using a merger. A ring counter generates an accurately-timed sequence of pulses. A merger is illustrated in Fig. 7(b) [7] and allows SFQ pulses from either input IN1 or IN2 to propagate to a single output OUT1.

Inspired from our ring counter, we design a storage buffer that stores pulses and preserves their temporal information in RL (i.e., the delay from the start of the epoch) for any number of epochs as dictated by control inputs using our SF-SR, as illustrated in Fig. 12(b). In order words, this circuit can store RL pulses for any number of epochs, and release the pulse

TABLE V
A COMPARISON OF DIFFERENT SHIFT REGISTERS AS REPORTED IN THE LITERATURE.

Reference	Circuit structure	JJ count per cell	Power	Operating margin	Frequency	Technology
[27]	Shuttle-flux	5 JJ	70uW	+/-10	10 GHz	$Nb/Al_2O_3/Nb$
[36]	Shuttle-flux	4 JJ	9uW	+/- 19 – 24%	2 GHz	200nm YBCO
[11]	RSFQ	9 JJ	4uW	+/- 15%	12 GHz	HYPES 10-level-niobium
[21]	RSFQ	8 JJ	1mW	+/- 14%	20 GHz	$Nb/AlO_3/Nb$
[22]	QFP	8 JJ	10nW	+/- 32%	10 GHz	$Nb/Al_2O_3/Nb$
Our Design	Shuttle-flux	4 JJ	3.6uW	+/- 34%	10 GHz	MITLL SFQ5ee

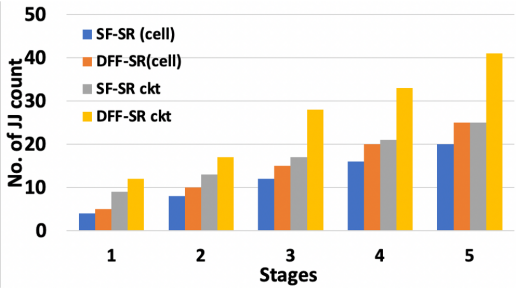

Fig. 11. Area evaluation: Number of JJs for a different number of stages.

in the same time slot of a future epoch, as determined by control inputs. The SF-SR can store multiple RL pulses as long as one cell contains at most one pulse. To that end, we add a nondestructive read-out (NDRO) "feedback" cell in the feedback loop to control the propagation of data pulses from the SF-SR's output back to its input. This NDRO controls whether pulses reaching the output of the SF-SR are looped back to the input in order to re-enter the SF-SR and thus remain stored. The circuit schematic of an NDRO is shown in Fig. 7(c). The NDRO's input ports are READ, SET, and RESET. The READ input comes from the SF-SR's output and carries data pulses back to the SF-SR's input. The NDRO's SET input enables this feedback loop by setting the NDRO to propagate any future pulses from its READ input to its output (OUT). The NDRO's RESET input reverts the NDRO back to its cleared state, where data pulses that arrive at the READ input port do not get propagated to the NDRO's output. This effectively disables the feedback loop so that pulses departing the SF-SR do not re-enter the SF-SR. SET and RESET are control inputs.

At the SF-SR's output we add a splitter (Fig. 7(a)) and another "output" NDRO in order to control when pulses in the SF-SR propagate to the final output after they traverse the last stage of the shift register. Using these two NDROs, our shift register can hold its contents in a loop, output its contents to the output, or both. When pulses remain in the loop, the relative delay between pulses is preserved. Also, since control signals setting or resetting the two NDROs arrive at the beginning of the epoch, we also preserve the relative timing of pulses in relation to the beginning of the epoch. Therefore, our buffer can store RL pulses for any number of epochs while preserving each pulse's value in RL.

To drain pulses stored in the SF-SR, we reset the feedback NDRO and set the draining NDRO at the arrival of the next epoch. In the latter case, any pulses exit the SF-SR from left to right while maintaining their RL timing T, as shown in Fig. 14. In contrast, when we want the buffer to preserve the pulses in the SF-SR, we reset the output NDRO and select the feedback NDRO. In that case, the SF-SR is essentially part of a loop that keeps pulses going around.

To provide a concrete comparison, we define an RL epoch that consists of 32 time slots (5 bits of an equivalent binary representation), as shown in Fig. 12(c). Therefore, each pulse can represent a value from 0 to 31. We can divide one epoch into 32 RL time slots of 100ps each for a total epoch duration of 3200ps. Each epoch holds one RL pulse, allowing us to configure the SF-SR with a single cell. In Fig. 14 and in order to reduce area, we configure our SF-SR to have four shift stages set at 100ps each. To be able to hold the 3200ps-long epoch, the SF-SR loops back 7 times. It is important to note that the feedback pulse should not overlap with the previous pulse to avoid collisions and maintain the shift interval as the pulse loops back. We need a four-stage DFF-based shift register to implement the same specifications using DFFs. However, our SF-SR for this example only requires a pair of JJs and coupled inductors per shifting interval, which results in 20% fewer JJs than the DFF-based shift register.

The restriction of only one pulse per epoch stems from configuring an SR ring counter with one cell. Instead, we can configure our SF-SR to have as many stages as time slots in an RL epoch. This allows the buffer to store multiple RL pulses per epoch, up to one pulse per time slot. This way, we can store multiple RL pulses without having them interact. We can also store pulse trains that typically consist of multiple pulses or even a serialized binary representation. While this requires more stages, our SF-SR makes this approach much more practical because of its favorable scaling characteristics compared to a DFF-based shift register.

A. *Shift Buffer Evaluation*

Our ring-based, SF-SR RL buffer is particularly useful to temporarily store information in RL without resorting to expensive conversions to and from binary representation. Using a single DFF per epoch would discard the temporal information that RL relies on to represent values [14]. Our ring counter based on our SF-SR is the most area-efficient way to store RL pulses and preserve their temporal information since the number of time slots in an RL epoch is proportional to the number of shifting intervals in the shift register. Therefore, using prior-art shift registers would require more JJs. A projection graph is shown in Fig. 15. The shift buffer circuit of Fig. 12(b) has a total JJ count of 56 JJs for a four-stage

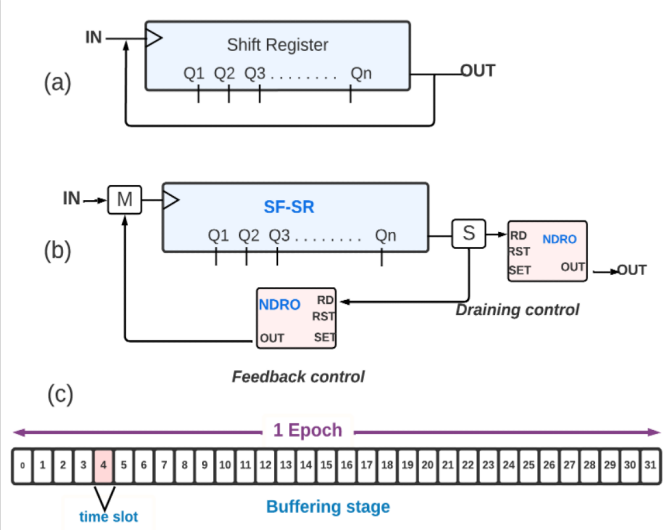

Fig. 12. (a) A simple ring counter based on a shift register (SR). (b) A storage buffer for RL pulses based on our shift register (SF-SR). (c) An RL epoch with 32 RL time slots.

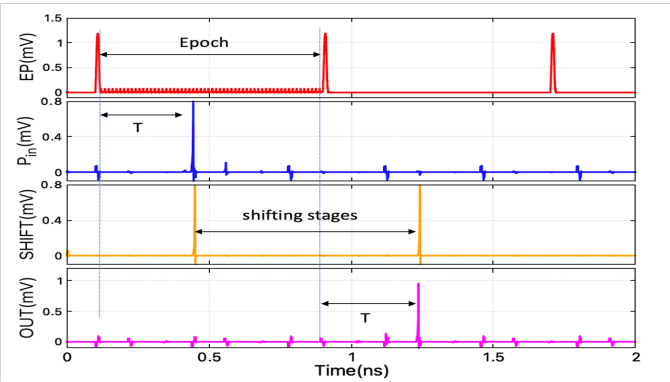

Fig. 13. Simulation waveform of the SF-SR-based buffer (Fig. 12). Input pulses are shown in blue. One pulse arrives at time slot L after time T from the beginning of the epoch. The number of shifting stages is equal to the number of time slots in an epoch. The last stage of the SF-SR routes the output pulse that will arrive at the same time slot in the next epoch, thus preserving the input pulse's value in RL.

SF-SR. Finally, Table VI, summarizes the JJ counts of each building block used to construct the SF-SR-based buffer. If we scale to 1024 stages, our SF-SR would have 4096 JJs, while a DFF-based shift register would have 6656 JJs. This means 38% fewer JJs for our shift register-based buffer.

TABLE VI
FOUR-STAGE SR JJ COUNT BREAKDOWN BY SUBCIRCUIT.
*SHUTTLE-FLUX SHIFT REGISTER

Sub-circuit	*SF-SR based	DFF-based
Shift register	12	24
DC-SFQ	4	8
Mergers	5	5
JTL	4	21
NDROs	28	28
Splitters	3	30
Total JJs	**56**	**97**
Power μW	**38**	**144**

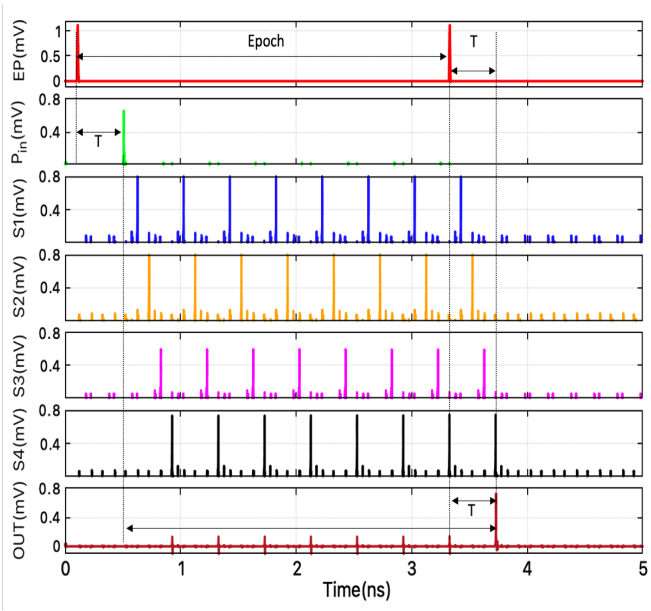

Fig. 14. Simulation waveform of the SF-SR-based buffer with feedback control using a four-stage SR. The input pulse is shown in green and arrives time T after the beginning of the epoch. The shifted pulses corresponding to the four stages of the SF-SR are S1, S2, S3, and S4. The shifting interval is set at 100ps and the loop-back delay for the four-stage shifting is 400ps. The SF-SR acts as a circulating counter. The feedback NDRO in Fig. 12 controls the number of pulses that loop back to the SF-SR's input. In this example, the loop-back NDRO is reset at the beginning of the next epoch (arrival of the epoch pulse) while the draining NDRO is set. Therefore, the draining NDRO propagates pulses from the SF-SR's output to the overall buffer circuit's output (OUT).

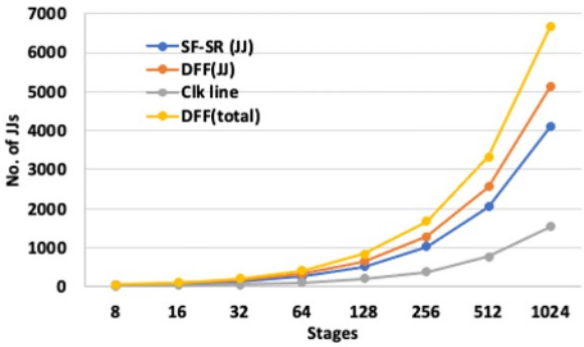

Fig. 15. Analytically-derived JJ counts for a variable number of shifting stages. The blue line is the number of JJs for our shuttle-flux shift register (SF-SR). The orange line is for a DFF-based shift register's data path, the gray line for the clock distribution lines and splitters of the DFF-based shift register, and the yellow line is the DFF shift register's total JJ count (DFF(JJ) + Clk line).

V. APPLICATION II: PSEUDO-RANDOM NUMBER GENERATOR

A pseudo-random number generator (PRNG) produces a sequence of pseudo-random numbers suitable for applications where random numbers are required in a deterministic or non-deterministic manner. Such applications include Monte Carlo analysis [38], test pattern generation for ASIC testing, and modeling [39]. A characteristic of a PRNG is that it can efficiently produce pseudo-random numbers in a short time. In addition, it is deterministic, where a given sequence

of numbers can be produced at any time and are easy to replay. Finally, a PRNG is periodic, meaning that the sequence eventually repeats.

One popular method to implement a PRNG is a linear-feedback shift register (LFSR) whose input bit is a linear function of its previous state [40]–[42]. Some of the output bits are combined in an exclusive-OR (XOR) gate to form a feedback mechanism, as shown in Fig. 16. The only periodic signal necessary to generate the test pattern is a clock. The LFSR produces a maximum number of $2^n - 1$ pseudo-random possible patterns, where n is the number of shift register stages.

Our LFSR implementation is based on our SF-SR. The input is a DC signal that is then converted to an SFQ pulse through a DC-SFQ converter circuit. The SFQ input pulse and the XOR'ed feedback pulse are the two inputs to a merger. The output of the merger is the input to the N-stage SF-SR. The output of the shift register is routed to a splitter, where one output is the input to the feedback XOR and the other output is the next pulse in the sequence of pseudo-randomly-generated output pulses. The simulation from Fig. 16 is shown in Fig. 17.

An example of a DFF-based PRNG is presented in [43]. In that implementation, input pulses to the DFF-based shift register are routed to the data input "D-line" and the clock input to the input "C-line". Clock and data pulses propagate through the shift register. Each D-line and C-line uses a resistor bias. When we scale up the circuit, this bias will likely have a higher power consumption compared to our SF-SR, where the bias resistor is only at the three clock lines. The number of resistors is only three in our implementation while in a DFF-based shift register the number of resistors scales up proportionally to the number of stages. Table VII shows that the JJ count of a PRNG with a four-stage LFSR using our proposed SF-SR is 55% lower in comparison to a DFF-based design.

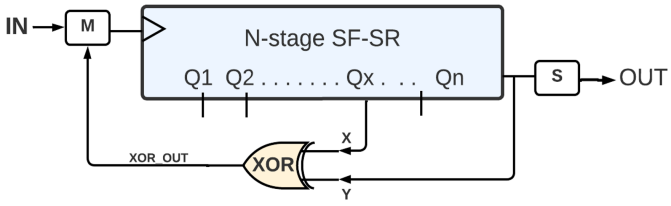

Fig. 16. Block diagram of a linear feedback shift register (LSFR) used to implement a pseudo-random number generator (PRNG). The XOR acts as the mixing element. The two inputs to the XOR are from the last and intermediate registers, respectively. The XOR's output (XOR_{out}) is connected to the SF-SR's input to form a feedback loop.

VI. Application III: Delay Element With Parallel SF-SRs

The simple structure of shift registers and low JJ counts of our SF-SR motivate designing further memory elements for temporary data storage at high speeds. However, shift register-based memories such as those based on DFFs traditionally suffer from large area overheads.

For instance, a data-driven self-timed (DDST) shift register presented in [44] uses a D3 flip-flop (D3FF) where each

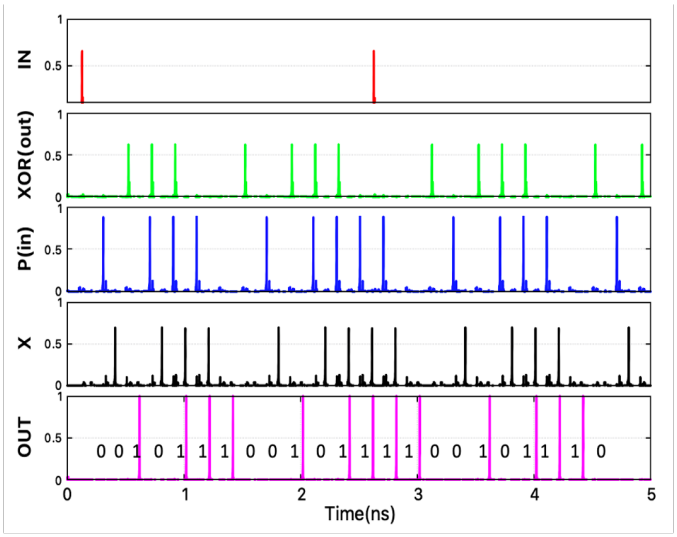

Fig. 17. Simulation waveform of a pseudo-random number generator (PRNG) using our shuttle-flux shift register (SF-SR) as an LSFR. IN is the input pulse converted from DC-SFQ. The mixing element's (XOR's) inputs are from the intermediate register X and the last register OUT. The signal XOR_{OUT} is the output from the XOR that is connected back to the first stage via a merger to form a feedback loop. OUT is the generated random sequence ([OUT = 001011100101111.....]).

TABLE VII
JJ COUNT AND POWER DISSIPATION (STATIC + DYNAMIC) COMPARISON FOR A PRNG WITH A FOUR-STAGE LSFR. POWER IS CALCULATED BY MEASURING CURRENT DRAW IN WRSPICE.

Sub-circuit	SF-SR-based	DFF-based	[43]
SR	12	24	32
DC-SFQ	4	8	5
Merger	5	5	-
XOR	9	9	9
Splitter	3	30	30
JTL	2	2	6
Total JJs	**35**	**78**	**82**
Power (μW)	**35**	**106**	**120**

module has dual-rail inputs and outputs. The D3FF is a non-destructive memory cell that is a modified version of a D2FF [45]. It has two data inputs, a clock input, and an output. The D3FF has 11 JJs, while a typical DFF has 5 JJs.

Another example of a memory-based shift register is presented in [11], [12]. Its implementation is based on a shift register cell that consists of a pair of JJs, as well as storage, and non-storage inductors. The cells are adjacent to each other, and the clock traverses a series of JTLs and a resistor to provide a delay to each shifting cell. The drawback of this structure is the presence of a resistive delay element that increases its power consumption.

Along similar lines, here we implement two variations of a delay element that consists of an array of N SF-SRs in parallel of M stages each. In the first variation of Fig. 18(a), input data pulses are routed to each shift register using a balancer tree [15] that distributes input pulses from its single input to all its outputs in a round-robin manner. For our demonstration, we use a TFF3-based balancer tree (three outputs). Therefore, incoming pulses are distributed evenly among the three parallel SF-SRs. On the output side, a merger tree is used to drain

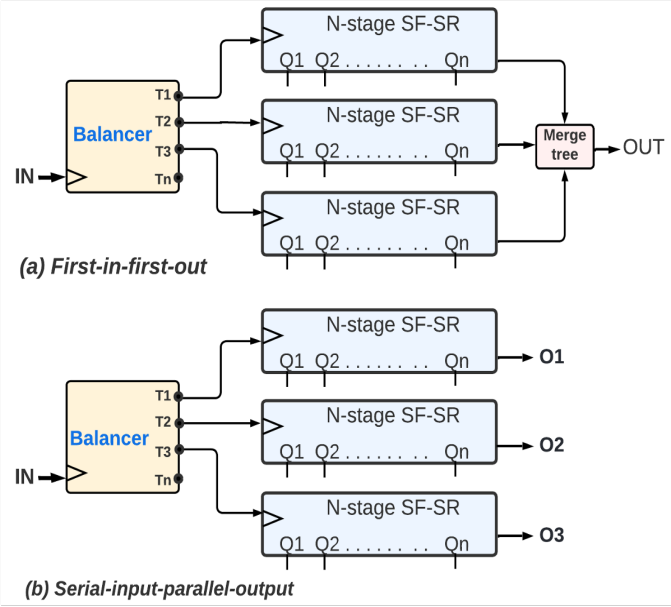

Fig. 18. (a) Block diagram of a delay element that functions as a First-in-first-out (FIFO) shift register. Each SF-SR holds a train of pulses. At the input, a balancer (TFF3) distributes incoming pulses among the parallel SF-SRs. The output is accumulated by a merger tree. (b) Block diagram of a delay element that functions as a serial-in-parallel-out shift register. Similarly, a balancer (TFF3) distributes the incoming pulses. Each shift register can be configured to output their pulses at the same or different times.

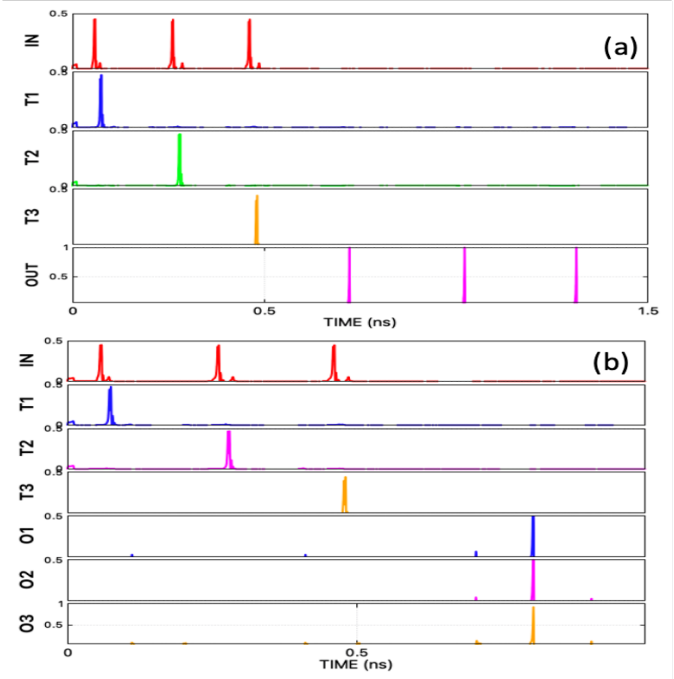

Fig. 19. Simulation waveform of the shift register-based delay element with parallel SF-SRs. (a) Shows a waveform that corresponds to Fig. 18(a). Incoming pulses from input IN are distributed to each shift register (T1, T2, and T3). Pulses departing any of the three parallel SF-SRs are routed to the output (OUT) through the merger tree. (b) This waveform corresponds to Fig. 18(b), where there is no merger tree. Thus, each SF-SR directly drives its own output. In this example, the three SF-SRs are configured to have different shifting lengths allowing the pulses to arrive simultaneously.

the data output pulses from each parallel SF-SR. With this arrangement, the circuit is functionally equivalent to a single SF-SR (first-in-first-out order of pulses) clocked at N times higher frequency than each of the N parallel SF-SRs. This way, we can achieve the functional equivalent of SF-SRs that is clocked faster than what a single SF-SR can correctly operate at.

In Fig. 18(b), we implement a serial-in parallel-out (SIPO) shift register. Similarly, the TFF3-based balancer tree distributes pulses from its single input across all its outputs in a round-robin manner. We can configure a different delay for the second and third SF-SR or clock all three SF-SRs at the same time after each SF-SR has a chance to receive an input pulse. In that case, the parallel SF-SRs produce output pulses at the same time. This circuit can therefore function as a deserializer that converts a serial train of pulses to an equivalent parallel representation. Additionally, this circuit can also be used as a data alignment unit (DAU) that forwards data to multiple processing elements (PEs) at the same or other pre-defined times. In the latter case, the parallel SF-SR can have different delay lengths. DAUs are useful for applications with systolic array networks and 2D-convolutional operations.

In future implementations, we can consider a demultiplexer instead of a balancer to enable flexible addressing that allows the circuit to operate as addressable memory. Since our SF-SR only uses three bias resistors at the clock, it has a lower power consumption at every stage than [11], [12] and fewer JJs than [45].

VII. Area and Power Evaluation

Table VIII shows an area and power evaluation of the building block circuits used to implement the three applications we previously described. Power is separated in static and dynamic. For dynamic power, we assume that each JJ switches every time slot. Static power is mostly dissipated at the bias resistors and can be calculated as $P_s = I_b * V_b$, where I_b is the current across the bias resistor, and V_b is the voltage input. Dynamic power is due to the switching of the JJs and can be calculated as $P_d = f\Phi_0 I_c$, where f is the switching frequency, I_c is the critical current and Φ_0 is the quantum flux constant.

RSFQ circuits have lower switching energy compared to CMOS. However, RSFQ circuits dissipate higher static power. In RSFQ, most of the static power is dissipated at the bias resistors because of the constant power drawn through them, while dynamic power is typically in the order of nanowatts and is dissipated during switching across JJs. An alternative approach to eliminate the static power dissipated in bias resistors is to use ERSFQ [46] or eSFQ [47] instead of RSFQ. In ERSFQ and eSFQ, the DC bias is preserved, while bias resistors are replaced with series inductors and JJs that regulate the DC bias current into the rest of the circuit [48]. However, both eSFQ and ERSFQ impose an area penalty due to the additional JJs and inductors in the biasing network.

We evaluate area in terms of the number of JJs. Using a modern SFQ process JJ model, we estimate the area of each JJ as a function of its critical current that is determined by

the relationship $I_c = J_c * A_j$, where A_j is the area of the JJ and J_c is the critical current density which is determined by the fabrication process. We also consider the area of the shunt resistance A_{Rsh}, thus $JJ_{area} = A_j + A_{Rsh}$. The inductor's physical size is not included in this area estimate. With this methodology, the single-stage SF-SR has an estimated area of $123\mu m^2$. The shift buffer (Section IV) occupies $778.57\mu m^2$, the PRNG (Section V) $546\mu m^2$, and the delay element with three parallel SF-SRs and a merger (Section VI) approximately $1060\mu m^2$.

TABLE VIII
METRIC EVALUATION OF INDIVIDUAL BUILDING BLOCK CIRCUITS.

Circuit	No. of JJs	Delay (ps)	Static Power (μW)	Dynamic Power (μW)
JTL	2	4	3.4	0.01
Merger	5	8	5.0	0.024
Splitter	3	2	5.66	0.01
Inverter	8	8	7.77	0.04
NDRO	10	9.5	7.62	0.04
DFF	6	4	3.68	0.02
DC-SFQ	4	-	3.7	0.0153
SR (cell)	5	8	3	0.072
XOR	9	8	1.74	0.036
TFF3	14	4	14	0.059

VIII. CONCLUSION

Temporal data encodings such as RL can mitigate the stringent area constraints of modern superconducting device technologies. In RL, information is encoded as a delay from a reference signal. Computation is performed by observing the relative propagation times of signals injected into a circuit. This paper presents a shuttle flux-based shift register (SF-SR) used to construct a storage buffer that correctly stores RL-encoded pulses by preserving each pulse's delay from a reference signal. By adding two non-destructive read-out (NDRO) cells, this SF-SR can preserve indefinitely RL pulses as well as pulses encoded differently, such as pulse trains for spiking neural networks. Based on our SF-SR, we also design a pseudo-random number generator (PNRG) and a delay element with parallel SF-SRs. Our SF-SR is more area efficient than state-of-the-art, thus making long RL epochs or storing long pulse trains practical.

ACKNOWLEDGMENTS

This work was supported by the Director, Office of Science, of the U.S. Department of Energy under Contract No. DE-AC02-05CH11231.

ACKNOWLEDGMENTS

This work was supported by the Director, Office of Science, of the U.S. Department of Energy under contract No. DE-AC02-05CH11231.

REFERENCES

[1] M. G. Bautista, P. Gonzalez-Guerrero, D. Lyles, and G. Michelogiannakis, "Superconducting shuttle-flux shift buffer for race logic," in *2021 IEEE International Midwest Symposium on Circuits and Systems (MWSCAS)*. IEEE, 2021, pp. 796–801.

[2] I. I. Soloviev, N. V. Klenov, S. V. Bakurskiy, M. Y. Kupriyanov, A. L. Gudkov, and A. S. Sidorenko, "Beyond moore's technologies: operation principles of a superconductor alternative," *Beilstein journal of nanotechnology*, vol. 8, no. 1, pp. 2689–2710, 2017.

[3] S. S. Tannu, P. Das, M. L. Lewis, R. Krick, D. M. Carmean, and M. K. Qureshi, "A case for superconducting accelerators," ser. CF, 2019.

[4] E. Debenedictis, J. Cook, T. S. Metodi, M. F. Hoemmen, M. Marinella, R. Schiek, and H. Zima, "Beyond moore's law and implications for computing in space." Sandia National Lab, Tech. Rep., 2015.

[5] Q. Xu, Y. Yamanashi, C. Ayala, N. Takeuchi, T. Ortlepp, and N. Yoshikawa, "Design of an extremely energy-efficient hardware algorithm using adiabatic superconductor logic," in *2015 15th International Superconductive Electronics Conference (ISEC)*. IEEE, 2015, pp. 1–3.

[6] M. A. Manheimer, "Cryogenic computing complexity program: Phase 1 introduction," *IEEE Transactions on Applied Superconductivity*, vol. 25, no. 3, pp. 1–4, 2015.

[7] K. K. Likharev and V. K. Semenov, "Rsfq logic/memory family: A new josephson-junction technology for sub-terahertz-clock-frequency digital systems," *IEEE Transactions on Applied Superconductivity*, vol. 1, no. 1, pp. 3–28, 1991.

[8] O. Mukhanov, V. Semenov, and K. Likharev, "Ultimate performance of the rsfq logic circuits," *IEEE Transactions on Magnetics*, vol. 23, no. 2, pp. 759–762, 1987.

[9] J. Hang and J. Przybysz, "Performance issues in single flux quantum shift registers," *IEEE transactions on applied superconductivity*, vol. 3, no. 1, pp. 2752–2755, 1993.

[10] O. A. Mukhanov, D. Gupta, A. M. Kadin, and V. K. Semenov, "Superconductor analog-to-digital converters," *Proceedings of the IEEE*, vol. 92, no. 10, pp. 1564–1584, 2004.

[11] O. A. Mukhanov, "Rsfq 1024-bit shift register for acquisition memory," *IEEE transactions on applied superconductivity*, vol. 3, no. 4, pp. 3102–3113, 1993.

[12] ——, "Rapid single flux quantum (rsfq) shift register family," *IEEE transactions on applied superconductivity*, vol. 3, no. 1, pp. 2578–2581, 1993.

[13] L. Zheng, "High-speed rapid-single-flux-quantum multiplexer and demultiplexer design and testing," CALIFORNIA UNIV BERKELEY GRADUATE DIV, Tech. Rep., 2007.

[14] G. Tzimpragos, D. Vasudevan, N. Tsiskaridze, G. Michelogiannakis, A. Madhavan, J. Volk, J. Shalf, and T. Sherwood, "A computational temporal logic for superconducting accelerators," in *Proceedings of the Twenty-Fifth International Conference on Architectural Support for Programming Languages and Operating Systems*, 2020.

[15] G. Michelogiannakis, D. Lyles, P. Gonzalez-Guerrero, M. Bautista, D. Vasudevan, and A. Butko, "Srnoc: A statically-scheduled circuit-switched superconducting race logic noc," in *2021 IEEE International Parallel and Distributed Processing Symposium (IPDPS)*, 2021, pp. 1046–1055.

[16] P. Gonzalez-Guerrero, M. G. Bautista, D. Lyles, and G. Michelogiannakis, "Temporal and sfq pulse-streams encoding for area-efficient superconducting accelerators," in *Proceedings of the 27th ACM International Conference on Architectural Support for Programming Languages and Operating Systems*, 2022, pp. 963–976.

[17] *Suny RSFQ Cell Library (pavel.physics.sunysb.edu/RSFQ/)*. [Online]. Available: http://www.physics.sunysb.edu/Physics/RSFQ/Lib/contents.html

[18] A. F. Murray and A. V. Smith, "Asynchronous vlsi neural networks using pulse-stream arithmetic," *IEEE Journal of Solid-State Circuits*, vol. 23, no. 3, pp. 688–697, 1988.

[19] S. V. Polonsky, J. C. Lin, and A. V. Rylyakov, "Rsfq arithmetic blocks for dsp applications," *IEEE Transactions on Applied Superconductivity*, vol. 5, no. 2, pp. 2823–2826, 1995.

[20] D. Y. Zinoviev, "Design issues in ultra-fast ultra-low-power superconductor batcher-banyan switching fabric based on rsfq logic/memory family," *Applied Superconductivity*, vol. 5, no. 7-12, pp. 235–239, 1997.

[21] O. A. Mukhanov, "Rsfq 1024-bit shift register for acquisition memory," *IEEE transactions on applied superconductivity*, vol. 3, no. 4, pp. 3102–3113, 1993.

[22] J. Casas, R. Kamikawai, and E. Goto, "Quantum flux parametron (qfp) shift registers clocked by an inductive power distribution network

and errorless operation of the qfp," *IEEE Transactions on applied superconductivity*, vol. 2, no. 1, pp. 26–32, 1992.
[23] T. Fulton, R. Dynes, and P. Anderson, "The flux shuttle—a josephson junction shift register employing single flux quanta," *Proceedings of the IEEE*, vol. 61, no. 1, pp. 28–35, 1973.
[24] P. W. Anderson, "How josephson discovered his effect," *Physics Today*, vol. 23, no. 11, p. 23, 1970.
[25] H. Beha, W. Jutzi, and G. Mischke, "Margins of a 16-ps/bit interferometer shift register," *IEEE Transactions on Electron Devices*, vol. 27, no. 10, pp. 1882–1887, 1980.
[26] H. L. Ko, "Analysis of the threshold characteristics of the uniform josephson flux shuttle," *IEEE Transactions on applied superconductivity*, vol. 1, no. 4, pp. 170–174, 1991.
[27] R. Lochschmied, R. Herwig, M. Neuhaus, and W. Jutzi, "A low power 12 bit flux shuttle shift register with nb technology," *IEEE transactions on applied superconductivity*, vol. 7, no. 2, pp. 2983–2986, 1997.
[28] A. M. Kadin, *Introduction to superconducting circuits*. Wiley-Interscience, 1999.
[29] I. Soloviev, S. Bakurskiy, V. Ruzhickiy, N. Klenov, M. Y. Kupriyanov, A. Golubov, O. Skryabina, and V. Stolyarov, "Miniaturization of josephson junctions for digital superconducting circuits," *Physical Review Applied*, vol. 16, no. 4, p. 044060, 2021.
[30] O. A. Mukhanov, "Energy-efficient single flux quantum technology," *IEEE Transactions on Applied Superconductivity*, vol. 21, no. 3, pp. 760–769, 2011.
[31] R. L. Kautz, "Picosecond pulses on superconducting striplines," *Journal of Applied Physics*, vol. 49, no. 1, pp. 308–314, 1978.
[32] K. Ishida, I. Byun, I. Nagaoka, K. Fukumitsu, M. Tanaka, S. Kawakami, T. Tanimoto, T. Ono, J. Kim, and K. Inoue, "SuperNPU: An extremely fast neural processing unit using superconducting logic devices," in *2020 53rd Annual IEEE/ACM International Symposium on Microarchitecture (MICRO)*. IEEE, 2020, pp. 58–72.
[33] F. Zokaee and L. Jiang, "Smart: A heterogeneous scratchpad memory architecture for superconductor sfq-based systolic cnn accelerators," *MICRO-54: 54th Annual IEEE/ACM International Symposium on Microarchitecture*, Oct 2021. [Online]. Available: http://dx.doi.org/10.1145/3466752.3480041
[34] B. D. Josephson, "Possible new effects in superconductive tunnelling," *Physics letters*, vol. 1, no. 7, pp. 251–253, 1962.
[35] J. C. Gallop, *SQUIDs, the Josephson effects and superconducting electronics*. CRC Press, 1991.
[36] R. Koch, T. Scherer, M. Winter, and W. Jutzi, "A 4 bit yba/sub 2/cu/sub 3/o/sub 7-/spl delta// bicrystal josephson junctions flux shuttle shift register," *IEEE Transactions on Applied Superconductivity*, vol. 7, no. 2, pp. 3646–3649, 1997.
[37] S. K. Tolpygo, V. Bolkhovsky, T. J. Weir, A. Wynn, D. E. Oates, L. M. Johnson, and M. A. Gouker, "Advanced fabrication processes for superconducting very large-scale integrated circuits," *IEEE Transactions on Applied Superconductivity*, vol. 26, no. 3, pp. 1–10, 2016.
[38] D. W. Hubbard, "A multi-dimensional, counter-based pseudo random number generator as a standard for monte carlo simulations," in *2019 Winter Simulation Conference (WSC)*, 2019, pp. 3064–3073.
[39] B. B. Youssef and R. Sammouda, "Pseudorandom number generation in the context of a 3d simulation model for tissue growth," *Procedia Computer Science*, vol. 29, pp. 2391–2400, 2014.
[40] D. B. Rita Mahajan, Komal Devi, "Review on standard and multibit linear feedback shift register," in *JETIR*, 2019.
[41] A. Kidiyarova-Shevchenko and D. Zinoviev, "RSFQ pseudo random generator and its possible applications," *IEEE Transactions on Applied Superconductivity*, vol. 5, no. 2, pp. 2820–2822, 1995.
[42] A. Y. Kidiyarova-Shevchenko and D. Y. Zinoviev, "Rsfq pseudo random generator and its possible applications," *IEEE Transactions on Applied Superconductivity*, vol. 5, no. 2, pp. 2820–2822, 1995.
[43] W. Kessel, F.-I. Buchholz, M. I. Khabipov, R. Dolata, and J. Niemeyer, "Development of a rapid-single-flux-quantum shift register for applications in rf noise power metrology," *IEEE transactions on instrumentation and measurement*, vol. 46, no. 2, pp. 477–481, 1997.
[44] K. Fujiwara, H. Hoshina, Y. Yamashiro, and N. Yoshikawa, "Design and component test of sfq shift register memories," *IEEE transactions on applied superconductivity*, vol. 13, no. 2, pp. 555–558, 2003.
[45] K. Fujiwara, H. Hoshina, J. Koshiyama, and N. Yoshikawa, "Design and component test of rsfq packet decoders for shift register memories," *Physica C: Superconductivity*, vol. 378, pp. 1475–1480, 2002.
[46] O. A. Mukhanov, "Energy-efficient single flux quantum technology," *IEEE Transactions on Applied Superconductivity*, vol. 21, no. 3, pp. 760–769, 2011.
[47] M. H. Volkmann, A. Sahu, C. J. Fourie, and O. A. Mukhanov, "Experimental investigation of energy-efficient digital circuits based on esfq logic," *IEEE Transactions on Applied Superconductivity*, vol. 23, no. 3, pp. 1 301 505–1 301 505, 2013.
[48] ——, "Implementation of energy efficient single flux quantum digital circuits with sub-aj/bit operation," *Superconductor Science and Technology*, vol. 26, no. 1, p. 015002, 2012.

IX. Author Biographies

Meriam Gay Bautista received her Ph.D. in Electronics from the University of Technology Sydney, Australia (2019). Her doctoral thesis focused on the miniaturization of passive integrated circuits using Si-based technology for microwave and millimeter wave radio frequency systems. She received her MS in Microelectronics from National Taipei University, Taiwan (2013), and BS in Electronics Engineering from Mindanao State University – Iligan Institute of Technology, Philippines (2008). Her research interest includes mixed-signal design, RF/mmIC, superconducting circuits, quantum circuits, and race logic.

Patrica Gonzalez-Guerrero received her Ph.D. (2019) and M.Sc. (2015) from the University of Virginia, Charlottesville, VA, USA; and her Bachelors Degree from the Pontifical Xavierian University (2008), Bogota, Colombia. She also worked as an ASIC Design and Verification Engineer in Hewlett-Packard, Costa Rica. Her research interests include ultra-low power VLSI digital and mixed-signal design. Her work focuses on non-conventional computing paradigms to reduce power and energy consumption, such as synchronous and asynchronous stochastic computing, computing with sigma-delta streams, and race logic.

Darren Lyles graduated from the University of California, Berkeley with a degree in Electrical Engineering and Computer Science (EECS). Prior to joining LBNL, he worked on designing, verifying, and evaluating various Field Programmable Gate Array (FPGA) components. His current work focuses on the design, testing, verification, and application of superconducting race logic circuits and the design and testing of Electronic Design Automation (EDA) tools that support superconducting designs.

George Michelogiannakis is a research scientist for the computer architecture group (CAG) in the computational research division (CRD). He has extensive work on networking (both off- and on-chip) and computer architecture. His latest work focuses on the post-Moore's law era looking into specialization, emerging devices (transistors), memories, photonics, and 3D integration. He is also currently working on optics and architecture for HPC and data center networks.