Scalable true random number generator using adiabatic superconductor logic

Wenhui Luo¹, Olivia Chen², Nobuyuki Yoshikawa¹,³ & Naoki Takeuchi⁴*

Alternative computing such as stochastic computing and bio-inspired computing holds promise for overcoming the limitations of von Neumann computers. However, one difficulty in the implementation of such alternative computing is the need for a large number of random bits at the same time. To address this issue, we propose a scalable true-random-number generating scheme that we refer to as XORing shift registers (XSR). XSR generates multiple uncorrelated true random bitstreams using only two true random number generators as entropy sources and can thus be implemented by a variety of logic devices. Toward superconducting alternative computing, we implement XSR using an energy-efficient superconductor logic family, adiabatic quantum-flux-parametron (AQFP) logic. Furthermore, to demonstrate its performance, we design and observe an AQFP-based XSR circuit that generates four random bitstreams in parallel. The results of the experiment confirm that the bitstreams generated by the XSR circuit exhibit no autocorrelation and that there is no correlation between the bitstreams.

General-purpose von Neumann computers are approaching their performance limit due to the end of Moore's law and Dennard scaling. As a consequence, alternative computing technologies are being extensively investigated. Of note is the fact that many of the alternative computing technologies (e.g., stochastic computing, simulated annealing, bio-inspired computing, and invertible logic) exploit stochastic operations to gain computing performance and require many random bits at the same time. Hence, in order to build computing systems based on alternative computing, the question of how to generate and distribute multiple random bitstreams in parallel in the entire system is of significant concern. In the case of typical semiconductor circuitry, a random bitstream is generated by a pseudo-random number generator (PRNG) such as a linear-feedback shift register, and a large number of PRNGs are used to generate multiple random bitstreams in parallel, resulting in high energy and hardware overhead. For example, in IBM's neuromorphic system TrueNorth, 27% of the logic gates in each neuron circuit are used for a PRNG. Furthermore, since a bitstream generated by a PRNG has a finite length, the seed (i.e., initial state) of each PRNG needs to be carefully selected to avoid correlation between bitstreams, which can adversely affect the accuracy of computation. Therefore, to implement alternative computing, logic devices that can generate uncorrelated true random bitstreams in an energy- and hardware-efficient manner are required.

In recent years, we have been developing adiabatic quantum-flux-parametron (AQFP) logic. The AQFP is an energy-efficient logic device based on the quantum flux parametron that can operate with extremely small energy dissipation near the thermodynamic limit due to an energy-efficient switching scheme, adiabatic switching. Moreover, the AQFP can easily perform stochastic operations by thermal fluctuations. As part of our effort, we have developed a true random number generator (TRNG) using AQFP logic and demonstrated the generation of a low-autocorrelation random bitstream. Thus, AQFP logic appears highly suitable as a building block for implementing alternative computing. The next step towards large-scale alternative computing-based systems is to develop a scheme to generate multiple random bitstreams in a scalable way using AQFP logic.

In the present study, we propose a scalable true-random-number generating scheme that we refer to as XORing shift registers (XSR) and implement XSR using AQFP logic. XSR generates multiple uncorrelated true random bitstreams in parallel using only two TRNGs as entropy sources. This is a huge advantage in the development of large-scale systems. In general, a TRNG is a somewhat complex circuit (various TRNGs can be found in the literature), and minimizing their number is highly desirable. Since XSR utilizes XOR gates to generate...
multiple random bitstreams, we first explain random number generation using XOR gates. We then explain the operating principle of XSR and how to implement XSR using AQFP logic. Finally, we experimentally demonstrate an AQFP-based XSR circuit that generates four random bitstreams, each of which has no autocorrelation or correlation with the other bitstreams. Our results indicate the path towards scalable, energy-efficient alternative computing-based systems using AQFP logic.

### Random number generation using XOR gates

Table 1 describes the truth table of an XOR gate, where \( A \) and \( B \) are the inputs and \( X (= A \oplus B) \) is the output. Hereafter \( A \) and \( B \) are assumed to be uncorrelated random bits (a random bit becomes a 0 or 1 stochastically with the same probability).

**Logical viewpoint.** Most importantly, XORing two random bits generates another random bit as follows: \( A \) and \( B \) are random bits, so that the four possible input combinations \( [(A, B) \in \{(0, 0), (0, 1), (1, 0), (1, 1)\}] \) appear randomly. Consequently, \( X \) becomes 0 or 1 randomly (i.e., \( X \) is also a random bit), since \( X \) includes the same number of 0 s as the number of 1 s in the truth table. Here we discuss the correlation regarding \( A, B, \) and \( X \) by calculating mutual information, which quantifies the correlation between probability variables. The mutual information between \( A \) and \( X \) is given by

\[
I(A; X) = H(A) + H(X) - H(A, X)
\]

where \( H(A) \) and \( H(X) \) are the logical entropy (i.e., Shannon entropy of the logic states) of \( A \) and \( X \), respectively, and \( H(A, X) \) is the joint logical entropy of \( A \) and \( X \). \( H(A) \) is given by

\[
H(A) = -\sum_a P(a)\log P(a)
\]

where \( A \) takes a value \( a \) with the probability \( P(a) \). According to Table 1, \( a \in \{0, 1\} \) and \( P(0) = P(1) = 0.5 \), which gives \( H(A) = \ln 2 \). Similarly, \( H(X) = -\sum_a P(x)\log P(x) = \ln 2 \), and \( H(A, X) = -\sum_a P(a, x)\log P(a, x) = 2\ln 2 \). As a result, \( I(A; X) = \ln 2 + \ln 2 - 2\ln 2 = 0 \), which indicates that \( A \) and \( X \) are not correlated with each other, i.e., one cannot tell the value of \( A \) from a given value of \( X \), and vice versa. Likewise, \( I(B; X) = 0 \) and \( I(A; B) = 0 \). Therefore, there is no correlation between any pair of \( A, B, \) and \( X \). However, \( A, B, \) and \( X \) are correlated since, if one knows the values of any two of the three \( (A, B, \) and \( X) \), one can tell the value of the other. This is quantified by the mutual information among \( A, B, \) and \( X \) as follows:

\[
I(A; B; X) = H(A) + H(B) + H(X) - H(A, B) - H(A, X) - H(B, X) + H(A, B, X)
\]

(3)

\[
H(A) = H(B) = H(X) = \ln 2, \text{ and } H(A, B) = H(A, X) = H(B, X) = H(A, B, X) = 2\ln 2.
\]

Consequently, \( I(A; B; X) = -\ln 2 \). The above discussion indicates that an XOR gate can increase two uncorrelated random bits \( (A \) and \( B) \) to three uncorrelated random bits \( (A, B, \) and \( X) \), where correlation appears only when all of \( A, B, \) and \( X \) are taken into account together.

**Thermodynamic viewpoint.** In physical systems, random number generation is related to thermodynamics because logical entropy is tied to (thermodynamic) entropy: in the quasi-static limit, \( \Delta S = \beta Q \), where \( \Delta H \) is the logical entropy change of the system, \( \Delta S \) is the entropy change of the system, \( \beta \) is inverse temperature, and \( Q \) is the heat absorbed by the system. For instance, an AQFP TRNG generates a random bit (i.e., \( \Delta H = \ln 2 \)) by increasing entropy via heat absorption (i.e., \( \Delta S = \beta Q = \ln 2 \)). Thus, we explore random number generation using XOR gates from the thermodynamic viewpoint.

We first derive the thermodynamic relations for a logic gate with two uncorrelated random inputs \( (A \) and \( B) \) and an output \( (X) \). From Eq. (3), the total logical entropy change during a logic operation is given by

\[
\Delta H(A, B, X) = \Delta H(A, B) + \Delta H(A, X) + \Delta H(B, X) - \Delta H(A) - \Delta H(B) - \Delta H(X) + \Delta I(A; B; X)
\]

(4)

The inputs do not change during a logic operation, so that \( \Delta H(A) = \Delta H(B) = \Delta H(A, B) = 0 \). Moreover, the total logical entropy change is linked to heat absorption. Hence, in the quasi-static limit (i.e., assuming that the logic operation is performed without energy dissipation), Eq. (4) becomes

\[
\Delta H(A, B, X) = \Delta H_{\text{eff}}(X) + \Delta I(A; B; X) = \beta Q
\]

(5)

| A | B | X = A \oplus B |
|---|---|----------------|
| 0 | 0 | 0              |
| 0 | 1 | 1              |
| 1 | 0 | 1              |
| 1 | 1 | 0              |

Table 1. Truth table of an XOR gate.
where $\Delta H_{\text{eff}}(X) = \Delta H(A, X) + \Delta H(B, X) - \Delta H(X)$ is the effective logical entropy change of $X$; $\Delta H(A, X)$ becomes $\ln 2$ when $X$ is a random bit that is not correlated with $A$ or $B$. Conventional logic gates operate deterministically and do not include entropy-increasing processes such as heat absorption; thus, $Q = 0$ and Eq. (5) is reduced to

$$\Delta H_{\text{eff}}(X) = -\Delta I(A; B; X)$$ (6)

This equation shows that even if a logic gate does not include entropy-increasing processes, the logic gate can generate a random bit by producing mutual information.

Figure 1 shows the change in logical entropy and mutual information regarding an XOR gate. (a) Initial state, where the output $X$ is not generated. (b) Final state, where $X$ (a random bit) is generated by producing mutual information $I(A; B; X)$.

XORing shift registers (XSR)

XSR generates multiple random bitstreams in parallel based on random number generation using XOR gates. Figure 2 shows XSR generating $n$ random bitstreams ($n \in \mathbb{N}$). The clock lines to the flip-flops are omitted for simplicity, and $t$ is time in clock cycles. As indicated, XSR involves only simple circuits: two uncorrelated TRNGs
(TRNGs A and B), two \((n+1)\)-bit shift registers (shift registers A and B), and \(n\) XOR gates (XOR 1, XOR 2, ..., XOR \(n\)). Shift register A transmits the random bitstream from TRNG A \([A(t), A(t-1), \ldots, A(t-n-1)]\), whereas shift register B transmits the random bitstream from TRNG B \([B(t), B(t-1), \ldots, B(t-n-1)]\). The \(n\) XOR gates produce \(n\) random bits \([X_1(t), X_2(t), \ldots, X_n(t)]\) in parallel, and each XOR gate generates a random bitstream; for instance, XOR 1 produces a random bitstream of \(X_1(t), X_1(t-1), X_1(t-2), \ldots\), and so forth.

The outputs of the XOR gates can be described as follows: First, the output \(X_i(t)\) \((i \in \{1, 2, \ldots, n\})\) from each XOR gate is a random bit since, as mentioned above, XORing two random bits produces another random bit; for instance, \(X_i(t)\) is produced by XORing two random bits \(A(t-1)\) and \(B(t-n)\) and is thus a random bit. Furthermore, the output bitstream from each XOR gate \([X_1(t), X_1(t-1), X_1(t-2), \ldots]\) does not exhibit autocorrelation because each output in the bitstream is generated from different random bit pairs; for instance, \(X_1(t)\) is generated from \(A(t-1)\) and \(B(t-n)\) whereas \(X_1(t-1)\) is generated from \(A(t-2)\) and \(B(t-n-1)\), so that \(X_1(t)\) is not correlated with \(X_1(t-1)\). Moreover, there is no correlation between the output bitstreams from different XOR gates. The important thing is that none of the outputs from the XOR gates are generated from the same random bit pairs, and that the output of an XOR gate is correlated with the inputs only when both inputs are taken into account; thus, correlation does not appear even if some outputs share the same random bit as an input. For instance, \(X_1(t)\) is generated from \(A(t-1)\) and \(B(t-n)\), and \(X_1(t-1)\) is generated from \(A(t-3)\) and \(B(t-n)\); i.e., \(X_1(t)\) and \(X_1(t-1)\) share \(B(t-n)\) as an input. However, \(X_1(t)\) is not correlated with \(X_1(t-1)\) because \(B(t-n)\) is not correlated with \(X_1(t)\) or \(X_1(t-1)\).

The above discussion establishes that each XOR gate in XSR generates a random bitstream without autocorrelation, and that there is no correlation between the random bitstreams from different XOR gates. Thus, XSR can generate many uncorrelated true random bitstreams in parallel using only two TRNGs. Moreover, since XSR utilizes simple logic gates, it can be easily implemented by various logic devices, including conventional semiconductor devices and emerging devices such as superconductor logic families. This is a significant advantage over the previously reported scheme\(^{37}\), which distributes random bits using asynchronous data collision with a careful timing design.

Note that XSR can be implemented using only one TRNG; for instance, XSR operates when TRNG B is removed and shift registers A and B are connected to each other. However, in this case, the autocorrelation of the TRNG may affect the quality of the generated random bit streams. Moreover, in general it is difficult to completely remove the autocorrelation of a TRNG\(^{27}\). Therefore, we decided to use two TRNGs in the present study.

**XSR using AQFP logic**

We implement XSR using AQFP logic. AQFP logic gates are powered and clocked by ac excitation currents, so that special clocking schemes\(^{38,39}\) are needed to operate AQFP circuits. In the present study, we use the most common clocking scheme, four-phase clocking\(^{38}\), to operate the AQFP circuits. Figure 3a shows an example of an AQFP-based XSR circuit for \(n = 4\). The entire circuit is clocked by the paired excitation currents, \(I_q\) and \(I_o\), with a phase separation of 90°. Logic operations are performed along the excitation phases, \(\phi_i\) through \(\phi_4\), with a phase separation of 90°. Consequently, the circuit shown in Fig. 3a operates in the same manner as that shown in Fig. 2, i.e., each XOR gate generates an uncorrelated random bit \(X_i\) \((i \in \{1, 2, 3, 4\})\) at every clock (excitation) cycle.

The circuit blocks can be explained as follows: A shift register is a buffer chain with feedback lines from \(\phi_4\) to \(\phi_1\), which enable data to transmit through a shift register in synchronization with the excitation phases. As
shown in Fig. 3b, a TRNG\textsuperscript{22} is a buffer chain without an input. The first buffer generates a random bit at every clock cycle because no input signal is applied and the logic state is determined by thermal fluctuations. The following buffers transmit the random bits from the first buffer to other circuits. An isolation inductor \( L_{\text{iso}} \) is placed between each adjacent pair of buffers to mitigate the back action from the following circuits to the first buffer. An XOR gate\textsuperscript{39} comprises two splitters, two AND gates, and an OR gate, and thus requires three excitation phases.

We conducted several simulations of the AQFP-based XSR circuit shown in Fig. 3a using a Josephson circuit simulator, JSIM,\textsuperscript{31-40} with the device parameters based on the AIST 10 kA/cm\textsuperscript{2} Nb high-speed standard process (HSTP).\textsuperscript{39} Figure 4 shows the simulation waveforms of the AQFP-based XSR circuit for a clock frequency \( f = 5 \) GHz, where \( I_A \) and \( I_B \) are the signal currents representing the outputs of TRNGs A and B (\( A \) and \( B \) in Fig. 3a), respectively, and \( I_{X1} \) through \( I_{X4} \) are the signal currents representing the outputs of XOR 1 through XOR 4 (\( X \) through \( X \) in Fig. 3a), respectively. This figure clearly shows that four random bitstreams are generated from two TRNGs in synchronization with \( t_1 \) and \( t_2 \). Here we take a look at one of the outputs to see if the outputs are generated as expected. The circled 1 in \( I_{X1} \) is generated by XORing the circled 0 in \( I_A \) and the circled 1 in \( I_B \); \( I_{X1} \) lags behind \( I_A \) and \( I_B \) by five and two clock cycles (twenty and eight phases), respectively, which agrees with the excitation phases shown in Fig. 3a. The autocorrelation of each output bitstream and the correlation between the output bitstreams will be experimentally evaluated in the next section.

Here we estimate the Josephson junction count and power dissipation of the AQFP-based XSR as functions of \( n \). As shown in Fig. 3a, a shift register includes five buffers per bit, except for the last bit slice including four buffers. Thus, shift registers A and B include approximately 10\( n \) buffers, which results in 20\( n \) Josephson junctions. An XOR gate includes 22 junctions\textsuperscript{39}, so that the XOR gate array includes 22\( n \) junctions. Therefore, the XSR circuit includes approximately 42\( n \) Josephson junctions, where the junctions in TRNGs A and B and those in the buffers in the last excitation stage are ignored. Based on the previous study\textsuperscript{22}, the energy dissipation of an AQFP circuit is roughly given by \( 1.4 \times 10^{-13} \) J per junction at 5 GHz operation. Hence, the power dissipation of the XSR circuit is approximately 290 nW at 5 GHz operation.

**Experiments**

As a proof of concept, we fabricated an AQFP-based XSR circuit for \( n = 4 \) and observed its performance. Figure 5 shows a micrograph of the XSR circuit fabricated with the HSTP, based on the schematic diagram in Fig. 3a. The paired excitation currents \( (I_A \) and \( I_B \) were externally provided by a function generator. The outputs of the circuit (\( X \) through \( X \) in Fig. 3a) were converted into the corresponding voltage signals (\( V_{X1} \) through \( V_{X4} \)) by dc superconducting interference quantum devices for read-out. The outputs of TRNGs A and B (\( A \) and \( B \) in Fig. 3a) were bypassed and also converted into voltage signals (\( V_A \) and \( V_B \)). Input currents \( I_{inA} \) and \( I_{inB} \) were used to adjust the probability distribution of TRNGs A and B, respectively. Ideally, an AQFP TRNG generates 0 s and 1 s with the same probability. However, the probability distribution can vary due to circuit parameter variation; thus, \( I_{inA} \) and \( I_{inB} \) were applied so that 0 s and 1 s would appear with the same probability\textsuperscript{27}. This suggests the importance of XSR; if \( n \) random bitstreams are generated from \( n \) AQFP TRNGs without using XSR, \( n \) input lines are needed to adjust the probability distribution of the TRNGs, which can deteriorate the scalability of the entire system.

We implemented the XSR circuit chip in the dipping probe and tested the chip in liquid He at 4.2 K and a low clock frequency \( (f = 100 \) kHz) due to the narrow bandwidth of the probe. First, we observed \( V_A \) and \( V_B \) and set \( I_{inA} \) and \( I_{inB} \) to 8.4 \( \mu \)A and 2.8 \( \mu \)A, respectively, so that 0 s and 1 s appear with the same probability. We then observed \( V_{X1} \) through \( V_{X4} \) to evaluate correlation. Figure 6 provides an example of the measurement waveforms of the XSR circuit, which clearly shows that four random bitstreams (\( V_{X1} \) through \( V_{X4} \)) are generated in synchronization with \( I_A \) and \( I_B \). The measured operating margins of \( I_A \) and \( I_B \) were \( \pm 25 \) % and \( \pm 24 \) %, respectively.

**Autocorrelation of each bitstream.** To evaluate autocorrelation, we measured the autocorrelation function \( R(I) \) of each bitstream (\( X \)) using the following equation:

![Figure 4](https://example.com/figure4.png)

**Figure 4.** Simulation waveforms of the AQFP-based XSR circuit for \( f = 5 \) GHz. Four random bitstreams (\( I_{X1} \) through \( I_{X4} \)) are generated from two random bitstreams (\( I_A \) and \( I_B \)).
where $i \in \{1, 2, 3, 4\}$, $N$ is the number of bits in the bitstream, $Y_i(t)$ is the normalized $X_i$ [$Y_i(t) = -1$ represents a logic 0 and $Y_i(t) = 1$ represents a logic 1] at time $t$, in clock cycles, and $l \in N$ is the time lag. Figure 7 shows the measured $R_i(l)$ for $N = 2^{17}$ and $l$ ranging from 1 to $10^4$. All the $|R_i(l)|$ values are much smaller than 1 for the entire range of $l$. Furthermore, for each $R_i(l)$, the average $\mu$ is close to zero and the standard deviation $\sigma$ is approximately $2.7 \times 10^{-3}$ or $2.8 \times 10^{-3}$. This confirms that each bitstream exhibits no autocorrelation since, if $X_i$ is without autocorrelation, $Y_i(t) \cdot Y_i(t + l)$ becomes $-1$ or 1 randomly and results in $\mu = 0$ and $\sigma = (N)^{-0.5} = 2.76 \times 10^{-3}$ for $N = 2^{17}$, which agrees with our measurement results. In sum, the above results show that an AQFP-based XSR circuit can generate multiple random bitstreams without autocorrelation from two TRNGs.

**Correlation between bitstreams.** To evaluate the correlation between the bitstreams generated by the XSR circuit, we measured the cross-correlation function $R_{ij}(l)$ for each pair of the bitstreams ($X_i$ and $X_j$) using the following equation:

$$R_{ij}(l) = \frac{1}{N} \sum_{t=0}^{N-1} Y_i(t) \cdot Y_j(t + l)$$

Figure 5. Micrograph of the AQFP-based XSR circuit for $n = 4$, which corresponds to the circuit shown in Fig. 3a. The circuit was fabricated using the HSTP.

Figure 6. Measurement waveforms of the AQFP-based XSR circuit for $f = 100$ kHz. The outputs are converted into the corresponding voltage signals ($V_X$ through $V_{X_4}$).

![Micrograph of the AQFP-based XSR circuit for n = 4](image-url)
where $i \in \{1, 2, 3, 4\}, j \in \{1, 2, 3, 4\},$ and $i \neq j$. Figure 8 shows the measured $R_{ij}(l)$ for $N=2^{17}$ and $l$ ranging from $-10^4$ to $10^4$. All the $|R_{ij}(l)|$ values are much smaller than 1 for the entire range of $l$. Furthermore, $\mu$ is close to zero and $\sigma$ is approximately $2.7 \times 10^{-3}$ for each $R_{ij}(l)$, which demonstrates that there is no correlation between the bitstreams for the same reason as that given in the preceding discussion of autocorrelation (i.e., $\mu = 0$ and $\sigma = 2.76 \times 10^{-3}$ if there is no correlation between the bitstreams). Taken together, these results show that an AQFP-based XSR circuit can generate multiple uncorrelated random bitstreams from two TRNGs.

**Conclusions**

We proposed a scalable true-random-number generating scheme, XSR, suitable for alternative computing. XSR generates multiple true random bitstreams in parallel using only two TRNGs, two shift registers, and a number of XOR gates. We discussed XSR from logical and thermodynamic viewpoints and showed that the XOR gates in XSR increase the number of random bits by producing mutual information. Since XSR utilizes simple logic gates, it can be implemented by a variety of logic devices. As a proof of concept, we designed and demonstrated an AQFP-based XSR circuit generating four random bitstreams. The measurement results for autocorrelation and cross-correlation showed that the XSR circuit can generate multiple uncorrelated random bitstreams. Our next step is to develop energy-efficient alternative-computing systems using AQFP-based XSR circuits.
Figure 8. Measured cross-correlation functions between the bitstreams for \(N = 2^{17}\). For each cross-correlation function, the average is close to zero and the standard deviation is approximately \(2.7 \times 10^{-3}\), which indicates that there is no correlation between the bitstreams generated by XSR.

Data availability

All data generated or analyzed during this study are included in this article.

Received: 6 September 2022; Accepted: 11 November 2022
Published online: 21 November 2022

References

1. Patterson, D. 50 Years of computer architecture: From the mainframe CPU to the domain-specific TPU and the open RISC-V instruction set. in 2018 IEEE International Solid-State Circuits Conference (ISSCC) vol. 61 27–31 (IEEE, 2018).
2. Gaines, B. R. Stochastic computing systems. in Advances in Information Systems Science 37–172 (Springer US, 1969). doi:https://doi.org/10.1007/978-1-4899-5841-9_2.
3. Brown, B. D. & Card, H. C. Stochastic neural computation I. Computational elements. IEEE Trans. Comput. 50, 891–905 (2001).
4. Alaghi, A., Qian, W. & Hayes, J. P. The promise and challenge of stochastic computing. IEEE Trans. Comput. Des. Integr. Circ. Syst. 37, 1515–1531 (2018).
5. Kirkpatrick, S., Gelatt, C. D. & Vecchi, M. P. Optimization by simulated annealing. Science 220, 671–680 (1983).
6. Takemoto, T., Hayashi, M., Yoshimura, C. & Yamaoka, M. A 2 × 30k-spin multi-chip scalable CMOS annealing processor based on a processing-in-memory approach for solving large-scale combinatorial optimization problems. IEEE J. Solid-State Circ. 55, 145–156 (2020).
7. Matsubara, S. et al. Digital annealer for high-speed solving of combinatorial optimization problems and its applications. in 2020 25th Asia and South Pacific Design Automation Conference (ASP-DAC) vols 2020-Janua 667–672 (IEEE, 2020).
8. Suri, M. et al. Bio-inspired stochastic computing using binary CBRAM synapses. IEEE Trans. Electron Dev. 60, 2402–2409 (2013).
9. Hsiao, H., Anderson, J. & Hara-Azumi, Y. Generating stochastic bitstreams. in Stochastic Computing: Techniques and Applications 137–152 (Springer International Publishing, 2019). doi:https://doi.org/10.1007/978-3-030-03730-7_7.
10. Marsaglia, G. Xorshift RNGs. J. Stat. Softw. 8, 1–6 (2003).
16. Anderson, H. J., Haru-Azumi, Y. & Yamashita, S. Effect of LFSR seeding, scrambling and feedback polynomial on stochastic computing accuracy. In Proceedings of the 2016 Design, Automation & Test in Europe Conference & Exhibition (DATE) 1550–1555 (Research Publishing Services, 2016). doi:https://10.3850/9783981537079_0162.

17. Takeuchi, N., Ozawa, D., Yamanashi, Y. & Yoshikawa, N. An adiabatic quantum flux parametron as an ultra-low-power logic device. Supercond. Sci. Technol. 26, 035010 (2013).

18. Takeuchi, N., Yamae, T., Ayala, C. L., Suzuki, H. & Yoshikawa, N. Adiabatic quantum-flux-parametron: A tutorial review. IEICE Trans. Electron. E105.C, 251–263 (2022).

19. Lee, K. & Goto, E. Analysis of flux input and output Josephson pair device. IEEE Trans. Magn. 21, 884–887 (1985).

20. Hosoya, M. et al. Quantum flux parametron: A single quantum flux device for Josephson supercomputer. IEEE Trans. Appl. Supercond. 1, 77–89 (1991).

21. Takeuchi, N., Yamae, T., Ayala, C. L., Suzuki, H. & Yoshikawa, N. An adiabatic superconductor 8-bit adder with 24k\textsubscript{T} energy dissipation per junction. Appl. Phys. Lett. 114, 042602 (2019).

22. Keyes, R. W. & Landauer, R. Minimal energy dissipation in logic. IBM J. Res. Dev. 14, 152–157 (1970).

23. Likharev, K. Dynamics of some single quantum devices: I. Parametric quanton. IEEE Trans. Magn. 13, 242–244 (1977).

24. Koller, J. G. & Athas, W. C. Adiabatic switching, low energy computing, and the physics of storing and erasing information. In Workshop on Physics and Computation 267–270 (IEEE, 1992). doi:https://10.1109/PHYCMP.1992.615554.

25. Takeuchi, N. & Yoshikawa, N. Minimum energy dissipation required for a logically irreversible operation. IBM J. Res. Dev. 97, 012124 (2018).

26. Takeuchi, N., Aono, M. & Yoshikawa, N. Superconductor amoeba-inspired problem solvers for combinatorial optimization. Phys. Rev. Appl. 11, 044069 (2019).

27. Luo, W., Takeuchi, N., Chen, O. & Yoshikawa, N. Low-autocorrelation random number generator based on adiabatic quantum-flux-parametron logic. IEEE Trans. Appl. Supercond. 31, 1302305 (2021).

28. Sunar, B., Martin, W. & Stinson, D. A provably secure true random number generator with built-in tolerance to active attacks. IEEE Trans. Comput. 56, 109–119 (2007).

29. Yamanashi, Y. & Yoshikawa, N. Superconductive random number generator using thermal noises in SFQ circuits. IEEE Trans. Appl. Supercond. 19, 630–633 (2009).

30. Hata, H. & Ikikawa, S. FPGA implementation of metastability-based true random number generator. IEICE Trans. Inf. Syst. E95.D, 426–436 (2012).

31. Vodenicarevic, D. et al. Low-energy truly random number generation with superparamagnetic tunnel junctions for unconventional computing. Phys. Rev. Appl. 8, 054045 (2017).

32. Li, H. et al. True random number generator realized by extracting entropy from a negative-inductance superconducting quantum interference device. Chin. Phys. Lett. 34, 014011 (2017).

33. Onomi, T. & Mizugaki, Y. Hardware random number generator using Josephson oscillation and SFQ logic circuits. IEEE Trans. Appl. Supercond. 30, 1301305 (2020).

34. Davies, R. B. Exclusive OR (XOR) and hardware random number generators. (2002). Accessed: Nov. 10, 2021. [Online]. Available: http://www.robertnz.net/pdf/xor2.pdf.

35. Sagawa, T. Thermodynamic and logical reversibilities revisited. J. Stat. Mech. Theory Exp. 2014, P03025 (2014).

36. Landauer, R. Irreversibility and heat generation in the computing process. IBM J. Res. Dev. 5, 183–191 (1961).

37. Hayashi, M. et al. Accelerator chip for ground-state searches of Ising model with asynchronous random pulse distribution. Int. J. Netw. Comput. 6, 195–211 (2016).

38. Takeuchi, N. et al. Adiabatic quantum-flux-parametron cell library designed using a 10 kA cm\textsuperscript{-2} niobium fabrication process. Supercond. Sci. Technol. 30, 035002 (2017).

39. Takeuchi, N., Nozoe, M., He, Y. & Yoshikawa, N. Low-latency adiabatic superconductor logic using delay-line clocking. Appl. Phys. Lett. 115, 072601 (2019).

40. Aono, T. et al. Three-dimensional adiabatic quantum-flux-parametron fabricated using a double-active-layered niobium process. Supercond. Sci. Technol. 30, 075003 (2017).

41. Fang, E. & Van Duzer, T. A Josephson integrated circuit simulator (JSIM) for superconductive electronics application. in 1989 International Superconductivity Electronics Conference (ISEC ’89) 407–410 (1989).

42. Jeffery, M., Xie, P. Y., Whiteley, S. R. & Van Duzer, T. Monte Carlo and thermal noise analysis of ultra-high-speed high temperature superconductor digital circuits. IEEE Trans. Appl. Supercond. 9, 4095–4098 (1999).

Acknowledgements
This study was supported by JSPS KAKENHI (Grants No JP22H00220 and No. JP19H05614) and JST PRESTO (Grant No. JPMJPR19M7). The circuits were fabricated in the Clean Room for Analog-digital superconductivity (GRAVITY) at the National Institute of Advanced Industrial Science and Technology (AIST). We would like to thank C. J. Fourie for providing a 3D inductance extractor, InducEx.

Author contributions
W.L. designed the circuits and performed experiments. O.C. and N.Y. supported theoretical aspects. N.T. proposed the idea and wrote the manuscript. All authors discussed the results.

Competing interests
The authors declare no competing interests.

Additional information
Correspondence and requests for materials should be addressed to N.T.

Reprints and permissions information is available at www.nature.com/reprints.

Publisher’s note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.
