Performance Analysis of a Repetition Cat Code Architecture: Computing 256-bit Elliptic Curve Logarithm in 9 Hours with 126 133 Cat Qubits

Élie Gouzien,[1,*] Diego Ruiz,[2,3] Francois-Marie Le Régent,[2,3] Jérémie Guillaud,[2] and Nicolas Sangouard[1,†]
[1]*Université Paris–Saclay, CNRS, CEA, Institut de physique théorique, 91191 Gif-sur-Yvette, France*
[2]*Alice & Bob, 53 boulevard du Général Martial Valin, 75015 Paris, France*
[3]*Laboratoire de Physique de l'École normale supérieure, École normale supérieure, Mines Paris, Université PSL, Sorbonne Université, CNRS, Inria, 75005 Paris, France*

(Received 26 January 2023; accepted 5 June 2023; published 24 July 2023)

Cat qubits provide appealing building blocks for quantum computing. They exhibit a tunable noise bias yielding an exponential suppression of bit flips with the average photon number and a protection against the remaining phase errors can be ensured by a simple repetition code. We here quantify the cost of a repetition code and provide valuable guidance for the choice of a large scale architecture using cat qubits by realizing a performance analysis based on the computation of discrete logarithms on an elliptic curve with Shor's algorithm. By focusing on a 2D grid of cat qubits with neighboring connectivity, we propose to implement 2-qubit gates via lattice surgery and Toffoli gates with off-line fault-tolerant preparation of magic states through projective measurements and subsequent gate teleportations. All-to-all connectivity between logical qubits is ensured by routing qubits. Assuming a ratio between single- and two-photon losses of 10^{-5} and a cycle time of 500 ns, we show concretely that such an architecture can compute a 256-bit elliptic curve logarithm in 9 h with 126 133 cat qubits and on average 19 photons by cat state. We give the details of the realization of Shor's algorithm so that the proposed performance analysis can be easily reused to guide the choice of architecture for others platforms.

DOI: 10.1103/PhysRevLett.131.040602

Introduction.—While quantum computing can offer substantial speedups for solving specific problems [1], billions of operations are typically required for implementing large scale algorithms [2–4]. This means that the convergence of quantum algorithms cannot realistically be ensured by requiring physical errors to occur with a probability smaller than the inverse of the number of required operations. Instead, the concept of fault-tolerant quantum computation [5] is envisioned. It relies on the idea that, if the rate of physical errors is below a certain threshold, quantum error correction schemes suppress the logical error rate to arbitrary low levels and make possible—at least in principle—arbitrary long sequences of operations [6–8].

With its relatively high thresholds, the surface code is one of the most popular quantum error correction codes [9,10]. As a 2D code, the number of physical qubits per logical qubit increases quadratically with the code distance. Their actual implementation hence comes at the price of a significant overhead in physical resources, with typically hundreds or even thousands of physical qubits per logical qubits to achieve the level of protection required for performing billions of noise-free operations [5].

Some physical platforms naturally exhibit a noise bias [11] that can be exploited to increase code thresholds and hence to reduce the overhead [12,13]. Bosonic systems stabilized in a two-dimensional manifold spanned by cat states—superpositions of coherent states with opposite phases—with an engineered dissipation scheme combining two-photon drive and two-photon dissipation stand out in this framework. The noise bias is indeed tunable in this case, with bit flips that are suppressed exponentially with the mean photon number [14–16]. The remaining phase errors can be corrected with a simple repetition code—a 1D code with a number of physical qubits per logical qubit increasing linearly with the code distance. Given that gate sets at the physical level preserving the noise asymmetry have been described and their use for the implementation of various universal sets at the logical level has been identified [17], cat qubits are becoming an option for realizing a large scale quantum computer. The gain of having a 1D over a 2D code and the details of the implementation of a large scale algorithm with cat qubits are, however, missing.

We here propose a generic tool for analyzing the performance of quantum computing architectures using Shor's algorithm [18,19] for computing discrete logarithms on elliptic curves over prime fields—a hard classical problem at the core of cryptosystems widely used for key exchange and digital signatures [20,21]. The security level of these cryptosystems against classical attacks relies on precise knowledge of the performance of classical algorithms—knowledge that is useful to witness a quantum advantage. The best currently known classical algorithms to compute elliptic curve discrete logarithms are exponential

in the size of the input parameters, whereas there exist subexponential algorithms for factoring. This facilitates the achievements of a quantum advantage with respect to algorithms with subexponential speedups, including Shor's algorithm for the factorization.

We propose a concrete layout in which cat qubits are placed at the nodes of a 2D grid with physical connections to their neighboring qubits only. All-to-all connectivity between logical qubits is ensured by means of routing qubits. 2-qubit gates are implemented by means of lattice surgery and Toffoli gates are obtained by gate teleportation through an off-line fault-tolerant magic state preparation based on projective measurements. From detailed models of physical qubits and their manipulations, we estimate precisely the errors related to the implementation of logical operations by considering a ratio between single- and two-photon losses of 10^{-5}. We show concretely that such an architecture can compute a discrete logarithm on the secp256k1 curve, which is used for securing signatures in Bitcoin transactions [22] in 9 h with 126 133 cat qubits and with 19 photons per cat state on average. Note that keeping the exponential reduction of bit flip for cat sizes up to 19 photons on average might be experimentally challenging [23,24], and our results suggest that either substantial improvements in the design of cat qubits are required, or a thin rectangular surface code [25] is more suitable for such a large computation (see the Supplemental Material [26], Secs. D1 and F for more details). Independent of the feasibility question, the gain in using a 1D instead of a 2D code is quantified in detail.

Elliptic curve and discrete logarithm.—An elliptic curve is defined as the set of points associated with the coordinates (x, y) satisfying the equation $y^2 = x^3 + ax + b$ with fixed values for a and b. We are interested in cryptographic relevant elliptic curves for which x, y, a, and b belong to the field of integers modulo p, with p a prime number (n bits long). We define a binary operation on these elliptic curves, called "addition" and denoted "+": for two points P and Q, $R = P + Q$ have, in the generic case [85], coordinates given by

$$x_R = \lambda^2 - x_P - x_Q, \tag{1a}$$

$$y_R = -y_P - \lambda(x_R - x_P), \tag{1b}$$

where $\lambda = (y_Q - y_P)/(x_Q - x_P)$ is the slope of the line joining P and Q. A multiplication by an integer k naturally arises as $kP = \underbrace{P + P + \cdots + P}_{k \text{ times}}$. A cyclic subgroup can be formed from the multiples of a point G (the subgroup generator) on the curve. The security of cryptographic algorithms based on elliptic curves, such as the elliptic curve digital signature algorithm [21], relies on the hardness to find a number k (the logarithm) from the knowledge of the generator G and the point $P = kG$ (see the Supplemental Material [26], Sec. A for details on elliptic curve cryptography).

Shor's algorithm.—Shor introduced an algorithm [18,86] to compute discrete logarithms on a quantum computer with a number of gates cubic in n. It takes three steps and three registers. In the first step, two registers encoding x_1 and x_2 are each prepared in a superposition of all possible integers. In the second step, $f(x_1, x_2) = x_1 G - x_2 P$ is computed and stored in the third register. In the last step, a quantum Fourier transform of the two registers containing x_1 and x_2 (which corresponds to a 2D quantum Fourier transform) reveals the value of k (see the Supplemental Material [26], Sec. B for more details and a discussion on Ekerå's version of Shor's algorithm [27]).

The preparation of registers in a superposition of all integers has a linear cost. It is indeed obtained through the preparation of qubits in state $|+\rangle = (|0\rangle + |1\rangle)/\sqrt{2}$, that is, by applying a Hadamard transformation on each qubit. Since the quantum Fourier transform precedes a measurement of each qubit, it can be implemented in a semiclassical way [87], with a linear cost as well. The cost of Shor's algorithm is largely dominated by the computation of f, which we evaluate in detail below.

Arithmetic circuits.—f is the difference between the results of two scalar multiplications. The elliptic curve scalar multiplication is implemented with a windowed approach, as in [28]. The principle is to decompose the factor k into groups of bits and to rewrite the multiplication as a sequence of elliptic curve point additions,

$$kG = \sum_{\substack{i=0 \\ i \equiv 0 \bmod w_e}}^{n_e} 2^i k_{i:i+w_e} G, \tag{2}$$

with n_e the number of bits in k, w_e the width of each window, and $k_{i:i+w_e}$ the number formed from w_e bits of k starting at bit i. For each term, the point $2^i k_{i:i+w_e} G$ is computed classically for all possible values of $k_{i:i+w_e}$ and loaded into a quantum register through a quantum table lookup circuit [29,30], with the qubits encoding $k_{i:i+w_e}$ as controls.

Each point addition is realized from Eq. (1) using a quantum implementation of each operation [88]. This takes arithmetic additions, subtractions, multiplications, and divisions, modulo the prime number p; see the Supplemental Material [26], Secs. C7 and C8 for details.

A ripple-carry circuit from [31] is used to perform the additions. The basic idea is to start with the low-order bits of the inputs, compute the first carry with a Toffoli gate, take the value of the carry and the next bits of the inputs to compute the second carry, and so on up to the high-order bits. We then work from the high-order bits back down to the low-order bits by computing the result of the addition bit by bit, store the value in the first input register, and restore the value of the second input register to get a

reversible computation. Note that the subtraction is obtained by conjugation of the addition. To make an addition modulo p, the standard addition is followed by a comparison of the result with p, which is obtained by subtracting p from the result of the addition and by checking the most significant bit of the result of this subtraction. A subtraction of p is then realized, conditioned on the result of the comparison. In order to save resources, the comparison and subtraction, which start identically, are merged together to form a modular reduction; see the Supplemental Material [26], Sec. C2 for the details on the circuits. Note that the modular subtraction is obtained by conjugation of the modular addition.

The multiplication can be implemented with a standard double-and-add method, which can be illustrated by considering the product of two (n bits) numbers x_1 and x_2. From the binary representation of $x_1 = \sum_i 2^i [x_1]_i$, we have $x_1 x_2 = \sum_i 2^i [x_1]_i x_2 = [x_1]_0 x_2 + 2\{[x_1]_1 x_2 + 2([x_1]_2 y + \cdots)\}$; i.e., the result of the product is obtained by first considering the last term (x_2 conditioned on the value of $[x_1]_{n-1}$), doubling the result, and adding x_2 conditioned on $[x_1]_{n-2}$, and so on up to the first term. The multiplication modulo p is then naturally obtained by performing additions and doublings modulo p, which takes $2n$ modular reductions. We used a representation (compatible with the addition), the Montgomery representation, to reduce the number of reductions [28,32]. It simply consists of representing a number x_1 by $y_1 = x_1 2^n \bmod p$. Considering the numbers x_1 and x_2, and their respective Montgomery representation y_1, y_2, the product $x_1 x_2$ is represented by $x_1 x_2 2^n \bmod p$, which is obtained by computing $y_1, y_2 \mapsto y_1 y_2 2^{-n} \bmod p$ from a double-and-add multiplier in which the doubling operation is replaced by halving. In this case, the sums need a single modular reduction to realize a multiplication modulo p, hence reducing the number of modular reductions to $n+1$ [89]. Note that the latter is further reduced by using a windowed version of the multiplication in the Montgomery representation; see the Supplemental Material [26], Sec. C4 for the details of the multiplication circuit.

The modular division between two numbers x_1 and x_2 is obtained by a modular multiplication of x_1 and the modular inverse of x_2. The modular inversion is performed with Kaliski's algorithm [33]. This algorithm is essentially a binary version of the extended Euclidean algorithm. To make it compatible with the Montgomery representation, the result is multiplied by 2^{2n} such that starting from the representation $y_2 = x_2 2^n \bmod p$, Kaliski's algorithm returns $x_2^{-1} 2^n \bmod p = y_2^{-1} 2^{2n} \bmod p$. The circuit we use is inspired by [28], with improvements, most notably by using subcircuits crafted for use of Toffoli gates; see the Supplemental Material [26], Sec. C5.

Given the number of point additions in the scalar multiplication at the core of the discrete logarithm computation, the decomposition of a point addition in elementary arithmetic operations and the number of gates that is required for implementing each of these elementary operations, we deduce that the implementation of Shor's algorithm takes $448n^3/w_e$ controlled NOT (CNOT) and $348n^3/w_e$ Toffoli gates at the leading order, with w_e the size of each window for the elliptic curve multiplication; see the Supplemental Material [26], Sec. C10.

Cat qubits with repetition code.—We are interested in cat qubits, in which information is encoded in two coherent states of a harmonic oscillator with the same amplitude and opposite phase $|\alpha\rangle$ and $|-\alpha\rangle$ [14,34]. Here α is assumed real, without loss of generality. To avoid that the state of the oscillator leaves the computation subspace $(|\alpha\rangle, |-\alpha\rangle)$ in the presence of loss and noise, a stabilization mechanism is needed. We consider a mechanism combining a two-photon drive and an engineered two-photon dissipation, which can be implemented appropriately in a physical realization using cavity modes coupled nonlinearly by Josephson junctions. When the corresponding stabilization rate is higher than that of typical errors, the bit-flip error rate induced by single-photon loss, thermal excitations or dephasing are exponentially suppressed with the mean number of photons in the cat size $\gamma_X \propto \exp(-2\alpha^2)$, while the phase-flip error rate typically scales linearly $\gamma_Z \propto \alpha^2$ [23,24]. In this Letter, the amplitude α is a free parameter. Its value is chosen such that bit flips happen with a low probability during the run-time of Shor's algorithm and a repetition code corrects phase-flip errors only [17,25]. Details on cat qubits and their implementation are given in the Supplemental Material [26], Sec. D.

As bit flips are not corrected, it is crucial not to introduce such errors during the algorithm execution. At the physical level, this is obtained by using bias-preserving operations, including the preparations $\mathcal{P}_{|0/1\rangle}$ and $\mathcal{P}_{|\pm\rangle}$ of the computational states $|\pm\alpha\rangle$ and cat states $|\mathcal{C}_\alpha^\pm\rangle = \left[1/\sqrt{2(1 \pm e^{-2\alpha^2})}\right](|\alpha\rangle \pm |-\alpha\rangle)$, respectively, the measurements $\mathcal{M}_Z$ and $\mathcal{M}_X$, the Z and X gates, and CNOT and Toffoli gates; see the details in the Supplemental Material [26], Sec. D.

The principle of a distance-d repetition code is to introduce redundancy in the information encoding $|\pm\rangle_L := |\pm\rangle^{\otimes d}$ and make use of $d-1$ stabilizer measurements $S_i = X_i X_{i+1}$ to identify and correct phase-flip errors after each operation. In our case, $|\pm\rangle^{\otimes d} = |\mathcal{C}_\alpha^\pm\rangle^{\otimes d}$ and S_i is measured from the bias-preserving operations $\mathcal{P}_{|+\rangle}$, CNOT, $\mathcal{M}_X$.

We consider the logical operations in the set $\mathcal{S}_L = \{\mathcal{P}_{|+\rangle_L}, \mathcal{P}_{|0\rangle_L}, \mathcal{M}_{Z_L}, \mathcal{M}_{X_L}, Z_L, X_L, \mathrm{CX}_L^k, \mathrm{CCX}_L\}$, where $\mathrm{CX}^k{}_L$ designates the multi-target CNOT gate and CCX_L the Toffoli gate, which can all be implemented transversally on the repetition code, except for the CCX_L gate; see the Supplemental Material [26], Sec. E. The transversal implementation of the CNOT_L gate, however, requires all-to-all couplings between the physical qubits of the processor, which is not a realistic feature of a superconducting quantum processor. Instead, we focus on a realization

based on lattice surgery using nearest-neighbor interactions only (with additional routing qubits included in the resource evaluation), as detailed in the Supplemental Material [26], Sec. E. The same idea can be extended to the multiple target CX_L^k gate, which applies an X gate on k qubits if the control qubit is in state $|1\rangle_L$ and the identity otherwise; see also the Supplemental Material [26], Sec. E. Finally, the logical Toffoli gate CCX_L is implemented using gate teleportation [90] from a "Toffoli magic state" $|CCX\rangle = \frac{1}{2}(|000\rangle + |010\rangle + |100\rangle + |111\rangle)$, as detailed in the Supplemental Material [26], Sec. E4 (with the circuit depicted in Fig. 32). The fault-tolerant preparation of the Toffoli magic state at the logical level, based on a projective measurement, is discussed in the Supplemental Material [26], Sec. E4.

Noise model.—We exclusively consider the single-photon loss at rate κ_1, as in the presence of two-photon dissipation, the other error mechanisms have little impact on the noise model [17]. Our resource estimates are based on the assumptions that a two-photon dissipation rate of $\kappa_2/2\pi = 1.59$ MHz and a resonator lifetime of $T_1 = 10$ ms can be achieved, which corresponds to a ratio $\kappa_1/\kappa_2 = 10^{-5}$ and a repetition code cycle time of $T_{\text{cycle}} = 5/\kappa_2 = 500$ ns.

For a fixed gate time of $1/\kappa_2$ (assumed to be identical for state preparation, measurement, and CNOT gates), the logical error rate per cycle of a distance-d repetition code is given by [35] (see the Supplemental Material [26], Sec. E2 for details),

$$\epsilon_L = 5.6\times 10^{-2}\left(\frac{(\alpha^2)^{0.86}\kappa_1/\kappa_2}{(\kappa_1/\kappa_2)_{\text{th}}}\right)^{\frac{d+1}{2}} + 2(d-1)\times 0.50 e^{-2\alpha^2}, \tag{3}$$

where the first term is the logical phase-flip error rate and the second term is the logical bit-flip error rate, and $(\kappa_1/\kappa_2)_{\text{th}} = 1.3\times 10^{-2}$.

ϵ_L corresponds to the error rate of all gates (including the identity gate), but the Toffoli gate. For the latter, we consider two variations of the state preparations of Toffoli magic states [25] using either error detection or error correction. The resource evaluation uses the most suitable implementation, which depends on the size of the elliptic curve logarithms to compute.

Methods and results.—As several parameters are involved, we run an exhaustive search to minimize the product of the average photon number, expected time to solution, and total number of physical qubits (the optimization code can be found in [36]). The required resources are shown in Fig. 1 as a function of the number of bits of the prime p; see the Supplemental Material [26], Sec. F for the details on the optimal parameters. We see that 126 133 qubits are needed to compute a 256-bit logarithm in 9 h, for example, with on average 19 photons as the size of each cat qubit. The required cat size suggests significant improvement

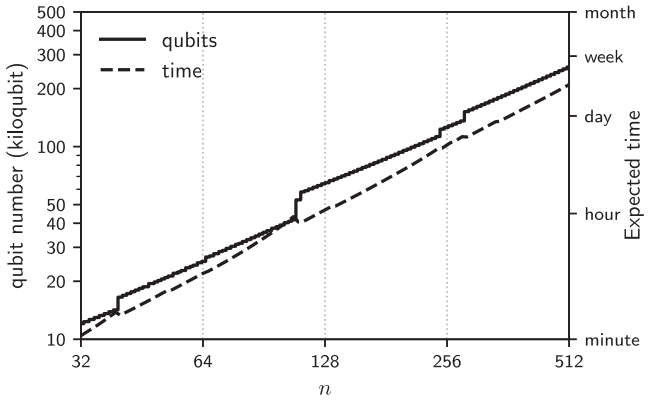

FIG. 1. Number of physical cat qubits and run-time for computing a discrete logarithm on an elliptic curve as a function of the number of bits in p.

in the design of cat qubits or, alternatively, the use of thin rectangular 2D codes [25] (see the Supplemental Material [26], Sec. F).

Conclusion.—We reported on a performance analysis and provided a valuable guidance for the choice of a large scale architecture of a platform using cat qubits under the assumption that the bit flips are negligible and the remaining phase errors are corrected by a simple repetition code using Shor's algorithm. We gave the details of an improved quantum computation of a discrete logarithm on elliptic curves, taking as an example the one used for securing signatures in Bitcoin transactions. Assuming a ratio between single- and two-photon losses of 10^{-5} and a cycle time of 500 ns, we have shown that 126 133 qubits are needed to compute a 256-bit logarithm in 9 h and 19 photons on average by cat state. We also estimated that the implementation of Shor's algorithm for the factorization of 2048 Rivest-Shamir-Adleman (RSA) integers would take 349 133 cat qubits and 4 days under the same assumption. This provides a comparative analysis of the security level of two widely used cryptographic schemes. This also favors comparisons with alternative platforms [91] and illustrates the gain in using a 1D code by comparing this cost estimation with the estimate reported in [3] showing that 20×10^6 qubits and 8 h would be needed for realizing the same factorization with a 2D grid of superconducting qubits and a standard surface code. Note that, in both cases, the number of processing qubits can be substantially reduced by adding a quantum memory to the processor [37]. Further note that parallelization has not been exploited in this Letter. This could dramatically reduce the run-time, especially as the preparation of magic states is resource efficient and increasing the number of their factories would not significantly increase the total number of qubits, which would allow adequate use of look-ahead adder [38] (see the Supplemental Material [26], Sec. G for details).

We would like to thank Mazyar Mirrahimi for many discussions about the Toffoli magic state preparation

schemes. We are also grateful to Martin Ekerå for his comments on a first version of this Letter. E. G. and N. S. acknowledge funding by the Institut de Physique Théorique (IPhT), Commissariat à l'Énergie Atomique et aux Energies Alternatives (CEA), the Region Île-de-France in the framework of DIM SIRTEQ, the European Union's Horizon 2020 research and innovation program European High-Performance Computing Joint Undertaking under Grant Agreement No. 101018180 (HPCQS), and a French national quantum initiative managed by Agence Nationale de la Recherche in the framework of France 2030 with the Reference No. ANR-22-PETQ-0009. D. R. and F.-M. L. R. acknowledge funding by Plan France 2030 through the Project No. ANR-22-PETQ-0006.

*elie.gouzien@cea.fr
†https://quantum.paris.

[1] A. Montanaro, Quantum algorithms: An overview, npj Quantum Inf. **2**, 15023 (2016).
[2] M. Reiher, N. Wiebe, K. M. Svore, D. Wecker, and M. Troyer, Elucidating reaction mechanisms on quantum computers, Proc. Natl. Acad. Sci. U.S.A. **114**, 7555 (2017).
[3] C. Gidney and M. Ekerå, How to factor 2048 bit RSA integers in 8 hours using 20 million noisy qubits, Quantum **5**, 433 (2021).
[4] Y. R. Sanders, D. W. Berry, P. C. S. Costa, L. W. Tessler, N. Wiebe, C. Gidney, H. Neven, and R. Babbush, Compilation of fault-tolerant quantum heuristics for combinatorial optimization, PRX Quantum **1**, 020312 (2020).
[5] E. T. Campbell, B. M. Terhal, and C. Vuillot, Roads towards fault-tolerant universal quantum computation, Nature (London) **549**, 172 (2017).
[6] D. Aharonov and M. Ben-Or, Fault-tolerant quantum computation with constant error rate, SIAM J. Comput. **38**, 1207 (2008).
[7] A. Y. Kitaev, Quantum computations: Algorithms and error correction, Russ. Math. Surv. **52**, 1191 (1997).
[8] E. Knill, R. Laflamme, and W. H. Zurek, Resilient quantum computation: Error models and thresholds, Proc. R. Soc. A **454**, 365 (1998).
[9] S. Bravyi and A. Kitaev, Universal quantum computation with ideal Clifford gates and noisy ancillas, Phys. Rev. A **71**, 022316 (2005).
[10] A. G. Fowler, M. Mariantoni, J. M. Martinis, and A. N. Cleland, Surface codes: Towards practical large-scale quantum computation, Phys. Rev. A **86**, 032324 (2012).
[11] L. Childress and R. Hanson, Diamond NV centers for quantum computing and quantum networks, MRS Bull. **38**, 134 (2013).
[12] P. Aliferis and J. Preskill, Fault-tolerant quantum computation against biased noise, Phys. Rev. A **78**, 052331 (2008).
[13] D. K. Tuckett, S. D. Bartlett, and S. T. Flammia, Ultrahigh Error Threshold for Surface Codes with Biased Noise, Phys. Rev. Lett. **120**, 050505 (2018).
[14] M. Mirrahimi, Z. Leghtas, V. V. Albert, S. Touzard, R. J. Schoelkopf, L. Jiang, and M. H. Devoret, Dynamically protected cat-qubits: A new paradigm for universal quantum computation, New J. Phys. **16**, 045014 (2014).
[15] V. V. Albert, C. Shu, S. Krastanov, C. Shen, R.-B. Liu, Z.-B. Yang, R. J. Schoelkopf, M. Mirrahimi, M. H. Devoret, and L. Jiang, Holonomic Quantum Control with Continuous Variable Systems, Phys. Rev. Lett. **116**, 140502 (2016).
[16] S. Puri, S. Boutin, and A. Blais, Engineering the quantum states of light in a Kerr-nonlinear resonator by two-photon driving, npj Quantum Inf. **3**, 18 (2017).
[17] J. Guillaud and M. Mirrahimi, Repetition Cat Qubits for Fault-Tolerant Quantum Computation, Phys. Rev. X **9**, 041053 (2019).
[18] P. W. Shor, Algorithms for quantum computation: Discrete logarithms and factoring, in *Proceedings 35th Annual Symposium on Foundations of Computer Science* (IEEE Computer Society Press, 1994), pp. 124–134.
[19] P. W. Shor, Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer, SIAM J. Comput. **26**, 1484 (1997).
[20] E. Barker, Guideline for using cryptographic standards in the federal government: Cryptographic mechanisms (2020), 10.6028/NIST.SP.800-175Br1.
[21] Information Technology Laboratory, Digital Signature Standard (DSS) (2013), 10.6028/NIST.FIPS.186-4.
[22] D. Aggarwal, G. Brennen, T. Lee, M. Santha, and M. Tomamichel, Quantum attacks on Bitcoin, and how to protect against them, Ledger **3**, (2018).
[23] R. Lescanne, M. Villiers, T. Peronnin, A. Sarlette, M. Delbecq, B. Huard, T. Kontos, M. Mirrahimi, and Z. Leghtas, Exponential suppression of bit-flips in a qubit encoded in an oscillator, Nat. Phys. **16**, 509 (2020).
[24] C. Berdou *et al.*, One hundred second bit-flip time in a two-photon dissipative oscillator, arXiv:2204.09128.
[25] C. Chamberland, K. Noh, P. Arrangoiz-Arriola, E. T. Campbell, C. T. Hann, J. Iverson, H. Putterman, T. C. Bohdanowicz, S. T. Flammia, A. Keller, G. Refael, J. Preskill, L. Jiang, A. H. Safavi-Naeini, O. Painter, and F. G. S. L. Brandão, Building a fault-tolerant quantum computer using concatenated cat codes, PRX Quantum **3**, 010329 (2022).
[26] See Supplemental Material at http://link.aps.org/supplemental/10.1103/PhysRevLett.131.040602 for details about elliptic curves cryptography, Shor's algorithm, arithmetics circuits, cat qubits, repetition code, and the results, which includes Refs. [4,10,12–14,17–19,23–25,27–84].
[27] M. Ekerå, Quantum algorithms for computing general discrete logarithms and orders with tradeoffs, J. Math. Cryptol. **15**, 359 (2021).
[28] T. Häner, S. Jaques, M. Naehrig, M. Roetteler, and M. Soeken, Improved quantum circuits for elliptic curve discrete logarithms, in *Post-Quantum Cryptography*, Lecture Notes in Computer Science Vol. 12100, edited by J. Ding and J.-P. Tillich (Springer International Publishing, New York, 2020), pp. 425–444.
[29] R. Babbush, C. Gidney, D. W. Berry, N. Wiebe, J. McClean, A. Paler, A. Fowler, and H. Neven, Encoding Electronic Spectra in Quantum Circuits with Linear T Complexity, Phys. Rev. X **8**, 041015 (2018).
[30] C. Gidney, Windowed quantum arithmetic, arXiv:1905.07682.
[31] S. A. Cuccaro, T. G. Draper, S. A. Kutin, and D. P. Moulton, A new quantum ripple-carry addition circuit, arXiv:quant-ph/0410184.

[32] M. Roetteler, M. Naehrig, K. M. Svore, and K. Lauter, Quantum resource estimates for computing elliptic curve discrete logarithms, in *Proceedings of the Advances in Cryptology—ASIACRYPT 2017*, Lecture Notes in Computer Science Vol. 10625, edited by T. Takagi and T. Peyrin (Springer International Publishing, 2017), pp. 241–270.

[33] B. S. Kaliski, The Montgomery inverse and its applications, IEEE Trans. Comput. **44**, 1064 (1995).

[34] P. T. Cochrane, G. J. Milburn, and W. J. Munro, Macroscopically distinct quantum-superposition states as a bosonic code for amplitude damping, Phys. Rev. A **59**, 2631 (1999).

[35] F.-M. Le Régent, C. Berdou, Z. Leghtas, J. Guillaud, and M. Mirrahimi, High-performance repetition cat code using fast noisy operations, arXiv:2212.11927.

[36] Code is available at https://github.com/ElieGouzien/elliptic_log_cat, Élie Gouzien, version 1.0 (2023), 10.5281/zenodo.8071018.

[37] É. Gouzien and N. Sangouard, Factoring 2048-bit RSA Integers in 177 Days with 13436 Qubits and a Multimode Memory, Phys. Rev. Lett. **127**, 140503 (2021).

[38] C. Gidney, Quantum block lookahead adders and the wait for magic states, arXiv:2012.01624.

[39] D. R. L. Brown, Standards for Efficient Cryptography (2010).

[40] A. Zieniewicz and J.-L. Pons, Pollard's kangaroo for SECPK1 (2020), https://github.com/JeanLucPons/Kangaroo.

[41] M. Ekerå, Revisiting Shor's quantum algorithm for computing general discrete logarithms, arXiv:1905.09084.

[42] P. L. Montgomery, Modular multiplication without trial division, Math. Comput. **44**, 519 (1985).

[43] R. Rines and I. Chuang, High performance quantum modular multipliers, arXiv:1801.01081.

[44] A. G. Fowler, Time-optimal quantum computation, arXiv:1210.4626.

[45] C. Gidney and A. G. Fowler, Flexible layout of surface code computations using AutoCCZ states, arXiv:1905.08916.

[46] V. Vedral, A. Barenco, and A. Ekert, Quantum networks for elementary arithmetic operations, Phys. Rev. A **54**, 147 (1996).

[47] S. Beauregard, Circuit for Shor's algorithm using $2n+3$ qubits, Quantum Inf. Comput. **3**, 175 (2003).

[48] D. W. Berry, C. Gidney, M. Motta, J. R. McClean, and R. Babbush, Qubitization of arbitrary basis quantum chemistry leveraging sparsity and low rank factorization, Quantum **3**, 208 (2019).

[49] C. H. Bennett, Logical reversibility of computation, IBM J. Res. Dev. **17**, 525 (1973).

[50] A. Barenco, C. H. Bennett, R. Cleve, D. P. DiVincenzo, N. Margolus, P. Shor, T. Sleator, J. A. Smolin, and H. Weinfurter, Elementary gates for quantum computation, Phys. Rev. A **52**, 3457 (1995).

[51] C. Gidney, Constructing Large Controlled Nots (2015), https://algassert.com/circuits/2015/06/05/Constructing-Large-Controlled-Nots.html.

[52] M. A. Nielsen and I. L. Chuang, *Quantum Computation and Quantum Information: 10th Anniversary Edition* (Cambridge University Press, Cambridge, England, 2010).

[53] T. Häner, M. Roetteler, and K. M. Svore, Factoring using $2n+2$ qubits with Toffoli based modular multiplication, Quantum Inf. Comput. **17**, 673 (2017).

[54] A. Joshi, K. Noh, and Y. Y. Gao, Quantum information processing with bosonic qubits in circuit QED, Quantum Sci. Technol. **6**, 033001 (2021).

[55] Z. Leghtas, S. Touzard, I. M. Pop, A. Kou, B. Vlastakis, A. Petrenko, K. M. Sliwa, A. Narla, S. Shankar, M. J. Hatridge, M. J. Reagor, L. Frunzio, R. J. Schoelkopf, M. Mirrahimi, and M. H. Devoret, Confining the state of light to a quantum manifold by engineered two-photon loss, Science **347**, 853 (2015).

[56] S. Touzard, A. Grimm, Z. Leghtas, S. O. Mundhada, P. Reinhold, C. Axline, M. J. Reagor, K. S. Chou, J. Z. Blumoff, K. M. Sliwa, S. Shankar, L. Frunzio, R. J. Schoelkopf, M. Mirrahimi, and M. H. Devoret, Coherent Oscillations inside a Quantum Manifold Stabilized by Dissipation, Phys. Rev. X **8**, 021005 (2018).

[57] D. K. Tuckett, A. S. Darmawan, C. T. Chubb, S. Bravyi, S. D. Bartlett, and S. T. Flammia, Tailoring Surface Codes for Highly Biased Noise, Phys. Rev. X **9**, 041031 (2019).

[58] D. K. Tuckett, S. D. Bartlett, S. T. Flammia, and B. J. Brown, Fault-Tolerant Thresholds for the Surface Code in Excess of 5% under Biased Noise, Phys. Rev. Lett. **124**, 130501 (2020).

[59] Q. Xu, N. Mannucci, A. Seif, A. Kubica, S. T. Flammia, and L. Jiang, Tailored XZZX codes for biased noise, Phys. Rev. Res. **5**, 013035 (2023).

[60] J. P. Bonilla Ataides, D. K. Tuckett, S. D. Bartlett, S. T. Flammia, and B. J. Brown, The XZZX surface code, Nat. Commun. **12**, 2172 (2021).

[61] J. Roffe, L. Z. Cohen, A. O. Quintivalle, D. Chandra, and E. T. Campbell, Bias-tailored quantum LDPC codes, Quantum **7**, 1005 (2023).

[62] A. Wallraff, D. I. Schuster, A. Blais, L. Frunzio, R.-S. Huang, J. Majer, S. Kumar, S. M. Girvin, and R. J. Schoelkopf, Strong coupling of a single photon to a superconducting qubit using circuit quantum electrodynamics, Nature (London) **431**, 162 (2004).

[63] A. Blais, A. L. Grimsmo, S. Girvin, and A. Wallraff, Circuit quantum electrodynamics, Rev. Mod. Phys. **93**, 025005 (2021).

[64] J. Clarke, A. N. Cleland, M. H. Devoret, D. Esteve, and J. M. Martinis, Quantum mechanics of a macroscopic variable: The phase difference of a Josephson junction, Science **239**, 992 (1988).

[65] P. Bertet, A. Auffeves, P. Maioli, S. Osnaghi, T. Meunier, M. Brune, J.-M. Raimond, and S. Haroche, Direct Measurement of the Wigner Function of a One-Photon Fock State in a Cavity, Phys. Rev. Lett. **89**, 200402 (2002).

[66] A. G. Fowler and C. Gidney, Low overhead quantum computation using lattice surgery, arXiv:1808.06709.

[67] D. Litinski, A game of surface codes: Large-scale quantum computing with lattice surgery, Quantum **3**, 128 (2019).

[68] C. Chamberland and E. T. Campbell, Universal quantum computing with twist-free and temporally encoded lattice surgery, PRX Quantum **3**, 010331 (2022).

[69] E. Dennis, A. Kitaev, A. Landahl, and J. Preskill, Topological quantum memory, J. Math. Phys. (N.Y.) **43**, 4452 (2002).

[70] O. Higgott, PyMatching: A PYTHON package for decoding quantum codes with minimum-weight perfect matching, ACM Trans. Quantum Comput. **3**, 1 (2022).

[71] P. Brooks and J. Preskill, Fault-tolerant quantum computation with asymmetric Bacon-Shor codes, Phys. Rev. A **87**, 032310 (2013).
[72] D. Gottesman, Fault-tolerant quantum computation with higher-dimensional systems, Chaos Solitons Fractals **10**, 1749 (1999).
[73] C. Horsman, A. G. Fowler, S. Devitt, and R. Van Meter, Surface code quantum computing by lattice surgery, New J. Phys. **14**, 123011 (2012).
[74] D. Litinski and F. v. Oppen, Lattice surgery with a twist: Simplifying clifford gates of surface codes, Quantum **2**, 62 (2018).
[75] C. Chamberland and E. T. Campbell, Circuit-level protocol and analysis for twist-based lattice surgery, Phys. Rev. Res. **4**, 023090 (2022).
[76] X. Zhou, D. W. Leung, and I. L. Chuang, Methodology for quantum logic gate construction, Phys. Rev. A **62**, 052316 (2000).
[77] J. Guillaud and M. Mirrahimi, Error rates and resource overheads of repetition cat qubits, Phys. Rev. A **103**, 042413 (2021).
[78] P. W. Shor, Fault-tolerant quantum computation, in *Proceedings of 37th Conference on Foundations of Computer Science* (IEEE Computer Society Press, 1996), pp. 56–65.
[79] D. S. Schlegel, F. Minganti, and V. Savona, Quantum error correction using squeezed Schrödinger cat states, Phys. Rev. A **106**, 022431 (2022).
[80] T. Hillmann and F. Quijandría, Quantum error correction with dissipatively stabilized squeezed cat qubits, Phys. Rev. A **107**, 032423 (2023).
[81] Q. Xu, G. Zheng, Y.-X. Wang, P. Zoller, A. A. Clerk, and L. Jiang, Autonomous quantum error correction and fault-tolerant quantum computation with squeezed cat qubits, arXiv:2210.13406.
[82] C. Gidney, Halving the cost of quantum addition, Quantum **2**, 74 (2018).
[83] C. Jones, Low-overhead constructions for the fault-tolerant Toffoli gate, Phys. Rev. A **87**, 022328 (2013).
[84] C. Gidney, Approximate encoded permutations and piecewise quantum adders, arXiv:1905.08488.
[85] The following description of point addition, as well as Eq. (1), are valid when P and Q are distinct and different from the neutral element. If $P = Q$, the tangent of the curve on this point is used. If Q is the neutral element, the line to consider is the one parallel to the y axis going through P, and the result is $P + Q = P$. If $Q = -P$, that is, Q is symmetrical of P with respect to the x axis, the line joining them is vertical and the result is the neutral element: $P + Q = 0$.
[86] D. P. DiVincenzo and P. W. Shor, Fault-Tolerant Error Correction with Efficient Quantum Codes, Phys. Rev. Lett. **77**, 3260 (1996).
[87] R. B. Griffiths and C.-S. Niu, Semiclassical Fourier Transform for Quantum Computation, Phys. Rev. Lett. **76**, 3228 (1996).
[88] Equation (1) only describes the elliptic curve addition in the generic case when P and Q are distinct and different from the neutral element. Only the generic case is implemented, as this saves resources at the cost of an approximate algorithm; the exact consequences are discussed in [32].
[89] We allow the accumulation register to store a number up to $2p$ after each add and half cycle. The division by 2 ensure its value not to be amplified beyond this limit, after each cycle, even when using nonmodular additions (what would not be the case when doubling). See the Supplemental Material [26], Sec. C4 for the details.
[90] D. Gottesman and I. L. Chuang, Demonstrating the viability of universal quantum computation using teleportation and single-qubit operations, Nature (London) **402**, 390 (1999).
[91] D. Litinski and N. Nickerson, Active volume: An architecture for efficient fault-tolerant quantum computers with limited non-local connections, arXiv:2211.15465.