Long-range-enhanced surface codes

Yifan Hong,[1,2,*] Matteo Marinelli,[1,3] Adam M. Kaufman,[1,3] and Andrew Lucas[1,2,†]
[1]*Department of Physics, University of Colorado, Boulder, Colorado 80309, USA*
[2]*Center for Theory of Quantum Matter, University of Colorado, Boulder, Colorado 80309, USA*
[3]*JILA and National Institute of Standards and Technology, Boulder, Colorado 80309, USA*

(Received 2 October 2023; accepted 16 July 2024; published 8 August 2024)

The surface code is a quantum error-correcting code for one logical qubit, protected by spatially localized parity checks in two dimensions. Due to fundamental constraints from spatial locality, storing more logical qubits requires either sacrificing the robustness of the surface code against errors or increasing the number of physical qubits. We bound the minimal number of spatially nonlocal parity checks necessary to add logical qubits to a surface code while maintaining, or improving, robustness to errors. We saturate the lower limit of this bound, when the number of added logical qubits is a constant, using a family of hypergraph product codes, interpolating between the surface code and constant-rate low-density parity-check codes. Fault-tolerant protocols for logical gates in the quantum code can be inherited from its classical parent codes. We provide near-term practical implementations of this code for hardware based on trapped ions or neutral atoms in mobile optical tweezers. Long-range-enhanced surface codes outperform conventional surface codes using hundreds of physical qubits, and they represent a practical strategy to enhance the robustness of logical qubits to errors in near-term devices.

DOI: 10.1103/PhysRevA.110.022607

I. INTRODUCTION

Noise is inherent in quantum computers, and if ignored, it will always destroy any quantum computational advantage. With advances in quantum hardware enabling controllable systems of hundreds of qubits, the use of quantum error correction to prolong the lifetime of quantum information is becoming feasible. At the hardware-theory interface, a key goal is to design optimal codes that leverage specific hardware-level advantages, such as gate nonlocality, to mitigate the effects of key challenges, such as the fidelity of few-qubit gates, or the resources required to increase the number of qubits in the system.

Quantum error correction is done by starting with a Hilbert space of n *physical qubits*, and identifying a subset $2^k < 2^n$ of the possible states within Hilbert space as encoding the wave function of k *logical qubits*. The smallest number of physical qubits on which a nontrivial logical operation can act determines the code distance d, and such a code is often abbreviated as $[\![n, k, d]\!]$. A practical set of codes are *stabilizer codes* [1] in which the logical codewords are the simultaneous $+1$ eigenstates of a commuting set of Pauli operators called the stabilizer group. A Calderbank-Shor-Steane (CSS) code [2,3] is a stabilizer code for which the generators of this set are strictly products of Pauli X's or Z's.

An important example of a CSS code is the toric code [4]. Together with its planar cousin, the surface code [5,6], they are leading candidates for near-term implementations of fault-tolerant quantum computation. It has local stabilizer generators supported on a checkerboard-style lattice; see Fig. 1. The toric code has been realized with neutral atoms [7], with fault tolerance later achieved in the surface code with superconducting qubits [8], although the "break-even" point after which the logical qubit is more robust than an isolated physical qubit remains to be reached. Even more recently, progress towards intermediate-scale fault tolerance has been demonstrated with 40 logical qubits encoded in color codes, a close relative of the surface code, using 280 neutral atoms [9]. For hardware with highly biased noise (e.g., Pauli Z error much more likely than Pauli X error), there exists elegant modifications to the surface code [10].

Due to the spatial locality of a surface code in two spatial dimensions, it is highly desirable for experimentalists; nearly all platforms, including atoms in optical tweezers [11–17], trapped ions [18–22], or superconducting qubits [23–27], can realize geometrically local interactions in two spatial dimensions. Unfortunately, quantum computation with $O(10^3)$ logical qubits in a surface code architecture with typical error rates of $O(10^{-3})$ may require an architecture with $O(10^7)$ physical qubits [28], which could be prohibitively difficult to build in the near term.

An exciting alternative are quantum low-density parity-check (qLDPC) codes, which can achieve $k \sim n$: the overhead for encoding logical information is finite. At the same time, the stabilizers are few-body just like the surface code (but not necessarily spatially local), meaning they can in principle be measured efficiently using few-qubit operations. The first qLDPC construction with a finite rate ($k \sim n$) and large distance ($d \sim \sqrt{n}$) was the hypergraph product (HGP) [29]; a series of improvements [30–32] eventually led to "good" codes with $k \sim d \sim n$ [33–35].

Spatial locality constrains the implementation of qLDPC codes in quantum hardware. Suppose that each physical qubit is arranged in a two-dimensional grid, and qubits can only

*Contact author: yifan.hong@colorado.edu
†Contact author: andrew.j.lucas@colorado.edu

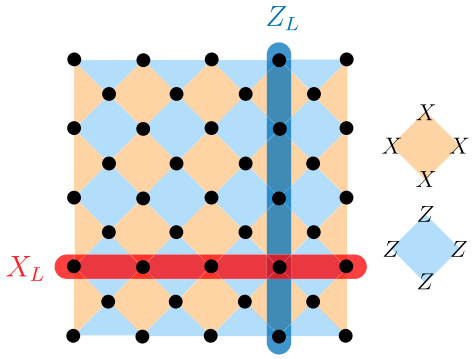

FIG. 1. The 2D layout of a $[\![41, 1, 5]\!]$ surface code. Black dots and colored tiles represent physical qubits and stabilizer generators, respectively. The stringlike logical operators are also depicted.

interact with other qubits a finite distance away. Then one can prove [36] that $kd^2 \lesssim n$: there is an unavoidable tradeoff between robustness to error (d) and number of logical qubits (k), given a fixed number of physical qubits (n). Conversely, it is known [37] that to implement a qLDPC in 2D, at least $\Omega(\sqrt{\frac{k}{n}}d)$ interactions of range $\Omega(\sqrt{\frac{k}{\sqrt{n}}})$ are necessary. If we only ask for $d \sim \sqrt{n}$ as in the surface code, the bounds of [37] alone admit the prospect of $k \sim \sqrt{n}$ using interactions of $O(1)$ range. Since [36] proves that these finite-range interactions only allow $k = O(1)$, the cost of nonlocality in qLDPC codes is even higher than implied by [37]. Further challenges to qLDPC implementation in 2D were discussed in [38,39]. It is thus of crucial interest to know the following: how many nonlocal stabilizers are needed to add logical qubits to a surface code, while keeping d and n fixed? If we find a code that uses the least nonlocality to add logical qubits to the surface code, is it realizable in any near-term quantum hardware?

This paper answers these questions. We present *long-range-enhanced surface codes* (LRESCs): an interpolating family of hypergraph product codes that bridges the surface code with constant-rate qLDPC codes. These codes (1) have as few nonlocal stabilizers as possible in the $k = O(1)$ limit, (2) maintain the code distance d of the surface code while adding logical qubits, i.e., increasing k, (3) have lower logical failure rates compared to a surface code under measurement noise, and (4) enable fault-tolerant gadgets to be inherited from those of classical codes. The simplest realization of the LRESC has a "hierarchical" structure similar to a recent construction [40]; however, unlike [40], LRESCs are LDPC stabilizer codes, employing as little nonlocality as possible. Moreover, as we will explain, these codes are well suited for implementation using multiple different architectures for quantum computation, as the specific form of nonlocality required by LRESCs is far more efficient to implement than a generic qLDPC code.

II. THE LRESC

A. Construction

We begin by summarizing intuitively the structure of LRESCs; technical details are provided in Appendices. Our construction consists of three parts, visualized in Fig. 2.

(1) First, begin with a classical base code. For instance, one can pick a "good" classical LDPC (cLDPC) code [41,42], which uses L_0 classical bits to store $\Theta(L_0)$ logical bits with $\Theta(L_0)$ distance. In this paper, we will focus on relatively small code sizes where $L_0 \sim 3 - 10$, both for pedagogy and near-term relevance. Note that a good cLDPC code will require nonlocal parity checks, with $\Omega(L_0)$ range in general, between the classical bits to ensure constant rate. Appendix A overviews classical codes.

(2) Next, we increase the number of classical bits: $L_0 \to cL_0 = L$, while proportionally increasing the distance of the code to $\Theta(L)$ and keeping the number of logical bits fixed as $\Theta(L_0)$. We can do so by replacing the bits of our starting code

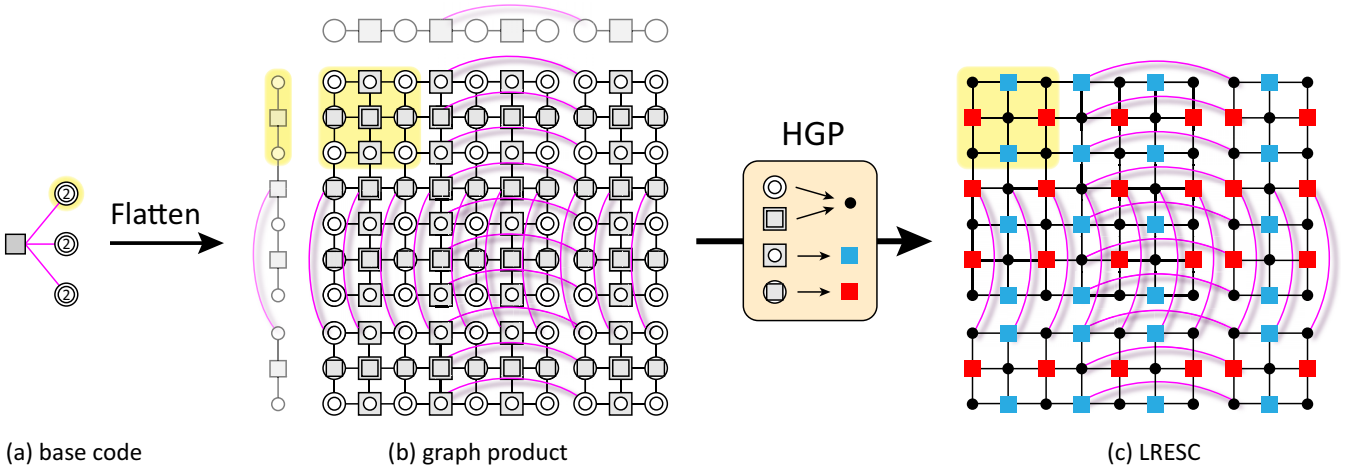

FIG. 2. The steps to construct a $[\![52, 4, 4]\!]$ HGP code based on a parent [3(2),2,2(2)] concatenated code are illustrated. (a) The Tanner graph of the base [3,2,2] code is depicted where the square and circles represent the parity check and bits, respectively. The double concentric circle with a "2" in the middle means each bit in this base code is actually the logical bit of a length-2 inner repetition code after concatenation. (b) The Tanner graph of the concatenated code is flattened in 1D, and then a Euclidean graph product is taken that produces four types of vertices in a 2D embedding. (c) The HGP procedure transforms the product graph into a CSS Tanner graph. Twenty long-range interactions (magenta curves) of range 4 are required for this code. A $k = 4$, $d = 4$ surface code of the same layout will require $n \geqslant 100$ physical qubits.

with another classical code, such as the repetition code, which stores a single logical bit in c physical bits, with codewords $0 \to 0\cdots0$ and $1 \to 1\cdots1$. The repetition code has spatially local parity checks when bits are laid out in one dimension: the parity checks demand that the parity of two nearest-neighbor bits agree. We thus build a *concatenated* code by replacing the cLDPC "physical bits" with repetition codes of length c. There is no code with fewer nonlocal edges in one spatial dimension that has $\Theta(L_0)$ logical bits and $\Theta(L)$ code distance (see Appendix A). Decoding this concatenated code can be done in two steps: we first decode each repetition code, and then decode the size-L_0 LDPC using the state of each repetition code as an effective "physical bit."

(3) We now build a quantum code by taking the *hypergraph product* (HGP) of this classical concatenated code with itself. A formal definition of the HGP is technical and relegated to Appendix B; Fig. 2 sketches the idea. We lay out two copies of the classical code of length L along the x and y directions in the plane. Based on the connections between checks and physical bits of the classical code, we lay out physical qubits and X and Z type stabilizers of the quantum code in two dimensions. Note that the hypergraph product of two classical repetition codes is the quantum surface code. Since our classical codes contain repetition code segments, our quantum code consists of two-dimensional surface code patches. Long-range parity checks from step (1) induce stabilizers with range $\Omega(L)$ that connect distant patches in the code, while ensuring that each stabilizer itself has low-weight [is a product of $O(1)$ X's or Z's].

These are the LRESCs. The total number of physical qubits is $n \sim L^2$, the quantum code distance is $d \sim L$, and the number of logical qubits is $k \sim L_0^2$. Alternatively, we have constructed a code with $d \sim \sqrt{n}$, just like the surface code, but where we have added k logical qubits at the cost of adding $O(L\sqrt{k})$ long-range stabilizers.

B. Bounding nonlocality

There is no code that has parametrically fewer long-range stabilizers in 2D than an LRESC, while maintaining the same code distance $d \sim \sqrt{n}$ and adding O(1) additional qubits beyond the bound of [36]. To see why, start with a 2D code $\mathcal{C}_0$ whose checks have support only on qubits within a ball of radius $r_0 = O(1)$ and whose distance satisfies $d_0 \sim \sqrt{n}$. BPT [36] tells us that there is at most a finite $k_0 = O(1)$ number of logical qubits, such that $k_0 d_0^2 \leqslant Kn$ for some $K = O(1)$. To increase the number of logical qubits from k_0, we will need to add longer-range interactions: how many are required? Suppose that by modifying $\ell \ll d_0$ of our local stabilizers to be spatially nonlocal (but still low weight, i.e., a few Paulis), we obtain a new code $\mathcal{C}_1$ with modified parameters $[\![n, k_1, d_1]\!]$. Further suppose that we have $d_1 \sim d_0$. Then, since the new code is still LDPC, the size of the region R which contains the long-range checks is $O(\ell) \ll d_1$ and hence a correctable region. The Cleaning lemma [43] ensures that we may choose all logical operators such that they are supported only on sites outside of the support of these long-range checks, as shown in Fig. 3. Since $\mathcal{C}_0$ and $\mathcal{C}_1$ share the same local checks outside of region R, and we have only cleaned the logical operators off of at most $O(r_0^2 \ell) \ll d_0$ sites, the cleaned logical operators

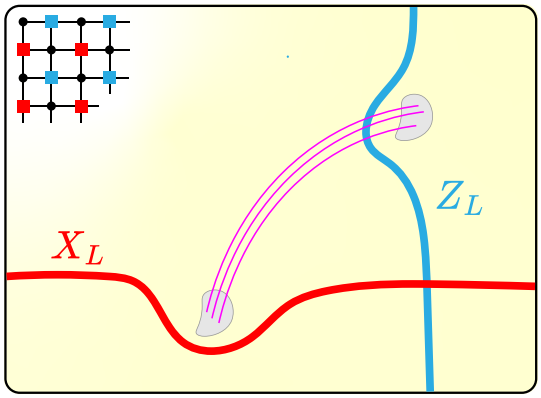

FIG. 3. A geometrically local 2D stabilizer code (surface code depicted) is modified with $\ell \ll d$ long-range interactions (magenta curves). For this modified code, all logical operators (two shown as thick red and blue curves) can be cleaned to exist completely outside of the support of the long-range checks (gray regions).

are also valid logical operators for the original local code $\mathcal{C}_0$; the number of such logical operators for inequivalent logical qubits is hence bounded by BPT: we can have no more than k_0 logical qubits.

The only way around the above argument is to relax the condition that the long-range region is correctable so that we cannot clean out all logical operators. As a consequence, there would then exist a logical operator supported entirely within this region, and so we would have $d_1 \ll d_0$. Thus, to add logical qubits to a surface code without sacrificing distance, $d \sim \sqrt{n} \sim L$ long-range stabilizers are required. The LRESC achieves this scaling, and is parametrically as local as possible for finite $k > k_0$. Interestingly, adding logical qubits beyond the $\ell = \Omega(d)$ restriction need not require additional long-range interactions, but rather a "rewiring" of them. For example, consider an $L \times L$ surface code and add a line of L qubits encoded in a good $[\![L, \Theta(L), \Theta(L)]\!]$ quantum LDPC code. Then the number of logical qubits is $k \sim L$ with $\ell \sim L$ long-range interactions.

Note that the Cleaning lemma is crucial to the above argument. It has no direct classical analogue, which is why we could improve the classical code dimension with only $\ell \ll d$ long-range interactions while maintaining the scaling of the distance. The physical intuition for the Cleaning lemma is that due to the unitarity of quantum mechanics, a correctable region must not reveal any encoded information; otherwise it may be possible to violate the No-Cloning theorem. As a consequence, a small correctable region is effectively "invisible" to the logical codespace. For classical codes this is not true: in the repetition code, the value of a single physical bit reveals the value of the logical bit.

C. Quantum error correction

Quantum error correction (QEC) for stabilizer codes is typically done by extracting the eigenvalues of all stabilizers, which can be deduced by measuring a set of generators called the check set; the outcomes of these measurements comprise the error syndrome. Decoding then proceeds by finding a suitable correction operator according to the syndrome. The

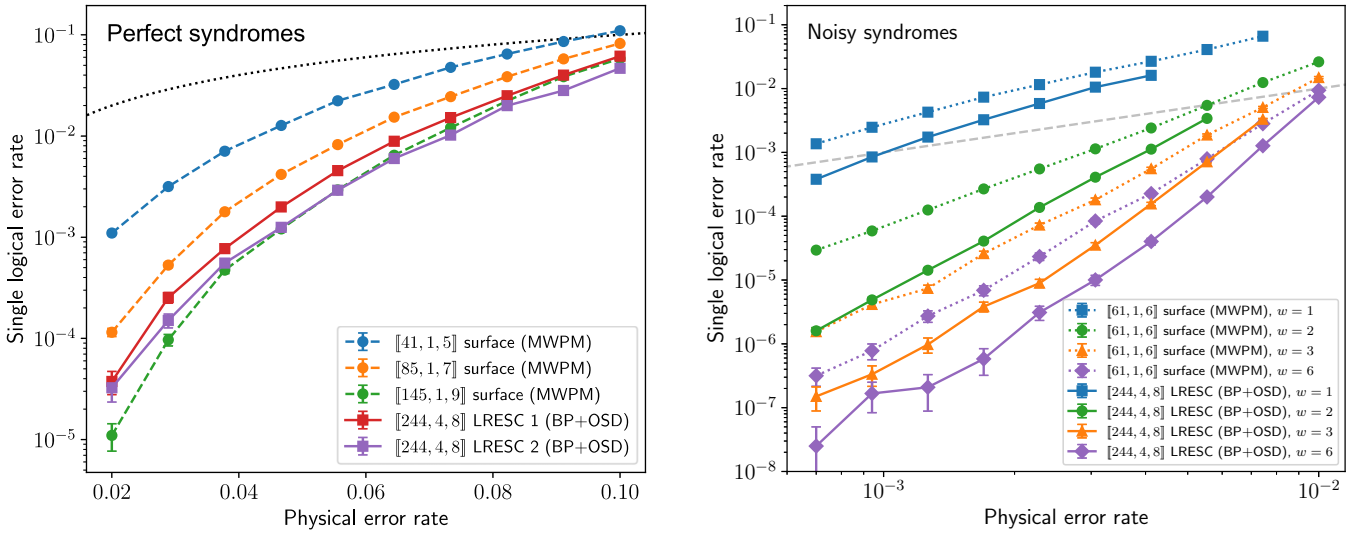

FIG. 4. QEC performance is numerically estimated using MWPM decoding for three surface codes with increasing distance as well as BP + OSD for LRESCs from Table I. Left: $\sim 10^5$ clean QEC cycles are averaged per data point. Right: A comparison between a $d = 6$ surface code and LRESC 2 is shown for $\sim 10^4$ samples of 100 noisy QEC cycles. Decoding is performed using a sliding window technique with variable window size w. Uncertainties are given by standard errors. A round of noiseless decoding is performed internally after each noisy cycle to probe the residual errors. The break-even line is plotted in dashed gray.

combination of the original error and the correction then either leaves the codespace unchanged (success) or enacts an undesirable logical operation (fail). For some codes, such as the surface code, when syndrome measurements can be faulty, it is necessary to perform several rounds of syndrome measurements and collectively decode over a "spacetime" decoding graph [44].

We conduct numerical simulations of QEC using both a code-capacity (clean syndromes) and a weighted phenomenological (noisy syndromes) noise model under a local, stochastic depolarization channel (single-qubit X, Y, or Z errors are equally likely) with probability p: see Fig. 4. For the phenomenological noise model, we scale both the physical error and the syndrome measurement error rates according the degree distribution of the Tanner graph: a qubit participating in v checks has an error rate of vp, and a check with weight w is incorrectly measured with probability wp; this mimics the experimental way that such syndromes are measured, as we will explain later. We implement a single-stage decoder utilizing belief propagation with ordered-statistics [45,46] postprocessing (BP + OSD), which has been previously shown to have favorable performance as a general-purpose qLDPC decoder. We use the "min-sum" and "combination-sweep" ($\lambda = 30$) variants of BP and OSD from open-source software [47]. Syndrome errors are accounted for by adding an additional variable node for each check node in the Tanner graph [48,49]. For the phenomenological model, we perform 100 noisy QEC cycles for each Monte Carlo trial. When syndrome errors are present, decoding is not perfect, and there will often be a "residual error" that is carried onto the next QEC cycle. To ensure that this residual error is not detrimental, we perform noiseless decoding after each noisy cycle as an internal flag to ensure that residual errors are successfully controlled; the QEC simulation outputs a failure if the residual error cannot be properly decoded. We decode using a "sliding window" technique [44], where the correction at time t is determined using the syndrome information from times t to $t + w$ with window size w. Recently, it was shown that this technique considerably reduced logical error rates for several classes of qLDPC codes [50] beyond the single-shot regime ($w = 1$).

TABLE I. The number of long-range interactions vs total number of pairwise interactions are displayed for three LRESCs. Optimized embeddings of the Tanner graphs lower the number of long-range interactions (in parentheses).

LRESC	Edges	LR edges (optimized)	LR ratio
⟦244, 4, 8⟧	924	60 (20)	6.5% (2.2%)
⟦244, 4, 8⟧	1056	528 (176)	50% (16.7%)

To construct the LRESCs, we use parent codes (1) [3(4),2,2(4)] and (2) [6(2),2,4(2)], where $[n'(c), k', d'(c)]$ is short for an outer $[n', k', d']$ code concatenated with an inner $[c, 1, c]$ repetition code. The corresponding LRESCs have mutual parameters ⟦244, 4, 8⟧ with rate $k/n \approx 1.64\%$ (61 physical qubits per logical qubit). The second LRESC contains more long-range interactions than the first; see Table I. The performance of BP + OSD decoding on these LRESCs is compared with that of minimum-weight perfect-matching (MWPM) on $d = 5, 7, 9$ surface codes. For the MWPM simulations we use the open-source software PyMatching [51]. We observe that the first two LRESCs perform similarly to surface codes of similar distance under the code-capacity model. The benefits of the long-range interactions are revealed when syndrome noise is taken into consideration. Under the phenomenological noise model, we observe that the second LRESC performs considerably

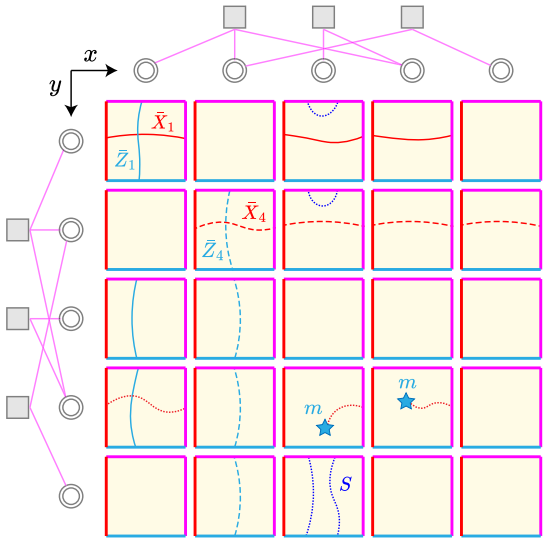

FIG. 5. A 2D layout of a LRESC with an outer [5,2,3] parent code is shown. Smooth (X-type) and rough (Z-type) boundaries are located on the left (thick red) and bottom sides (thick blue) of each patch, respectively, with long-range boundaries on the other (magenta) sides. Four logical operators ($\bar{X}_1, \bar{Z}_1, \bar{X}_4, \bar{Z}_4$) corresponding to two logical qubits are drawn (solid and dashed curves). A Z-type stabilizer and an X-type error string are also shown (dotted red and blue).

better than a surface code with the same rate for window sizes $w = 1, 2, 3, 6$.

With an encoding rate of 61 physical qubits per logical qubit, the LRESCs begin to significantly outperform the surface codes of similar rate. With hundreds of physical qubits, an LRESC can surpass the break-even point—where the collective logical qubit is more stable than a single isolated qubit—once one- and two-qubit operations are achieved with $\gtrsim$99.5% fidelity, which in many platforms is near-term [16,17] or within reach [52–54].

D. Logical operators

To understand why the LRESC not only stores more logical qubits, but also has reduced logical error rates, we need to understand how LRESCs encode logical qubits. As hinted at previously, since logical operators *locally* look like repetition codes in the concatenated cLDPC codes (Step 2 above), in the HGP, logical operators *locally* look like surface code logicals, which are strings of Pauli X or Z stretching across a surface code patch. What differs from the usual surface code is the global structure of the logical operator, i.e., how strings in different patches are joined together. A sketch is shown in Fig. 5, with technical details in Appendix C. In a nutshell: the simplest logical operator in an LRESC corresponds to strings in $O(\sqrt{k})$ of the surface code patches, corresponding to an analogous logical codeword of our cLDPC from Step 1 above.

We can intuitively understand why LRESCs are more effective at protecting logical information by showing that no matter how a logical error forms via local processes, during the formation of the error we always violate more check operators than in an ordinary surface code. Since more checks are violated, we have more opportunities to catch the physical qubit errors before they introduce a logical error. In the surface code, we can create a logical error by introducing a physical error near one boundary and then causing a cascade of additional errors on adjacent sites, i.e., growing a logical string in Fig. 5. At any step during this process, for the ordinary surface code, only one check is violated, meaning the error is almost undetected. In condensed-matter physics, we can interpret this as an *anyonic* particle that is free to diffuse around the system. In the LRESC, we can similarly grow an error through a single patch; however, when the error hits the long-range boundary, it will flip *multiple* checks in adjacent patches (anyons are not conserved across the long-range boundaries of the LRESC). The rules for anyon splitting are discussed in Appendix C. Since the error must grow across multiple patches to constitute a logical, we must inevitably flip more checks during the formation of the logical error, implying that it is easier to detect.

E. Logical gates

Implementing one- and two-qubit logical gates on an LRESC is (in principle) quite simple due to its similarly to a surface code. By readily organizing one of our logical qubits into a contiguous surface-code patch (e.g., moving surface-code patches in Fig. 5 so that a logical string becomes "continuous" and adjacent to the global boundaries), we can apply generalized lattice surgery techniques to perform Clifford gates [55]. Note that one will require surface code patches of $O(n)$ physical qubits to implement logical gates on $O(1)$ logical qubits. Alternatively, one can use the same lattice surgery to teleport a logical qubit onto a surface code [56]. Once a logical qubit is in an ordinary surface code patch, standard methods [57,58] can then be used to apply all logical Clifford operations in a fault-tolerant way. This process can be repeated to pass multiple qubits into surface code patches, onto which two-qubit gates can be fault-tolerantly applied. As an alternative, some work has been done on applying logical Clifford gates without the need for lattice surgery by generalizing hole-braiding in the surface code [59].

Transversal gates have both constant spatial and temporal cost due to their inherent parallelization at the physical level, but a hypergraph product code has yet to be found that can transversally implement the entire Clifford group on all its logical qubits. Some progress on transversal gates has been previously made, which can fill the Clifford group when supplemented with more expensive techniques such as code switching and state injection [60]. Because the previous techniques apply to generic hypergraph product codes, they apply to LRESCs as well.

We now build upon these previous results by presenting additional transversal gates in specific LRESCs, inherited from those of specially structured parent codes, with the amount of fault tolerance, quantified by transversality, tuned by adjusting the size of the local surface-code patches (i.e., the concatenation parameter c). We showcase an example using the [3,2,2] parity code, with details regarding the general procedure in Appendix D. The [3,2,2] code is the dual of the 3-bit repetition code with a single parity check $H = (1\ 1\ 1)$ and two logical bits with codewords $\bar{1}\bar{0} = 101$ and $\bar{0}\bar{1} = 011$. The

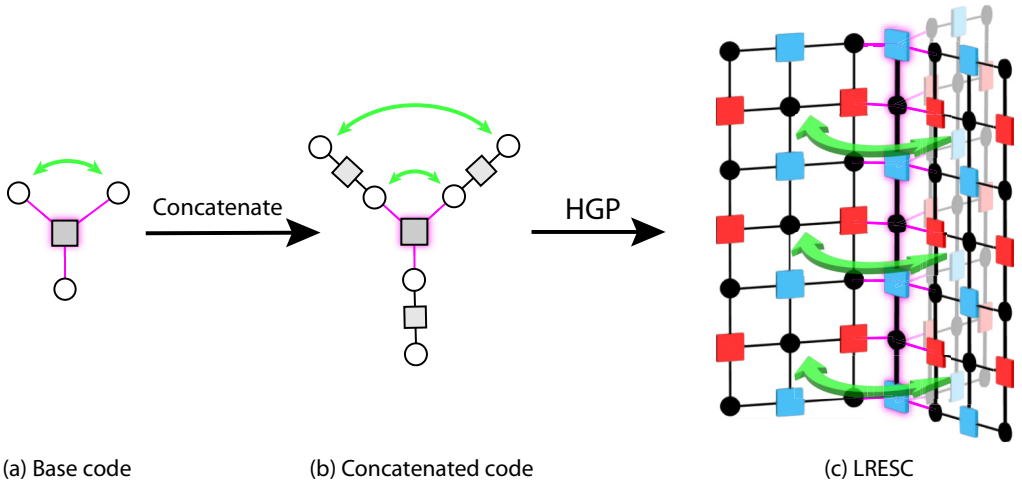

FIG. 6. The inheritance of the SWAP-CNOT logical gadget from the [3,2,2] base code to the corresponding LRESC is illustrated. (a) The Tanner graph of the [3,2,2] is shown, where circles and squares represent bits and checks, respectively. A logical CNOT is performed by swapping the bits according to the green arrow. (b) Upon concatenation with a repetition code, the bits in the base code now become segments of repetition code. The single SWAP in the base code now becomes a segment-transversal SWAP between the respective repetition codes. (c) The hypergraph product code using the previous concatenated code with a repetition code of length 4 is depicted. The long-range components are organized into a single central column to which the surface-code patches are attached at their boundaries. The logical CNOT is implemented by a patch-transversal SWAP between the corresponding patches of surface codes.

logical $\text{CNOT}_{\bar{1}\to\bar{2}}$ between the first (control) and second (target) logical bits maps $\bar{1}\bar{0} \to \bar{1}\bar{1}$ and $\bar{1}\bar{1} \to \bar{1}\bar{0}$ while leaving the other two logical bitstrings invariant. This transformation can be realized by physically swapping the second and third bits. The complementary logical $\text{CNOT}_{\bar{2}\to\bar{1}}$ with control and target switched is realized by physically swapping the first and third bits. Note that the single 111 parity check remains invariant under both of these physical SWAPs.

When we concatenate with a 1D repetition code of length c, we obtain a $[3(c), 2, 2(c)]$ code. The codewords for this new concatenated code mimic those of the original base code but each 0 and 1 now become strings of 0s and 1s of length c. Because the values of the bits along each repetition-code segment are simply copies of the original bits in the [3,2,2] base code, any transformations of the base code now become *segment-transversal* in the concatenated code: we simply apply the same transformation in parallel to all physical bits in the corresponding repetition codes. See Fig. 6(b) for an illustration.

Taking the hypergraph product of the above concatenated $[3(c), 2, 2(c)]$ code with a 1D repetition code results in a quantum CSS code that can be arranged as three surface-code patches connected by a shared central boundary; see Fig. 6(c). One can quickly verify by inspection that the stabilizer checks remain invariant upon swapping any two surface-code patches due to the "fold" symmetry about the central column. In addition, like the segment-transversal implementation of the concatenated parent code, this patch swap can be implemented in a *patch-transversal* manner: apply parallel SWAPs between qubits paired under the "fold" symmetry. Choosing our "horizontal" logical $\bar{X}$ operators to mimic the structure of the codewords of the parent $[3(c), 2, 2(c)]$ code, we can ensure that swapping surface-code patches enacts the correct CNOT transformation on the logical $\bar{X}$ operators. It suffices to verify the SWAP action on the logical $\bar{Z}$ operators. Choose $\bar{Z}_1$ and $\bar{Z}_2$ to be "vertical" strings living in the first and second patches, respectively. Then swapping the second and third patches leaves $\bar{Z}_1$ invariant while moving $\bar{Z}_2$ from the second to the third patch, which becomes stabilizer-equivalent to $\bar{Z}_1\bar{Z}_2$. Similarly, swapping the first and third patches leaves $\bar{Z}_2$ invariant while transforming $\bar{Z}_1 \to \bar{Z}_1\bar{Z}_2$. So we see that the patch-transversal SWAPs successfully implement the desired logical CNOT gates.

We conclude this section with some comments regarding the general procedure for logical gate inheritance and implementations for non-Clifford gates. The remarkable feature of performing entangling gates at the logical level with nonentangling gates at the physical level stems from the long-range interactions of the LRESC: the codespace contains additional entanglement from the long-range interactions, and simply rearranging this entanglement is enough to couple logical qubits. In the same example, we could have also taken the hypergraph product of the $[3(c), 2, 3(c)]$ code with itself to obtain a LRESC with four logical qubits. We would then have four logical gadgets comprised of simultaneous CNOTs between $(\bar{1}\bar{2}, \bar{3}\bar{4})$ and $(\bar{1}\bar{3}, \bar{2}\bar{4})$ with either left or right logical qubits as control or target. In general, we will only be able to perform simultaneous logical gates along "rows" and "columns" of logical qubits. Performing arbitrary two-qubit logical CNOTs in this setting may require the code-switching techniques of [60]. The [3,2,2] code is also the smallest code in the family of (shortened) Hadamard, or equivalently dual Hamming, codes with parameters $[2^k - 1, k, 2^{k-1}]$. This family of codes is equidistant: all codewords have weight $d = 2^{k-1}$. As a consequence, the SWAP-CCNOT gadget for the [3,2,2] code generalizes to this entire code family, and any two-bit logical CNOT gate can be implemented by physical SWAPs. However, these codes are not LDPC and so may be difficult to implement in a fault-tolerant setting. In addition, in the corresponding hypergraph product codes, one also needs

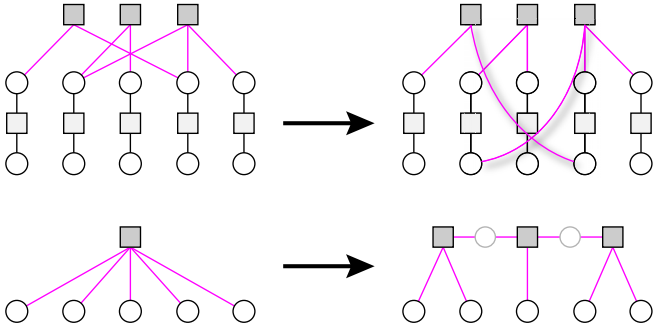

FIG. 7. The two different weight-balancing procedures are depicted. Top: a [5(2),2,3(2)] code is modified so that all physical bits participate in at most one long-range parity check. Bottom: A weight-5 parity check is decomposed into three weight-3 checks with two additional auxiliary bits (gray circles).

to track the action of the SWAPs on the parity-check matrices (detailed in Appendix D), which will in general not be invariant under these SWAPs. So some level of concatenation with repetition codes (thus creating LRESCs) may be necessary to maintain fault tolerance. Nonetheless, for small base code sizes, the Hadamard codes provide many new additional CNOT gadgets available to their corresponding LRESCs, which can reduce the need for code switching or lattice surgery. The SWAP-CNOT gadget is also very amenable for near-term experiments where the dominant source of error is from two-qubit entangling gates, as we will explain in Sec. III.

To achieve a universal gate set, the Clifford group needs to be supplemented with a non-Clifford gate, such as the T gate ($\pi/8$ rotation). Since logical operators of HGP codes can be chosen to be perpendicular "strips" intersecting on only one qubit (recall Fig. 5), we do not expect non-Clifford gates to be transversally implementable [61]. Nonetheless, as previously mentioned, one can teleport logical qubits onto ordinary surface codes [56], from which magic state distillation [62,63] can subsequently be applied, though [64–66] provide alternatives.

F. Weight balancing

There are two simple but practical enhancements to the LRESC described thus far by modifying the parent codes. In Step 2 of the LRESC construction, notice that each "physical bit" of the cLDPC from Step 1 consists of a repetition code, but we assigned all of the "long-range" parity checks to a single bit. We can instead evenly distribute these parity checks to different bits inside of the repetition code: see Fig. 7—so long as c is larger than the maximal number of parity checks per bit of the cLDPC (Step 1), this will mean that each physical bit is involved in at most one long-range parity check in Step 2.

The second modification is to introduce auxiliary bits into the parent codes in order to reduce the weight of each long-range parity check. The parity-check constraints of a classical code can be reformulated as a boolean satisfiability problem (SAT). It is well known in computer science that any SAT problem can be decomposed into conjunctions of smaller SATs of maximum size three (3-SAT), with the potential of introducing some auxiliary bits. Moreover, this SAT $\rightarrow$ 3-SAT decomposition can be performed in polynomial time [67]; for our linear constraints, this decomposition takes a particularly simple form; see Fig. 7. When we apply this decomposition to the parity checks of a classical code, we obtain new parity checks with bounded weights $\leqslant 3$ acting on the combination of our original physical bits and some new auxiliary bits. Importantly, the code distance remains unchanged, though the relative distance may decrease by an $O(1)$ factor if this method is applied to all checks. At the quantum level, this decomposition bears resemblance to a measurement-only version of Shor's cat-state syndrome extraction circuit [68], where we have included the cat-state ancillas and measured operators as auxiliary qubits and new stabilizer checks, respectively.

The modified parent codes will now have at most weight-3 parity checks with each physical bit participating in at most one long-range interaction. Furthermore, by arranging each long-range parity check to be adjacent to an end point of a repetition-code segment, we can always localize at least one of its long-range edges. In turn, the LRESCs will contain at most weight-6 stabilizer checks with each physical and ancilla qubit participating in at most four long-range interactions. A major caveat of the weight-balancing procedure is that we will lose the concatenated structure for logical gate inheritance described in Sec. II E. Nonetheless, if we simply desire to use an LRESC as a quantum memory block, then the two "weight-balancing" procedures will be particularly advantageous for experimental implementations, as we will discuss in Sec. III.

III. EXPERIMENTAL IMPLEMENTATIONS

Typically, experimental design of quantum hardware has been strongly limited by the choice of QEC code and its resulting requirements on circuit connectivity. LRESCs imply that one can exploit the tunable addition of nonlocality on top of the most local of codes, the surface code, once improvements in physical error rates, or increases in physical qubit number, have been exhausted. Such a theoretical advance offers a timely new tool for improving the performance of state-of-the-art platforms, including superconducting qubits [23–27], trapped-ions [18–22], and neutral-atom arrays [11–17] since, as we now explain, the specific type of nonlocality needed for the LRESC is relatively mild in multiple experimental platforms.

In superconducting circuits, novel circuit graphs have simulated many-body physics in novel geometries [69]. To realize the LRESC, one must use multiple planes of wiring [70], and we expect that this construction is doable for modest values of k (i.e., not encoding too many logical qubits). For devices with larger values of k, we can also employ fault-tolerant quantum repeater networks [71,72] to teleport ancilla qubits down a strictly two-dimensional architecture. The number of such quantum repeater rounds is constrained by the requirement that we cannot pass two logical qubits "through each other." This latter construction is quite similar to the "hierarchical codes" recently discussed in [40].

While a superconducting-qubit-based quantum computer may take advantage of LRESCs or hierarchical codes, we believe that the LRESC is significantly more optimized for architectures capable of nonlocal gates and reconfigurability, namely trapped ions and neutral atom arrays. Using the

race-track geometry, trapped-ion quantum computers have shown all-to-all connectivity with high fidelity gates and midcircuit operations, including measurement and reset, which have been exploited for state-of-the-art demonstrations of quantum error correction [73]. Meanwhile, recent advances in neutral-atom quantum computers using reconfigurable tweezer arrays have enabled demonstrations of the circuit encodings for a variety of quantum error correction codes [7,9]. Importantly, while both platforms fundamentally allow for all-to-all connectivity, the qubit moves required for this are up to a few orders of magnitude slower than other operations, such as single- and two-qubit gates [7,9,73]. Accordingly, in order to develop optimally performing quantum computers—constrained by hardware level limitations associated with qubit number, circuit encoding time, operational fidelities, etc.—quantum error-correcting codes are desired that allow for flexible optimization within the space of these constraints. The LRESC construction allows for precisely this flexibility within these two platforms, by balancing physical qubit number against nonlocality and computation time in order to optimize encoding capacity. The proposed implementations below reflect this perspective.

A. Trapped ions

The quantum charge-coupled device (QCCD) approach [74] to quantum computation with trapped ions has the potential to realize LRESCs in the near future. This architecture relies on a trap device capable of confining multiple one-dimensional arrays of ions. Within these so-called "ion crystals," multiqubit operations are achieved through laser- or microwave-induced spin-motion couplings. To facilitate interactions between ions initially residing in separate ion crystals, the architecture requires real-time shuttling, splitting, and merging operations of ion crystals that occur on faster timescales compared to the coherence time of the data qubits. This dynamic control over system connectivity is made possible through precise manipulation of electric fields that generate the trapping potentials.

The high operational fidelities (up to 99.9999% single-qubit fidelity [75] and 99.94% two-qubit fidelity [76]) allowed fault-tolerant demonstrations of quantum error-correcting codes encoding a single logical qubit in small-scale quantum processors [77,78]. As systems with 100s of controllable qubits become available in the near future, it will be feasible to incorporate LRESCs in the QCCD architecture. In particular, if state-of-the-art fidelities can be maintained for a large-scale device, then LRESC 2 from Fig. 4 significantly surpasses the break-even point with 244 physical qubits, and a similar number of ancilla qubits, assuming two-qubit fidelity from [76].

A possible implementation of LRESCs with trapped ions is shown in Fig. 8. The envisioned architecture is structured into multiple unit cells, each representing a surface code tile (yellow tile in Fig. 2). Within each unit cell, multiple interaction regions are designed to facilitate parallel single- and two-qubit gates. Every unit cell contains both the data qubits necessary for surface code operations and the necessary ancilla ions. Data qubits are transported between interaction regions to perform the necessary two-qubit gates. During transport, unwanted motional excitations may arise due to imperfect control of applied fields. To maintain high two-qubit fidelities, ancilla ions are then used to recool an ion crystal following a transport operation. While qubits primarily move within a unit cell, an LRESC requires sparse long-range operations. The QCCD architecture in Fig. 8 efficiently facilitates the parallel transport of multiple ions to different unit cells, thus minimizing the time associated with nonlocal operations.

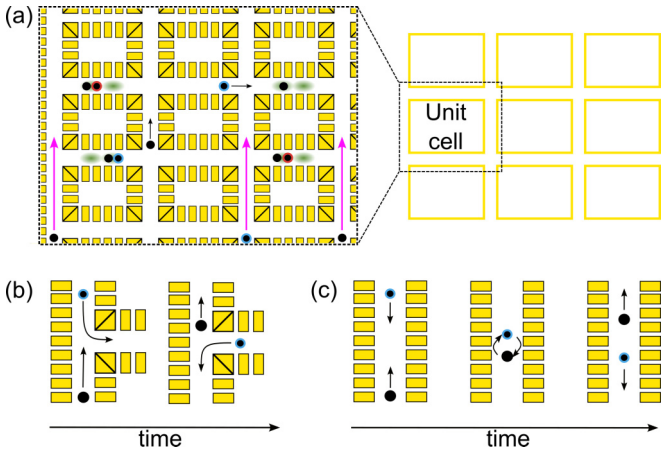

FIG. 8. LRESC implementation using a trapped-ion quantum charge-coupled device architecture. (a) A possible quantum processor is structured into multiple unit cells, each representing a surface code tile (yellow tile in Fig. 2). Each unit cell contains the necessary data qubits (black dots) and measure qubits for parity checks (blue and red rims). Ions are transported across different interaction regions (green-shaded areas) by precise control of the voltage applied to the trap electrodes (yellow boxes) to perform the necessary local operations. Each cell also contains additional ancilla qubits (not shown) used for recooling operations after transport operations. Sparse nonlocal operations required by LRESC are performed via long-range transport of qubits across different cells (pink arrows). (b),(c) Two possible ways to perform qubit permutation within the QCCD architecture.

The main complexity lies in the optimal scheduling of gates and transport operations, as well as the delivery of the laser light used to coherently control the qubits in each interaction region. For trapped ions, scalable laser light delivery based on free-space optics can be challenging due to the need for tightly focused beams for single-qubit addressing and the presence of nearby trap electrodes. The complexity is further increased if multiple wavelengths of light are needed in each region. A promising approach to address the issue of light delivery is using optical waveguides integrated into the structure of the ion trap. This approach allows compact routing of light to the various interaction regions and the generation of tightly focused beams using grating couplers [79,80]. Ion traps with integrated waveguides have been successfully employed in devices controlling a single region [80–82] as well as multiple ones [83,84]. Furthermore, experiments demonstrated the use of integrated waveguides with multiple wavelengths ranging from violet to infrared [81] as well as schemes for performing all qubit control with longer wavelengths of light [85,86], thus simplifying the waveguide requirements. To mitigate the challenges associated with ion transport and the scheduling of

gate operation, a possible solution is a system that allows the manipulation of ion crystals with more than two data qubits. Such a system not only would reduce scheduling complexity but is also likely to reduce the unit cell's size, as fewer interaction regions are needed. Consequently, it would also decrease the overall execution time, since the time required to transport and recool a crystal is generally longer than those of two-qubit gates in medium-sized ion crystals [73,87]. However, working with large ion crystals can add extra control challenges. State-of-the-art two-qubit gates between ions in long ion chains are generally slower than gates on two-qubit ion crystals and also yield lower fidelities [88,89]. Furthermore, multiqubit gates mediated by normal modes of motion cannot be easily executed in parallel. Therefore, we speculate that a likely optimal architecture that implements LRESCs will compromise the advantages offered by the QCCD architecture and those offered by the manipulation of medium-sized ion chains.

Depending on the details of the experimental apparatus (i.e., the physical size of the quantum processor, qubit coherence time, maximum achievable transport speed, and recooling times), long-range transport may cause an increased physical error rate due to the finite qubit coherence time and the longer time required for long-range ion shuttling. To mitigate this issue, teleportation of the qubit state can be employed to replace long-distance transport. This approach requires generating entangled Bell pairs between two distant regions of the quantum processor using schemes for remote entanglement generation [90–92]. This scheme would also be compatible with a modular ion-trap architecture [90] composed of multiple interlinked small devices each with a limited number of qubits and correspondingly little computational power [90,93].

B. Neutral atom arrays

Reconfigurable atom arrays manipulated with optical tweezers are also well-suited to reap the benefits of LRESC [7,9,12,13,94]. In particular, scaling to 100s of controllable qubits has already been demonstrated [95–97], while scaling to 1000s is a near-term prospect [98]; two-qubit gate fidelities of >98.5% have been shown in multiple atomic species, with the state-of-the-art performance at 99.5% [16,17]. Accordingly, this platform lies within an order of magnitude of the break-even point of an LRESC (see Fig. 4). Importantly, the optical methods used for atomic reconfigurability enable parallelism that is well-suited to the surface code and LRESCs [7,14].

Figure 9 illustrates a possible implementation of an LRESC using atom arrays. A static array—formed with a spatial light modulator or optical lattice [7,95–97,99]—holds atomic data qubits. The measure qubits that yield X and Z parity checks (red and blue rims) sit on a grid of traps rotated 45° from the x/y axes. This array of traps is formed with crossed acoustic-optic deflectors [AOD1-MQ, AOD2-MQ in Fig. 9(b)] driven with a comb of radiofrequencies. This entire array can be moved by adding an overall offset frequency to the comb of tones inside each deflector, allowing any rigid array translation in the $x-y$ plane. Such moves are used to bring all measure qubits into proximity with the appropriate

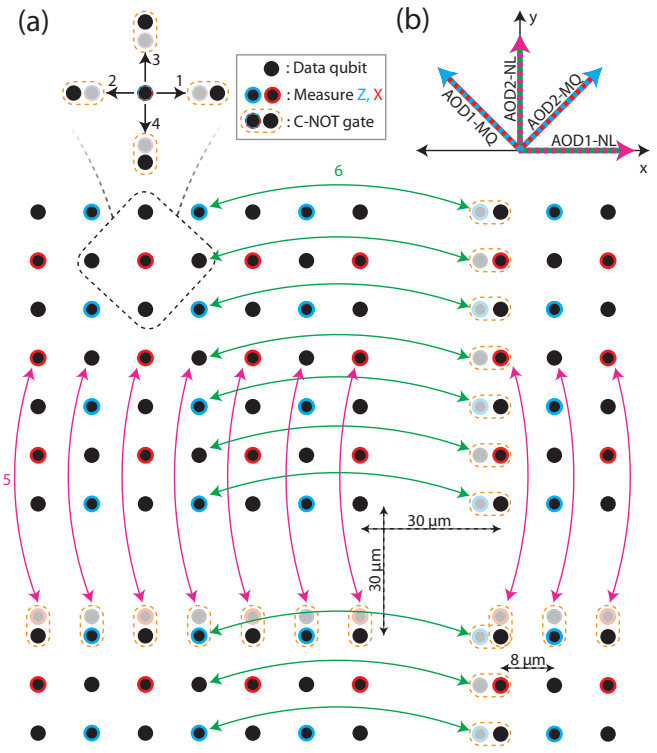

FIG. 9. LRESC implementation on a neutral-atom-based processor using Rydberg-mediated interactions. (a) Data qubits (black circles) and measure qubits (blue and red rims) are initialized in a static ordered 2D array generated by a spatial light modulator or optical lattice. Local parity checks are performed with sequential two-qubit gates (orange-dashed lines) performed on all measure/data qubits in parallel, where each measure qubit is transported in close proximity with a neighbor qubit (steps 1–4) using fast crossed acousto-optic deflectors (AODs). Another pair of crossed AODs is used to perform nonlocal operations by transporting data and measure qubits between different locations of the quantum processor. (b) Two different pairs of crossed AODs are used for short-range and long-range atom transport, respectively, labeled as MQ and NL. Arrows represent the transport direction for a varying radiofrequency offset in each AOD.

neighbor, in order to exploit short-range Rydberg-mediated interactions for a two-qubit gate (orange-dashed lines) for parallelized two-qubit gates [7,14]. Due to the short distance scales and the use of AODs, each stepwise move of the SC (top of Fig. 9) can be executed in $\lesssim 10$ μs.

The nonlocal gates that underlie LRESCs likewise can be implemented in a straightforward fashion, with one adjustment. A pair of crossed AODs [Fig. 9(b)]—AOD1-NL and AOD2-NL—can be used for row translations along the y direction [step 5 in Fig. 9(a)], as well as column translations along the x direction (step 6). For the nonlocal gates, both measure and data qubits are moved, which necessitates qubit transfers between different optical potentials. Such methods have been demonstrated and can be done while preserving coherence [9,100,101], yet they come at the price of longer timescales (∼100 μs) to mitigate motional heating.

In addition to allowing the core components of LRESCs, the atom array platform is compatible with other more general

needs of QEC. Initialization of the qubit array into the set of optical potentials discussed above can be accomplished with atomic rearrangement [99,102,103]. Parity checks on the measure qubits will require midcircuit readout and reset. This can be done *in situ* using qubit shelving methods, as recently demonstrated for ^{171}Yb or by using mixed atomic species [104–106]—this circumvents the need for large moves and zoned read-out [7,87]. Lossless state detection of neutral atoms can be slow (at best, a few milliseconds [105]); this timescale can be improved using destructive state detection [9,17,107], which is then paired with a qubit reservoir for rapid replenishment [106,108]. High-fidelity single- and two-qubit gates can be accomplished at low cross-talk with the qubit separations illustrated, using a combination of tightly focused and laser beams and globally addressing fields [16,17,105,109–113]. Qubit loss—a prevalent error channel during two-qubit gates and measurement—can be mitigated using syndrome extraction circuits and three-outcome measurements [114].

Finally, the weight-balancing procedures described earlier (Sec. II F), which allow for reducing the number of qubits per check (and checks per qubit), are relevant for the implementation of LRESCs in atom arrays. So long as each qubit participates in at most four long-range interactions, a single physical qubit will be involved in at most eight rounds of row and column swaps—four local and four nonlocal—to couple all corresponding physical and ancilla qubits during syndrome extraction for one round of QEC. Using a single AOD each for long-range row and column permutations, this may require $O(\sqrt{k})$ sequential swaps. These swaps could be further parallelized by carefully arranging the long-range edges, or by adding additional AODs, though we leave further optimization for future work.

IV. OUTLOOK

We have described the LRESC: a minimal generalization of the surface code capable of encoding multiple logical qubits without sacrificing code distance. We show how long-range interactions can (i) improve code overhead, (ii) improve code performance, and (iii) enable new kinds of fault-tolerant gadgets. The LRESC is well-suited for near-term hardware, where we anticipate that our fault-tolerant code might be realizable within the next few years.

An immediate direction for future work is to design a better decoder for LRESCs. Depending on qubit shuttling times, a more sophisticated two-stage decoder could be designed as follows. (i) Perform multiple rounds of local syndrome measurements in the surface code patches while waiting for the long-range syndrome measurements to complete [115]. (ii) Use one's favorite standard decoder (e.g., MWPM [44] or Union-Find [116]) for the multiround syndromes within the surface-code patches and feed the output decisions into a single-stage BP + OSD decoder for global decoding. In this manner, one can strike a balance between the "fast" (but less robust) checks of the surface code and the "slower" (but more robust) long-range checks. Another related avenue is the construction of additional fault-tolerant gadgets, taking advantage of specially structured base codes. While the more compact encoding of a LRESC facilitates the implementation of certain multiqubit logical gates, single-qubit logical gates become harder for the same reason. In an LRESC, the support of logical operators overlaps, and it is difficult to only manipulate one logical qubit without affecting others.

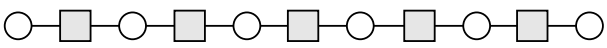

FIG. 10. The Tanner graph of an $n = 6$ repetition code is illustrated. The circles and squares represent physical bits and parity checks, respectively.

The construction of the LRESC also opens possible avenues to investigate new quantum phases of matter. In particular, it suggests new "topological phases" are enabled using only a small density of long-range interactions, and can thus be investigated in experiment. In the longer term, a large-scale LRESC may also be the foundation for an autonomous self-correcting quantum memory. Indeed, our proposed architecture may well represent a more convenient strategy for passive error correction versus a four-dimensional toric code [44]. It may also be more amenable to single-shot error correction than three-dimensional single-shot codes [117–119] due to its lower overhead.

ACKNOWLEDGMENTS

We thank Evan Wickenden and Charles Stahl for useful discussions, and Jeff Thompson for a careful reading of the manuscript. This work was supported by the Office of Naval Research via Grant No. N00014-23-1-2533 (Y.H., A.M.K., A.L.), the Alfred P. Sloan Foundation via Grant No. FG-2020-13795 (A.L.), NIST (A.M.K.), and the Swiss National Science Foundation under Grant No. 211072 (M.M.).

APPENDIX A: CLASSICAL LDPC CODES

We begin by reviewing classical low-density parity-check (cLDPC) codes [41], which play an important role in our construction. A classical linear code $\mathcal{C}$ is specified by a set of constraints called parity checks and a set of codeword generators satisfying those constraints. The state of the system can be represented as an element of $\mathbb{F}_2^n$, where $\mathbb{F}_2 = \{0, 1\}$, and in $\mathbb{F}_2$, $1 + 1 = 0$. We often represent the parity checks as rows of an $\mathbb{F}_2$-valued *parity-check matrix* H and logical codewords as rows of a matrix G. The statement that the codewords satisfy the parity-check constraints becomes $HG^{\mathsf{T}} = 0$. The dual code $\mathcal{C}^{\perp}$ is defined as the code where G and H are swapped. We say a linear code is LDPC if its parity-check matrix H is sparse: the number of ones per row and column are bounded by a constant irrespective of n. The code is useful if G is not sparse: the code distance d is the smallest number of 1s in a codeword. We can represent any linear code as a bipartite *Tanner graph*, drawing an edge between a "variable node" v and a "check node" c if the corresponding element of H is nonzero: $H_{cv} = 1$. The Tanner graph of a repetition code is depicted in Fig. 10. All linear codes satisfy the *Singleton bound*:

$$k \leqslant n - |C|, \tag{A1}$$

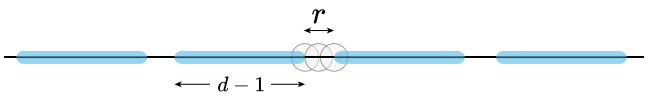

FIG. 11. A 1D chain is partitioned into disconnected, correctable regions (blue) of length $\approx d$ with separation $\approx r$ such that all checks (circles) have support in at most one region.

where C is a correctable region satisfying $|C| \geqslant d-1$. Correctable here means that all codewords can be successfully recovered upon erasure of C.

A code generated by a random sparse H has both $k = \Theta(n)$ and $d = \Theta(n)$ with high probability [42]; its corresponding Tanner graph is an asymptotically good expander [120]. However, if we arrange the variable nodes locally in one dimension, such a code will necessarily involve checks c that are nonlocal. If we enforce geometric locality in D-dimensional Euclidean space, then the code parameters must satisfy [36]

$$kd^{1/D} = O(n). \tag{A2}$$

The sketch of the proof in $D = 1$ is as follows. The idea is to partition the 1D chain into disjoint, correctable regions C_i of length $|C_i| \approx d$ where the separation between each region is large enough (say r) so that no parity check acts in more than one region; see Fig. 11. Since all the correctable regions do not share any checks, their union is entirely correctable [43]. The Singleton bound (A1) then imposes that $k \leqslant n - |C| = |\bar{C}| = O(rn/d)$, and we thus arrive at (A2) for $r = O(1)$ and $D = 1$. Now suppose we add in ℓ long-range connections to surpass (A2). We can simply avoid the long-range edges and partition the rest of the chain as before, arriving at $|C| \to |C| - O(\ell)$ and thus $k \to k + O(\ell)$. Hence, the number of logical bits k can scale at most linearly with the number of long-range connections ℓ.

We now saturate the asymptotic constraints above with a cLDPC code of $d = \Theta(n)$, k logical bits, and $\Theta(k)$ long-range checks. An $[n'c, k', d'c]$ code is produced from the concatenation of an "outer" $[n', k', d']$ code with an "inner" $[c, 1, c]$ repetition code of variable length c (denoted $[n'(c), k', d'(c)]$); see Step 1 of Fig. 2. Concatenation means that we connect a single bit of each inner repetition code to the parity checks of the outer $[n', k', d']$ code. This concatenating procedure can also be interpreted as first cutting up a 1D repetition code into disconnected segments and then reconnecting these segments with long-range interactions. The only long-range checks come from the outer code, and if it is a "good" $[n', k', d'] = [\Theta(k'), k', \Theta(k')]$ cLDPC code, the concatenated code has parameters $[\Theta(ck'), k', \Theta(ck')]$ with $\Theta(k')$ long-range connections, which is parametrically optimal. Since we are allowed to attach the long-range edges to *any* bits of the inner repetition codes, we have some flexibility in designing the long-range couplings (recall Sec. II F). This concatenation procedure can be considered as a "dual" variant to the edge-augmentation construction of [121]: instead of having the repetition codes live on the edges of a cLDPC code, we attach them to the variable nodes themselves. For a cLDPC with average vertex degree $\bar{w}$, the concatenated construction reduces the number of surface-code patches by a factor of $\bar{w}^2$ compared to the approach in [121]. As we will later see, the "hierarchical" structure of concatenated codes also lends the dynamics to be factorized in a systematic manner: we can analyze the dynamics within the inner and outer codes separately.

APPENDIX B: HYPERGRAPH PRODUCT CODES

Using an $\mathbb{F}_2^{2n}$ representation for Paulis, the stabilizer checks of a CSS code can be represented by the parity-check matrix

$$H = \begin{pmatrix} H_X & 0 \\ 0 & H_Z \end{pmatrix}, \tag{B1}$$

where commutativity requires $H_Z H_X^{\mathsf{T}} = 0$. We use the hypergraph product (HGP) [29] to construct a quantum CSS code from two classical linear codes. Specifically, suppose we have two classical codes with parameters $[n_1, k_1, d_1]$, $[n_2, k_2, d_2]$ and parity-check matrices H_1, H_2 with m_1, m_2 rows, respectively. The associated HGP code has parity-check matrices defined as

$$H_X = \left(H_1 \otimes \mathbb{1}_{n_2} \mid \mathbb{1}_{m_1} \otimes H_2^{\mathsf{T}}\right), \tag{B2a}$$

$$H_Z = \left(\mathbb{1}_{n_1} \otimes H_2 \mid H_1^{\mathsf{T}} \otimes \mathbb{1}_{m_2}\right). \tag{B2b}$$

By construction, the orthogonality constraint $H_Z H_X^{\mathsf{T}} = 2(H_1 \otimes H_2^{\mathsf{T}}) = 0$ is automatically satisfied. The $[\![N, K, D]\!]$ parameters of the HGP code are given by

$$N = n_1 n_2 + m_1 m_2, \tag{B3}$$

$$K = k_1 k_2 + k_1^{\mathsf{T}} k_2^{\mathsf{T}}, \quad D = \min\left(d_1, d_2, d_1^{\mathsf{T}}, d_2^{\mathsf{T}}\right), \tag{B4}$$

where k^{T} and d^{T} are the usual k and d for the transpose code. For the rest of the Appendix, we will use lower-case letters for classical code parameters and upper-case letters for quantum code parameters. If H_1 and H_2 have full rank (no redundant parity checks), then their transpose codes are trivial, and the above HGP code parameters (B3) simplify to

$$N = n_1 n_2 + m_1 m_2, \; K = k_1 k_2, \; D = \min(d_1, d_2). \tag{B5}$$

Geometrically, the Tanner graph of the HGP code takes the form of a Cartesian graph product between those of the two classical parent codes. Given two graphs $\mathcal{G}_1 = (V_1, E_1)$ and $\mathcal{G}_2 = (V_2, E_2)$, the product graph $\mathcal{G}_1 \times \mathcal{G}_2$ is a graph with vertices labeled by pairs (x, y) where $x \in V_1$ and $y \in V_2$. Two vertices (x, y), (x', y') are connected by an edge if either $x = x'$ and $\{y, y'\} \in E_2$ or $y = y'$ and $\{x, x'\} \in E_1$. The steps to convert this product graph into a CSS Tanner graph are as follows:

(i) If the vertex of the product graph is of the form (node, node) or (factor, factor), then that vertex becomes a node representing a physical qubit.

(ii) If the vertex of the product graph is of the form (node, factor), then that vertex becomes a factor representing an X stabilizer.

(iii) If the vertex of the product graph is of the form (factor, node), then that vertex becomes a factor representing a Z stabilizer.

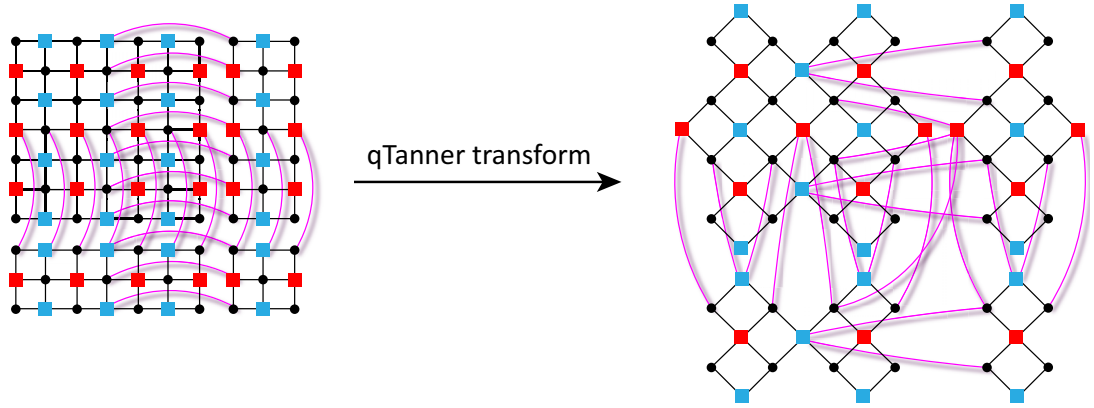

FIG. 12. The transformation of a $[\![52, 4, 4]\!]$ HGP code into a $[\![36, 4, 4]\!]$ quantum Tanner code is shown. Solid black dots represent physical qubits, and red (blue) squares represent X (Z) checks. Twenty-one long-range interactions (magenta curves) are required. A $k = 4$, $d = 4$ surface code of the same layout will require $n \geqslant 64$ physical qubits.

Importantly, if the two parent codes are LDPC, then so is the resultant HGP code. If the two parent codes can be locally embedded in D_1 and D_2 spatial dimensions, then the HGP code can in $D_1 + D_2$ dimensions. The surface code is the HGP of two 1D repetition codes.

The LRESC is simply the HGP of the classical concatenated code defined earlier with itself. Its parameters are $[\![\Theta(c^2k'^2), k'^2, \Theta(ck')]\!]$ with $\Theta(ck'^2)$ long-range interactions. Denoting $L \equiv ck'$ and $K \equiv k'^2$, the code parameters simplify as $[\![\Theta(L^2), K, \Theta(L)]\!]$ with $\Theta(L\sqrt{K})$ long-range interactions. For $K \ll N$, the 2D layout of this HGP code can be understood as patches of surface code of length c, whose boundaries are connected by long-range stabilizers; see Fig. 2. The graph product structure arranges these long-range interactions as parallel row and column couplings.

In the surface code, the quantum Tanner transformation [122,123] can reduce $N = D^2 + (D-1)^2$ to $N = D^2$ while maintaining the same distance, producing the so-called "rotated surface code" with parameters $[\![D^2, 1, D]\!]$. Examining Fig. 2, we see that the qubits of the HGP code can be partitioned into two sublattices corresponding to node-node (primary) and check-check (secondary) vertices of the graph product. The idea of the quantum Tanner transform is to multiply adjacent checks of the same type in order to produce new checks which commute when restricted to the primary sublattice; the secondary sublattice can then be discarded; see Fig. 12. Applied to a HGP code, the transformation will reduce $N = n_1n_2 + m_1m_2$ to $N = n_1n_2$. The Tanner transform of a LRESC will unfortunately introduce a "diagonal" interaction for every long-range 4-cycle in the original Tanner graph. If the parent code has $O(k')$ long-range edges, then the HGP code will contain $O(k'^2) = O(K)$ additional "diagonal" interactions in a 2D layout. For small platforms, the factor of ≈ 2 reduction in overhead may still be advantageous despite the increase in the routing complexity needed to implement the long-range checks. In practice, one will also need to worry about decreasing the effective code distance under circuit-level noise. For traditional HGP codes, it has been shown that the usual methods for syndrome extraction maintain the code distance [124]. After performing a quantum Tanner transformation, a specially engineered syndrome extraction circuit may be required to maintain this effective code distance (cf. surface code vs rotated surface code syndrome extraction circuits).

APPENDIX C: ANYONS, LONG-RANGE BOUNDARIES, AND CONFINEMENT

In this Appendix, we characterize the structure of logical operators in LRESCs using concepts from condensed-matter physics. We show how anyon transport properties in the LRESC are related to domain-wall dynamics in the classical parent codes. Finally, we describe how the long-range boundaries in an LRESC can lead to anyon confinement and improved single-shot decoding.

1. Logical operators and boundary dynamics

Interpreting the parity checks of the 1D repetition code (Fig. 10) as energetic terms in a Hamiltonian, we arrive at the 1D Ising model. A local bit flip in the 1D Ising model creates a pair of domain walls separating 1s and 0s. When these domain walls move via additional bit flips, the number of violated checks remains constant, and we say that the domain walls are "deconfined." These domain walls can then travel to opposite end points of the chain, flipping all physical bits in the process. Thus, a logical error in the 1D repetition code can be enacted with local processes while violating only $O(1)$ checks.

The concatenated codes mentioned earlier consist of an $[n', k', d']$ base code and an inner repetition code. We now describe how the structure of the base code dictates the dynamics of propagating domain walls. As a concrete example, suppose our base code was a [5,2,3] code with

$$H = \begin{pmatrix} 1 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 \end{pmatrix}, \quad G = \left(\begin{array}{cc|ccc} 1 & 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 & 1 \end{array}\right), \tag{C1}$$

where we have expressed G in reduced row echelon (standard) form. Domain walls can freely propagate within each inner repetition code. However, upon hitting the long-range boundaries, these domain walls will excite the long-range checks of

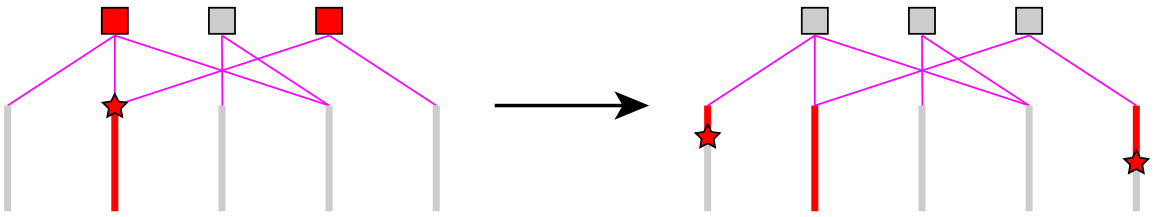

FIG. 13. The classical dynamics of long-range boundaries is depicted for a $[5(c), 2, 3(c)]$ concatenated code. A domain-wall excitation (star) is created in an inner repetition code and is transported across to the long-range boundaries (magenta lines). Left: Upon reaching this boundary, the long-range checks of the outer code will be violated (red squares). Right: Additional $d' - 1 = 2$ excitations must appear among the other connected surface-code patches in order to complete a logical operation (codeword 11001).

the base code: see Fig. 13. Satisfying the long-range checks requires locally exciting domain walls on other repetition code segments according to the codewords generated by G. When a domain wall reaches a long-range check, we examine the codewords which contain a 1 at the position of its corresponding repetition-code segment. The other 1s in the codeword label the other segments which can spawn the additional domain walls, thereby satisfying all long-range parity checks. The minimum number of additional domain walls is $d' - 1$, where d' is the distance of the outer code. By using a good cLDPC code as the base code, expansion properties guarantee that this domain-wall splitting scales with the size of the base code.

The HGP will produce two types of horizontal and vertical long-range boundaries: an X-type and a Z-type. For every long-range edge connecting a node and a check, the graph product will produce a long-range edge connecting a (node, node) $\rightarrow$ qubit to a (check, node) $\rightarrow$ Z-check or a (node, check) $\rightarrow$ X-check to a (check, check) $\rightarrow$ qubit. We denote an excitation of a $X(Z)$-check as an e (m) anyon. Using these conventions, we can now analyze anyon transport through the long-range boundaries. Suppose we try and move an e particle through an X-type long-range boundary (by growing its "error" string of Z's). The combination of the original and newly emerging strings must overlap on an even number of sites with each long-range X-check. This constraint is satisfied precisely by the codewords generated by G. If the code distance of the outer code is d', then the e must split into at least $d' - 1$ additional e particles. Now if we try to move an e particle through a Z-type long-range boundary, we can simply multiply the error string by long-range Z-checks, which will extend the support of this error to additional surface-code patches given by H. Growing the error strings in these other patches will create additional e particles. The rules for m particles follow analogously by switching the roles of X and Z. For each surface-code patch labeled (x, y) with the origin at the upper-left, we can arrange the rough (e absorbing), smooth (m absorbing), and long-range boundaries as depicted in Fig. 14. Rough boundaries are present at the bottom and smooth boundaries on the left. The top and right boundaries are the long-range boundaries. The anyon transport rules through the long-range boundaries can now be summarized as follows:

(i) e anyons tunnel through horizontal (vertical) boundaries according to G (H).

(ii) m anyons tunnel through horizontal (vertical) boundaries according to H (G).

So the tunneling of e (m) anyons through horizontal (vertical) boundaries is analogous to that of domain walls in the classical parent code: the codewords generated by G label the y (x) coordinates of surface-code patches where additional anyons can appear. The tunneling rules in the other directions are analogous but using the dual codewords generated by H. Because e and m anyons behave differently through long-range boundaries ($G \neq H$ in general), we have lost the usual $e \leftrightarrow m$ duality that is present in the ordinary surface code. However, if we use a *self-dual* code where $G \simeq H$ (e.g., [8,4,4] extended Hamming), then this duality is restored.

We can use the above tunneling rules to construct our logical operators. We choose the standard form of G ($\mathbb{1}_{k'}$ on the left) as a canonical basis for our logical operators like in (C1). Starting on each surface-code patch ($1 \leqslant x \leqslant k', 1 \leqslant y \leqslant k'$) in the upper-left corner, the X (Z) -type logical operators are horizontal (vertical) lines spanning the surface-code patches given by the $x(y)$th row of G with the other coordinate fixed. The X (Z) -type logical strings can be interpreted as dragging a single m (e) from a smooth (rough) boundary where they are condensed, transporting it to the long-range boundary on the opposite side, and then transporting all tunneled anyons across the new surface-code patches and absorbing them at opposing smooth (rough) boundaries. Using this procedure, we successfully construct X and Z logical operators for all $K = k'^2$ logical qubits. Since G is in standard form, these logical operators only intersect once inside the patches in the upper-left $k' \times k'$ corner.

2. Anyon confinement and single-shot decoding

The presence of syndrome measurement errors is detrimental for surface code decoding, often lowering the error threshold by an order of magnitude. The intuitive reason is that error strings are only detectable at their end points, and so if both end points have a syndrome measurement error, then that string becomes undetectable. The usual scheme to account for syndrome measurement errors is to perform multiple rounds of syndrome measurements and use the global space-time history for decoding. However, the number of measurement rounds per QEC cycle will scale with the system size [44]. If a decoder is able to account for these measurement errors with a only a small overhead, then we say this decoder is capable of *single-shot* correction. For stabilizer codes, the relation between confinement and single-shot ability has been well established [117,125]. Confinement implies that enacting a logical operation via local moves will

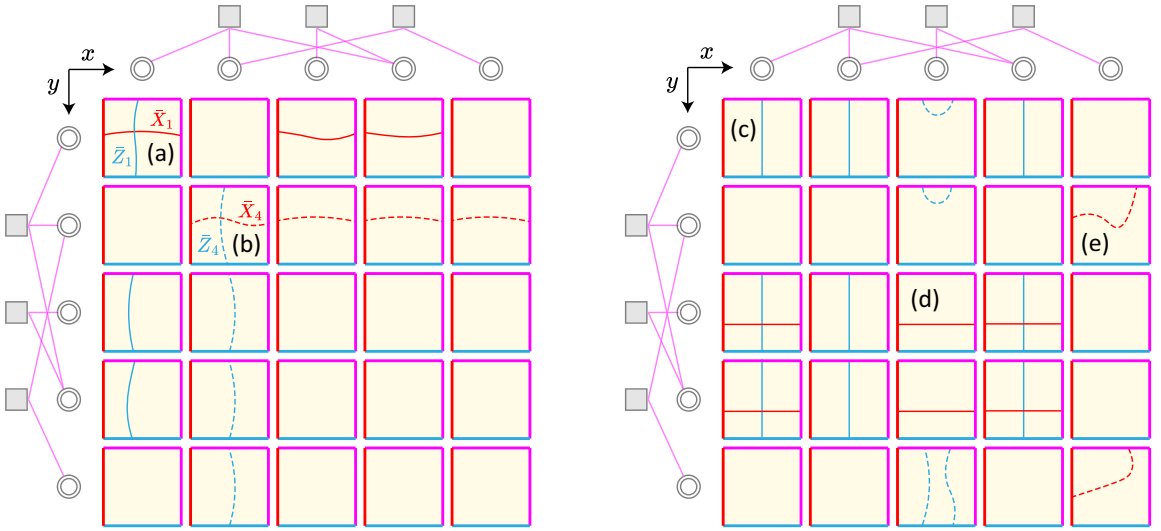

FIG. 14. Some logical (left diagram) and stabilizer (right diagram) operators are depicted for a LRESC with a parent [5,2,3] outer code (C1). (a) The $\bar{X}_1$ and $\bar{Z}_1$ logical operators are constructed using the codeword $10110 \in G$. (b) The $\bar{X}_4$ and $\bar{Z}_4$ logical operators are constructed using the codeword $01111 \in G$. (c) A Z-type stabilizer is constructed using $x : 11010 \in H$ and $y : 10110 \in G$. (d) An X-type stabilizer is constructed using $x : 10110 \in G$ and $y : 00110 \in H$. (e) An X-type stabilizer is constructed from a "contractible loop" through the long-range boundaries according to $y : 01001 \in H$. Logical operators may be deformed through the long-range boundaries by multiplying appropriate stabilizers.

necessarily violate an increasing amount of stabilizers, and so even if a few measurements are faulty, there still exists a sufficient number of violated stabilizers to undo most of the error such that any residual error remains controlled over subsequent QEC cycles.

The Tanner graphs of good cLDPC codes are typically constructed from bipartite expander graphs. Expander graphs have the property that the boundary of a small subset of vertices scales proportionally to the size of that subset. In particular, we say that a (regular) bipartite graph $G = (B \cup C, E)$ of size $\{|B|, |C|\} = \{n, m\}$ and degrees Δ_B, Δ_C is (γ, δ) left-expanding if for any subset $S \subset B$ with volume $|S| \leqslant \gamma n$, the size of the boundary, the number of connected checks, obeys $|\partial S| \geqslant (1-\delta)\Delta_B|S|$. The definition of right-expansion follows analogously by switching the roles of B and C above. On a Tanner graph, the boundary of a subset of nodes is an upper bound on the number of violated parity checks for an error supported on that subset. For Tanner graphs that are left-expanding with $\delta < 1/2$, one can show a linear code distance $d \geqslant \gamma n$ by counting the number of *unique* neighbors, a lower bound on the syndrome weight, of small subsets of nodes. One can further show that the syndrome weight $|\mathbf{s}| \geqslant (1-2\delta)\Delta_B|\mathbf{e}|$ for any error with weight $|\mathbf{e}| \leqslant \gamma n$, with $\gamma n < d$ [126]. In other words, a cLDPC code is guaranteed to exhibit *linear* confinement if the underlying Tanner graph exhibits sufficient left-expansion. As a consequence, effecting a logical error via local bit flips, or equivalently moving a domain-wall excitation, will necessarily violate a growing number of checks in the process. Reformulating the parity checks as multispin interactions in a classical Hamiltonian, we can reinterpret the previous statement as the existence of macroscopic energy barriers between different ground states.

The notion of confinement for cLDPC codes has been generalized to HGP codes [127,128]. (In these references, confinement is referred to as robustness.) We summarize the relevant results as follows. For HGP codes to achieve $D = \Theta(\sqrt{N})$, we require the Tanner graph of the parent cLDPC codes to be both left- *and* right-expanding with $\delta < 1/2$ so that both the distance and transpose distance are linear in the system size. In quantum codes, due to degeneracy, there are many errors related by stabilizer elements that produce the same error syndrome. It suffices to examine the so-called "reduced weight" of a given error, which is defined as the minimum Hamming weight over its stabilizer group orbit. If the parent expansion satisfies $\delta < 1/6$, then it has been shown that the syndrome weight obeys $|\mathbf{s}| \geqslant \frac{1}{3}|\mathbf{e}|_{\text{red}}$ for errors in the HGP code with reduced weight $|\mathbf{e}|_{\text{red}} \leqslant \min(\gamma n, \gamma m)$ with $\gamma \min(n, m) \leqslant \min(d, d^{\mathsf{T}})$ [60]. For a HGP code to provably exhibit linear confinement, its parent code requires a larger expansion ($\delta < 1/6$) than what is needed for the classical analogue ($\delta < 1/2$).

Finding explicit constructions of bipartite graphs with the necessary expansion parameters above is often difficult. Fortunately, a random regular bipartite graph with degrees $\Delta_B, \Delta_C > 1/\delta$ exhibits $(\gamma = \Omega(1), \delta)$ expansion with high probability [120]. We also note that in practice, one can often get away with smaller expansions compared to the theoretical guarantees [129].

To construct the parent code of a LRESC, we begin with a good cLDPC code and concatenate a repetition code of variable length c. If the Tanner graph of the outer cLDPC code exhibits $\delta < 1/6$ expansion, then we know that $|\mathbf{s}| \geqslant \frac{2}{3}\Delta_B|\mathbf{e}|$. The concatenation with a repetition code simply decreases the confinement to $|\mathbf{s}| \geqslant \frac{2}{3c}\Delta_B|\mathbf{e}|$, since the structure of the outer code is unchanged but each physical outer bit can now host c inner bits. An analogy holds for the associated LRESC due to the product structure of the HGP. The long-range checks in both horizontal and vertical directions mimic those of the

nonconcatenated HGP code, but now the surface codes can host an additional $\approx 2c^2$ physical qubits. Thus, the confinement in the LRESC can be quantified as $|\mathbf{s}| \geqslant \frac{1}{6c^2}|\mathbf{e}|_{\text{red}}$. Let us now analyze the scaling of confinement with the number of long-range interactions. Let $[n', \Theta(n'), \Theta(n')]$ be the parameters of the outer cLDPC code constructed using the expander graph methods previously mentioned; this code will necessarily contain $\Theta(n')$ long-range interactions. After concatenating with a length-c repetition code, the parameters become $[n'c, \Theta(n'), \Theta(n'c)]$. The confinement of this concatenated code scales as $|\mathbf{s}| = \Omega(n'/n \cdot |\mathbf{e}|)$. Suppose the number of long-range interactions scales as $n' = n^b$ for some $0 \leqslant b < 1$. Then small errors of weight $|\mathbf{e}| < n^{1-b}$ have no confinement because they can be chosen to live on a single repetition-code segment. For large errors with weight $|\mathbf{e}| = \Omega(n)$, the confinement scales as $|\mathbf{s}| = \Omega(n^b)$. Taking the HGP of this code with itself yields a LRESC with $[\![\Theta(n'^2c^2), \Theta(n'^2), \Theta(n'c)]\!]$ code parameters, $|\mathbf{s}| = \Omega(N^{b-1} \cdot |\mathbf{e}|_{\text{red}})$ confinement, and $O(n'\sqrt{N})$ long-range interactions. For a $b > 0$ scaling of the long-range interactions in the parent code, the LRESC satisfies the "good confinement" definition of [125] and is provably single-shot decodable under adversarial noise. A sustainable threshold under local stochastic noise has been proven for $b = 1$ (linear confinement and fully nonlocal limit), but it remains an open problem as to whether this threshold can exist for $b < 1$, though numerical evidence suggests an affirmative for certain families of 3D homological product codes [125]. Because LRESCs can systematically vary their density of long-range interactions, they provide tunable qLDPC codes to numerically benchmark sustainable thresholds for $0 < b < 1$. We leave such studies to future work.

We also emphasize that in practice, the existence of a threshold may not be as important when dealing with finite-size overheads, as supported by the numerical simulations in Fig. 4.

APPENDIX D: LOGICAL GATE INHERITANCE

In this Appendix, we elaborate on how a HGP code can inherit certain logical gates from its parent classical codes. In the parent codes, we show how non-fault-tolerant operations can be made increasingly transversal upon concatenating with a repetition code. We then demonstrate that this transversality is inherited by the associated LRESC, using the parent $[3(c), 2, 2(c)]$ code as a guiding example.

1. Linear transformations on the codespace

Suppose we have a classical linear code with parity-check matrix $H \in \mathbb{F}_2^{m\times n}$ and generator matrix $G \in \mathbb{F}_2^{k\times n}$. We examine transformations on the code of the form

$$H \longrightarrow HU, \quad G \longrightarrow GU, \tag{D1}$$

where $U \in \mathbb{F}_2^{n\times n}$ is an orthogonal matrix obeying $U^{\mathsf{T}}U = UU^{\mathsf{T}} = \mathbb{1}$. The columns of U represent *linear* transformations on the columns (physical bits) of H and G. In particular, since U is invertible, we can always decompose this linear transformation into a series of elementary ones corresponding to adding (A) and swapping (S) columns. Each column swap corresponds to a physical SWAP, while each column addition can be implemented by a CNOT gate on the physical bits:

$$\text{CNOT}_{i\to j} \Rightarrow \begin{matrix} H_{\star i} \longrightarrow H_{\star i} + H_{\star j} \\ G_{\star i} \longrightarrow G_{\star i} + G_{\star j}, \end{matrix} \tag{D2}$$

where $H_{\star i}$ denotes the ith column of H. For U to be a valid transformation on the codespace, we require the row space of G [i.e., $\ker(H)$] to remain invariant, which is satisfied if and only if

$$GU = VG \tag{D3}$$

for some invertible $V \in \mathbb{F}_2^{m\times m}$. Since U is orthogonal, we have $(HU)(GU)^{\mathsf{T}} = HUU^{\mathsf{T}}G^{\mathsf{T}} = HG^{\mathsf{T}} = 0$. At the same time, $H(GU)^{\mathsf{T}} = H(VG)^{\mathsf{T}} = HG^{\mathsf{T}}V^{\mathsf{T}} = 0$. Hence $\ker(H) = \ker(HU)$, which implies that H and HU are row-equivalent, i.e., there exists some invertible $W \in \mathbb{F}_2^{m\times m}$ such that

$$HU = WH. \tag{D4}$$

As an example, let us take the 3-bit parity code with

$$H = \begin{pmatrix} 1 & 1 & 1 \end{pmatrix}, \quad G = \begin{pmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \end{pmatrix}. \tag{D5}$$

A logical CNOT gate between the first logical qubit (first row of G, control) and the second logical qubit (second row of G, target) results in the transformation

$$g_1 \longrightarrow g_1 + g_2, \tag{D6a}$$

$$g_2 \longrightarrow g_2, \tag{D6b}$$

which becomes $101 \to 110$ and $011 \to 011$ explicitly for the 3-bit parity code. This logical operation can be achieved by swapping the second and third bits (columns of G) given by the permutation matrix

$$U = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{pmatrix}. \tag{D7}$$

Similarly, the logical CNOT with control and target roles reversed is achieved by swapping bits 1 and 3. The column swaps results in the desired transformation (D1) where U is a permutation, and thereby orthogonal, matrix. The logical CNOTs correspond to row addition on G, and so (D3) is also satisfied. Because H is permutation-symmetric, we have $W = \mathbb{1}$ in (D4).

2. Segment-transversality with repetition

We now show how concatenation with a repetition code maintains the same circuit depth to perform logical operations. Intuitively, this is because the repetition codes simply clone the value of the physical bits of the outer code. Any operations performed on the outer bits now become *parallel* (transversal) operations among the repetition code segments. Since the code distance is increased in the process, this concatenation will increase the fault tolerance of the original gadgets.

Suppose the $m' \times n'$ parity-check matrix of the outer code is H. Then the parity-check matrix $H^{(c)}$ of the concatenated code can be written as

$$H^{(c)} = \begin{pmatrix} H \otimes \mathbf{v} \\ \mathbb{1}_{n'} \otimes H_{\text{rep}}^{(c)} \end{pmatrix}, \tag{D8}$$

where $\otimes$ denotes the Kronecker product, $\mathbf{v} = (1\ 0\ 0\ \ldots)$ is a vector of length c with 1 in the first entry and 0 elsewhere, and $H_{\text{rep}}^{(c)}$ is the parity-check matrix of the 1D repetition code of length c with dimensions $(c-1)\times c$, e.g.,

$$H_{\text{rep}}^{(3)} = \begin{pmatrix} 1 & 1 & 0 \\ 0 & 1 & 1 \end{pmatrix}. \tag{D9}$$

The generator matrix $G^{(c)}$ takes the form

$$G^{(c)} = G \otimes \mathbf{1}_{1\times c}, \tag{D10}$$

where $\mathbf{1}_{1\times c} = (111\ldots)$ is the nonzero codeword of the repetition code of length c. The orthogonal column transformation (D1) on the concatenated code takes the transversal form

$$U^{(c)} = U \otimes \mathbb{1}_c, \tag{D11}$$

where the Kronecker product can be physically interpreted as performing U transversally between the repetition code segments; we accordingly describe this physical operation as *segment-transversal*. It is straightforward to verify that the right-action of (D11) on (D8) and (D10) reproduces the concatenated analogue of (D4) and (D3), respectively, with

$$V^{(c)} = V, \tag{D12a}$$

$$W^{(c)} = \text{diag}(W,\ U \otimes \mathbb{1}_{c-1}). \tag{D12b}$$

Since $V^{(c)} = V$, we conclude that the right-action of $U^{(c)}$ (D11) enacts the same logical transformation as the original U.

3. Patch-transversality in LRESCs

Suppose now that we use the previous concatenated code as the parent code in the hypergraph product. Recall that the parity-check matrices for a HGP code with parent code $H^{(c)}$ take the form

$$H_X = (H^{(c)} \otimes \mathbb{1}_n \ \mid\ \mathbb{1}_m \otimes H^{(c)\mathsf{T}}), \tag{D13a}$$

$$H_Z = (\mathbb{1}_n \otimes H^{(c)} \ \mid\ H^{(c)\mathsf{T}} \otimes \mathbb{1}_m). \tag{D13b}$$

For simplicity, we will focus on codewords induced by $G^{(c)}$ on the primary lattice (node-node qubits); the analysis of the transpose codewords on the secondary lattice (check-check qubits) follows analogously. Mirroring the structure of the parity checks, we can construct generator matrices for the $K = k^2$ logical qubits as follows:

$$G_Z = (G^{(c)} \otimes \mathbb{1}_n \ \mid\ 0), \tag{D14a}$$

$$G_X = (\mathbb{1}_n \otimes G^{(c)} \ \mid\ 0). \tag{D14b}$$

It is easy to verify that $H_X G_Z^{\mathsf{T}} = H_Z G_X^{\mathsf{T}} = 0$. For quantum CSS codes, we can choose either X-type or Z-type logical operators to inherit our classical transformations. Suppose we want to enact our desired logical transformation on the logical $\bar{Z}$ operators (D14a). Define the following right-action on H_X:

$$U_X^{(\bar{Z})} = \text{diag}(U^{(c)} \otimes \mathbb{1}_n,\ W^{(c)} \otimes \mathbb{1}_m), \tag{D15}$$

which physically corresponds to applying $U^{(c)}$ simultaneously to all primary rows and $W^{(c)}$ to all secondary rows in the 2D layout of Fig. 2. Examining the structure of $U^{(c)}$ (D11) and $W^{(c)}$ (D12b), we see that the above operation can be interpreted as *transversal* operations among surface-code patches given by the hypergraph product of the repetition code segments; we accordingly describe this physical operation as *patch-transversal*. For ease of notation, we will henceforth drop the superscript $(\bar{Z})$; unless stated otherwise, U_X corresponds to the choice (D15). Note that the physical action of U_X does *not* transform H_Z in the same way. Nonetheless, we can easily compute the corresponding U_Z by decomposing U_X into elementary matrices:

$$U_X = S_6 A_5 S_4 A_3 \cdots. \tag{D16}$$

The swap operations S correspond to physical SWAP gates and so remain the same for U_Z. However, the addition operations become transposed because CNOT reverses the roles of control and target for $X \leftrightarrow Z$. So the corresponding U_Z is given by

$$\begin{aligned} U_Z^{(\bar{Z})} &= S_6 A_5^{\mathsf{T}} S_4 A_3^{\mathsf{T}} \cdots = \left(U_X^{-1}\right)^{\mathsf{T}} \equiv U_X^{-\mathsf{T}} \\ &= \text{diag}(U^{(c)} \otimes \mathbb{1}_n,\ (W^{(c)})^{-\mathsf{T}} \otimes \mathbb{1}_m), \end{aligned} \tag{D17}$$

where we have used the fact that $S^2 = A^2 = \mathbb{1}$ over $\mathbb{F}_2$ in the first line and $U^{\mathsf{T}} = U^{-1}$ in the second line. The orthogonality condition $H_X H_Z^{\mathsf{T}} \to H_X U_X U_Z^{\mathsf{T}} H_Z^{\mathsf{T}} = H_X U_X U_X^{-1} H_Z^{\mathsf{T}} = H_X H_Z^{\mathsf{T}} = 0$ is thus preserved, which is expected since we know that physical unitary operations preserve commutativity.

To ensure that the stabilizer group remains invariant, it suffices to verify that $(H_X U_X) H_Z^{\mathsf{T}} = 0$. Acting U_X (D15) to the right of H_X (D13a) gives

$$\begin{aligned} H_X U_X &= (H^{(c)} U^{(c)} \otimes \mathbb{1}_n \ \mid\ W^{(c)} \otimes H^{(c)\mathsf{T}}) \\ &= (W^{(c)} H^{(c)} \otimes \mathbb{1}_n \ \mid\ W^{(c)} \otimes H^{(c)\mathsf{T}}) \\ &= (W^{(c)} \otimes \mathbb{1}_n)\, H_X \\ &\equiv W_X H_X, \end{aligned} \tag{D18}$$

from which the orthogonality condition $(H_X U_X) H_Z^{\mathsf{T}} = W_X H_X H_Z^{\mathsf{T}} = 0$ follows. Since W_X is invertible, we conclude that $H_X U_X$ is row-equivalent to the original H_X. A similar calculation also shows row-equivalence between $H_Z U_Z$ and H_Z. Thus, the patch-transversal operation corresponding to (D15) and (D17) preserves the stabilizer group.

The action on the codespace is given by

$$G_Z U_Z = (G^{(c)} U^{(c)} \otimes \mathbb{1}_n \ \mid\ 0) = (V G^{(c)} \otimes \mathbb{1}_n \ \mid\ 0), \tag{D19a}$$

$$G_X U_X = (U^{(c)} \otimes G^{(c)} \ \mid\ 0), \tag{D19b}$$

where in the second line we have used (D12a) and (D3). We see that the patch-transversal operation enacts parallel logical operations along the $n' = n/c$ columns of surface-code patches, transforming the logical $\bar{Z}$ operators in the same manner as in the classical code (D3). We can also enact patch-transversal operations between columns of surface-code patches, accordingly transforming the logical $\bar{X}$ operators, using the choice

$$U_X^{(\bar{X})} = \text{diag}(\mathbb{1}_n \otimes U^{(c)},\ \mathbb{1}_m \otimes (W^{(c)})^{-\mathsf{T}}), \tag{D20a}$$

$$U_Z^{(\bar{X})} = \text{diag}(\mathbb{1}_n \otimes U^{(c)},\ \mathbb{1}_m \otimes W^{(c)}). \tag{D20b}$$

The analogous operations for the transpose code follow similarly by switching the roles of the primary and secondary lattices [left and right sides of (D13) and (D14)]. In general,

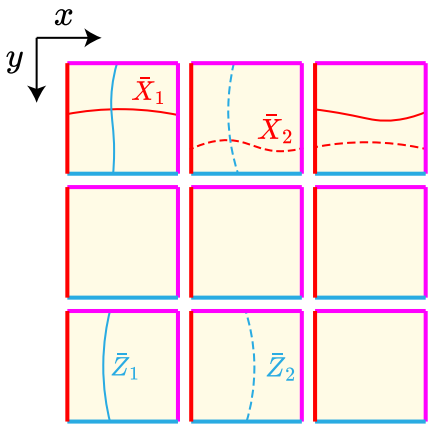

FIG. 15. The layout of the local surface-code patches of the [3,2,2] LRESC family is illustrated. The patches can be labeled by coordinates (x, y), where $x, y = 1, 2, 3$. The logical $\bar{X}$ and $\bar{Z}$ operators for the first two logical qubits are also shown.

from the form of (D20), we see that both codewords and transpose codewords are transformed simultaneously. Another thing to check is whether the complementary transformation on the other type of logical (D19b) is the desired one.

Now let us see how the 3-bit parity code fits into the above machinery. Its associated LRESC family has four logical qubits living among nine surface-code patches arranged in a 3×3 layout in the manner of Fig. 15. In the base [3,2,2] code, the U that implements the logical $\overline{\text{CNOT}}_{1\to 2}$ is given by (D7). In the LRESC, the corresponding transformation (D20) involves exchanging the $x = 2$ column of patches with those of $x = 3$. We will now show that this transformation enacts $\overline{\text{CNOT}}_{1\to 2} \cdot \overline{\text{CNOT}}_{3\to 4}$, focusing on $\overline{\text{CNOT}}_{1\to 2}$ since the analysis of $\overline{\text{CNOT}}_{3\to 4}$ follows identically by examining the second $(y = 2)$ row instead of the first $(y = 1)$. It is easy to see that this column swap maps $\bar{X}_1 \to \bar{X}_1\bar{X}_2$ while keeping $\bar{X}_2$ and $\bar{Z}_1$ unchanged, up to stabilizer equivalence. To complete the (operator) CNOT truth table, it suffices to verify that $\bar{Z}_2 \to \bar{Z}_1\bar{Z}_2$. Notice that $\bar{Z}_2$ gets mapped from the second $(x = 2)$ to the third column $(x = 3)$; denote the transformed operator as $\bar{Z}_2'$. Using the tunneling rules of Appendix C 1, we can construct a Z-type stabilizer element consisting of vertical strings living in patches indexed by $h_1 = (1\ 1\ 1)$ for the x coordinate and $g_1 = (1\ 0\ 1)$ for the y coordinate. This Z-stabilizer element is precisely given by

$$S_Z = \bar{Z}_1\bar{Z}_2\bar{Z}_2'. \tag{D21}$$

We thus verify that $\bar{Z}_2' \simeq \bar{Z}_2' S_Z = \bar{Z}_1\bar{Z}_2$ for the logical $\overline{\text{CNOT}}_{1\to 2}$, completing the truth table.

For larger codes in the Hadamard code family, we will in general lose the permutation symmetry of H in the parent code, leading to a nontrivial W in (D4). As a consequence, from (D12b) and (D15), we see that additional transversal gates may need to be applied between the long-range boundaries of the associated LRESCs. For suitably small base codes and presentations of H, the additional SWAP-CNOT gadgets may still be advantageous despite the added gate costs.

[1] D. Gottesman, Stabilizer codes and quantum error correction, arXiv:quant-ph/9705052.

[2] A. R. Calderbank and P. W. Shor, Good quantum error-correcting codes exist, Phys. Rev. A **54**, 1098 (1996).

[3] A. Steane, Multiple-particle interference and quantum error correction, Proc. R. Soc. London, Ser. A **452**, 2551 (1996).

[4] A.Yu. Kitaev, Fault-tolerant quantum computation by anyons, Ann. Phys. **303**, 2 (2003).

[5] S. B. Bravyi and A. Yu. Kitaev, Quantum codes on a lattice with boundary, arXiv:quant-ph/9811052.

[6] 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).

[7] D. Bluvstein, H. Levine, G. Semeghini, T. T. Wang, S. Ebadi, M. Kalinowski, A. Keesling, N. Maskara, H. Pichler, M. Greiner *et al.*, A quantum processor based on coherent transport of entangled atom arrays, Nature (London) **604**, 451 (2022).

[8] Google Quantum AI, Suppressing quantum errors by scaling a surface code logical qubit, Nature (London) **614**, 676 (2023).

[9] D. Bluvstein, S. J. Evered, A. A. Geim, S. H. Li, H. Zhou, T. Manovitz, S. Ebadi, M. Cain, M. Kalinowski, D. Hangleiter, J. Pablo Bonilla Ataides, N. Maskara, I. Cong, X. Gao, P. Sales Rodriguez, T. Karolyshyn, G. Semeghini, M. J. Gullans, M. Greiner, V. Vuletić *et al.*, Logical quantum processor based on reconfigurable atom arrays, Nature (London) **626**, 58 (2023).

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

[11] M. Saffman, T. G. Walker, and K. Mølmer, Quantum information with Rydberg atoms, Rev. Mod. Phys. **82**, 2313 (2010).

[12] M Saffman, Quantum computing with atomic qubits and Rydberg interactions: Progress and challenges, J. Phys. B **49**, 202001 (2016).

[13] A. M. Kaufman and K.-K. Ni, Quantum science with optical tweezer arrays of ultracold atoms and molecules, Nat. Phys. **17**, 1324 (2021).

[14] Y. Wu, S. Kolkowitz, S. Puri, and J. D. Thompson, Erasure conversion for fault-tolerant quantum computing in alkaline earth Rydberg atom arrays, Nat. Commun. **13**, 4657 (2022).

[15] I. Cong, H. Levine, A. Keesling, D. Bluvstein, S.-T. Wang, and M. D. Lukin, Hardware-efficient, fault-tolerant quantum computation with Rydberg atoms, Phys. Rev. X **12**, 021049 (2022).

[16] S. J. Evered, D. Bluvstein, M. Kalinowski, S. Ebadi, T. Manovitz, H. Zhou, S. H. Li, A. A. Geim, T. T. Wang, N. Maskara *et al.*, High-fidelity parallel entangling gates on a neutral atom quantum computer, Nature **622**, 268 (2023).

[17] S. Ma, G. Liu, P. Peng, B. Zhang, S. Jandura, J. Claes, A. P. Burgers, G. Pupillo, S. Puri, and J. D. Thompson, High-fidelity gates with mid-circuit erasure conversion in a metastable neutral atom qubit, Nature **622**, 279 (2023).

[18] J. I. Cirac and P. Zoller, Quantum computations with cold trapped ions, Phys. Rev. Lett. **74**, 4091 (1995).

[19] C. D. Bruzewicz, J. Chiaverini, R. McConnell, and J. M. Sage, Trapped-ion quantum computing: Progress and challenges, Appl. Phys. Rev. **6**, 021314 (2019).

[20] K. Kim, M.-S. Chang, S. Korenblit, R. Islam, E. Edwards, J. Freericks, G.-D. Lin, L.-M. Duan, and C. Monroe, Quantum simulation of frustrated ising spins with trapped ions, Nature (London) **465**, 590 (2010).

[21] J. W. Britton, B. C. Sawyer, A. C. Keith, C.-C. Joseph Wang, J. K. Freericks, H. Uys, M. J. Biercuk, and J. J. Bollinger, Engineered two-dimensional ising interactions in a trapped-ion quantum simulator with hundreds of spins, Nature (London) **484**, 489 (2012).

[22] J. T. Barreiro, M. Müller, P. Schindler, D. Nigg, T. Monz, M. Chwalla, M. Hennrich, C. F. Roos, P. Zoller, and R. Blatt, An open-system quantum simulator with trapped ions, Nature (London) **470**, 486 (2011).

[23] J. M. Martinis, S. Nam, J. Aumentado, and C. Urbina, Rabi oscillations in a large Josephson-Junction qubit, Phys. Rev. Lett. **89**, 117901 (2002).

[24] Y. Nakamura, C. D. Chen, and J. S. Tsai, Spectroscopy of energy-level splitting between two macroscopic quantum states of charge coherently superposed by Josephson coupling, Phys. Rev. Lett. **79**, 2328 (1997).

[25] P. Krantz, M. Kjaergaard, F. Yan, T. P. Orlando, S. Gustavsson, and W. D. Oliver, A quantum engineer's guide to superconducting qubits, Appl. Phys. Rev. **6**, 021318 (2019).

[26] P. Brooks, A. Kitaev, and J. Preskill, Protected gates for superconducting qubits, Phys. Rev. A **87**, 052306 (2013).

[27] X. Gu, A. F. Kockum, A. Miranowicz, Y.-x. Liu, and F. Nori, Microwave photonics with superconducting quantum circuits, Phys. Rep. **718–719**, 1 (2017).

[28] C. Gidney and M. Ekerå, How to factor 2048 bit RSA integers in 8 hours using 20 million noisy qubits, Quantum **5**, 433 (2021).

[29] J.-P. Tillich and G. Zemor, Quantum LDPC codes with positive rate and minimum distance proportional to the square root of the blocklength, IEEE Trans. Inf. Theor. **60**, 1193 (2014).

[30] M. B. Hastings, J. Haah, and R. O'Donnell, Fiber bundle codes: Breaking the $\sqrt{n}$ polylog(n) barrier for quantum ldpc codes, in *Proceedings of the 53rd Annual ACM SIGACT Symposium on Theory of Computing*, STOC 2021 (Association for Computing Machinery, New York, 2021), pp. 1276–1288.

[31] N. P. Breuckmann and J. N. Eberhardt, Balanced product quantum codes, IEEE Trans. Inf. Theor. **67**, 6653 (2021).

[32] P. Panteleev and G. Kalachev, Quantum LDPC codes with almost linear minimum distance, IEEE Trans. Inf. Theor. **68**, 213 (2022).

[33] P. Panteleev and G. Kalachev, Asymptotically good quantum and locally testable classical LDPC codes, in *Proceedings of the 54th Annual ACM SIGACT Symposium on Theory of Computing*, STOC 2022 (Association for Computing Machinery, New York, 2022), pp. 375-388.

[34] A. Leverrier and G. Zémor, Quantum tanner codes, arXiv:2202.13641.

[35] I. Dinur, M.-H. Hsieh, T.-C. Lin, and T. Vidick, Good quantum LDPC codes with linear time decoders, arXiv:2206.07750.

[36] S. Bravyi, D. Poulin, and B. Terhal, Tradeoffs for reliable quantum information storage in 2D systems, Phys. Rev. Lett. **104**, 050503 (2010).

[37] N. Baspin and A. Krishna, Quantifying nonlocality: How outperforming local quantum codes is expensive, Phys. Rev. Lett. **129**, 050505 (2022).

[38] N. Delfosse, M. E. Beverland, and M. A. Tremblay, Bounds on stabilizer measurement circuits and obstructions to local implementations of quantum LDPC codes, arXiv:2109.14599.

[39] Nouédyn Baspin, V. Guruswami, A. Krishna, and R. Li, Improved rate-distance trade-offs for quantum codes with restricted connectivity, arXiv:2307.03283.

[40] C. A. Pattison, A. Krishna, and J. Preskill, Hierarchical memories: Simulating quantum LDPC codes with local gates, arXiv:2303.04798.

[41] R. Gallager, Low-density parity-check codes, IEEE Trans. Inf. Theor. **8**, 21 (1962).

[42] D. J. C. MacKay and R. M. Neal, Near Shannon limit performance of low density parity check codes, Electron. Lett. **32**, 1645 (1996).

[43] S. Bravyi and B. Terhal, A no-go theorem for a two-dimensional self-correcting quantum memory based on stabilizer codes, New J. Phys. **11**, 043029 (2009).

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

[45] M. P. C. Fossorier and S. Lin, Soft-decision decoding of linear block codes based on ordered statistics, IEEE Trans. Inf. Theor. **41**, 1379 (1995).

[46] P. Panteleev and G. Kalachev, Degenerate quantum LDPC codes with good finite length performance, Quantum **5**, 585 (2021).

[47] J. Roffe, LDPC: Python tools for low density parity check codes (2022).

[48] K.-Y. Kuo, I.-C. Chern, and C.-Y. Lai, Decoding of quantum data-syndrome codes via belief propagation, in *2021 IEEE International Symposium on Information Theory (ISIT)* (IEEE, Piscataway, NJ, 2021).

[49] O. Higgott and N. P. Breuckmann, Improved single-shot decoding of higher-dimensional hypergraph-product codes, PRX Quantum **4**, 020332 (2023).

[50] S. Huang and S. Puri, Improved noisy syndrome decoding of quantum LDPC codes with sliding window, arXiv:2311.03307.

[51] O. Higgott and C. Gidney, Sparse blossom: Correcting a million errors per core second with minimum-weight matching, arXiv:2303.15933.

[52] A. D. Leu, M. F. Gely, M. A. Weber, M. C. Smith, D. P. Nadlinger, and D. M. Lucas, Fast, high-fidelity addressed single-qubit gates using efficient composite pulse sequences, Phys. Rev. Lett. **131**, 120601 (2023).

[53] L. Ding, M. Hays, Y. Sung, B. Kannan, J. An, A. Di Paolo, A. H. Karamlou, T. M. Hazard, K. Azar, D. K. Kim *et al.*, High-fidelity, frequency-flexible two-qubit fluxonium gates with a transmon coupler, Phys. Rev. X **13**, 031035 (2023).

[54] H. Zhang, C. Ding, D. K. Weiss, Z. Huang, Y. Ma, C. Guinn, S. Sussman, S. P. Chitta, D. Chen, A. A. Houck *et al.*, Tunable inductive coupler for high fidelity gates between fluxonium qubits, arXiv:2309.05720.

[55] L. Z. Cohen, I. H. Kim, S. D. Bartlett, and B. J. Brown, Low-overhead fault-tolerant quantum computing using long-range connectivity, Sci. Adv. **8**, eabn1717 (2022).

[56] Q. Xu, J. P. B. Ataides, C. A. Pattison, N. Raveendran, D. Bluvstein, J. Wurtz, B. Vasic, M. D. Lukin, L. Jiang, and H. Zhou, Constant-overhead fault-tolerant quantum computation with reconfigurable atom arrays, arXiv:2308.08648.

[57] D. Horsman, A. G. Fowler, S. Devitt, and R. V. Meter, Surface code quantum computing by lattice surgery, New J. Phys. **14**, 123011 (2012).

[58] B. J. Brown, K. Laubscher, M. S. Kesselring, and J. R. Wootton, Poking holes and cutting corners to achieve clifford gates with the surface code, Phys. Rev. X **7**, 021029 (2017).

[59] A. Krishna and D. Poulin, Fault-tolerant gates on hypergraph product codes, Phys. Rev. X **11**, 011023 (2021).

[60] A. O. Quintavalle, P. Webster, and M. Vasmer, Partitioning qubits in hypergraph product codes to implement logical gates, Quantum **7**, 1153 (2023).

[61] S. Bravyi and R. König, Classification of topologically protected gates for local stabilizer codes, Phys. Rev. Lett. **110**, 170503 (2013).

[62] E. Knill, Fault-tolerant postselected quantum computation: Schemes, arXiv:quant-ph/0402171.

[63] S. Bravyi and A. Kitaev, Universal quantum computation with ideal clifford gates and noisy ancillas, Phys. Rev. A **71**, 022316 (2005).

[64] H. Bombín, Gauge color codes: Optimal transversal gates and gauge fixing in topological stabilizer codes, New J. Phys. **17**, 083002 (2015).

[65] A. Kubica and M. E. Beverland, Universal transversal gates with color codes: A simplified approach, Phys. Rev. A **91**, 032330 (2015).

[66] B. J. Brown, A fault-tolerant non-clifford gate for the surface code in two dimensions, Sci. Adv. **6**, eaay4929 (2020).

[67] R. M. Karp, Reducibility among combinatorial problems, *Complexity of Computer Computations*, edited by R. E. Miller, J. W. Thatcher, and J. D. Bohlinger, The IBM Research Symposia Series (Springer, Boston, MA, 1972), pp. 85–103.

[68] P. W. Shor, Fault-tolerant quantum computation, arXiv:quant-ph/9605011.

[69] A. J. Kollar, M. Fitzpatrick, and A. A. Houck, Hyperbolic lattices in circuit quantum electrodynamics, Nature (London) **571**, 45 (2019).

[70] D. Rosenberg, D. Kim, R. Das, D. Yost, S. Gustavsson, D. Hover, P. Krantz, A. Melville, L. Racz, G. O. Samach, S. J. Weber, F. Yan, J. L. Yoder, A. J. Kerman, and W. D. Oliver, 3D integrated superconducting qubits, npj Quantum Inf. **3**, 42 (2017).

[71] A. G. Fowler, D. S. Wang, C. D. Hill, T. D. Ladd, R. Van Meter, and L. C. L. Hollenberg, Surface code quantum communication, Phys. Rev. Lett. **104**, 180503 (2010).

[72] F. Rozpedek, K. Noh, Q. Xu, S. Guha, and L. Jiang, Quantum repeaters based on concatenated bosonic and discrete-variable quantum codes, npj Quantum Inf. **7**, 102 (2021).

[73] S. A. Moses *et al.*, A race track trapped-ion quantum processor, Phys. Rev. X **13**, 041052 (2023).

[74] D. Kielpinski, C. Monroe, and D. J. Wineland, Architecture for a large-scale ion-trap quantum computer, Nature (London) **417**, 709 (2002).

[75] T. P. Harty, D. T. C. Allcock, C. J. Ballance, L. Guidoni, H. A. Janacek, N. M. Linke, D. N. Stacey, and D. M. Lucas, High-fidelity preparation, gates, memory, and readout of a trapped-ion quantum bit, Phys. Rev. Lett. **113**, 220501 (2014).

[76] C. R. Clark, H. N. Tinkey, B. C. Sawyer, A. M. Meier, K. A. Burkhardt, C. M. Seck, C. M. Shappert, N. D. Guise, C. E. Volin, S. D. Fallek, H. T. Hayden, W. G. Rellergert, and K. R. Brown, High-fidelity bell-state preparation with $^{40}ca^{+}$ optical qubits, Phys. Rev. Lett. **127**, 130505 (2021).

[77] C. Ryan-Anderson *et al.*, Implementing fault-tolerant entangling gates on the five-qubit code and the color code, arXiv:2208.01863.

[78] L. Postler, S. Heuβen, I. Pogorelov, M. Rispler, T. Feldker, M. Meth, C. D. Marciniak, R. Stricker, M. Ringbauer, R. Blatt, P. Schindler, M. Müller, and T. Monz, Demonstration of fault-tolerant universal quantum gate operations, Nature (London) **605**, 675 (2022).

[79] D. G. Dalgoutte and C. D. W. Wilkinson, Thin grating couplers for integrated optics: An experimental and theoretical study, Appl. Opt. **14**, 2983 (1975).

[80] K. K. Mehta, C. D. Bruzewicz, R. McConnell, R. J. Ram, J. M. Sage, and J. Chiaverini, Integrated optical addressing of an ion qubit, Nat. Nanotechnol. **11**, 1066 (2016).

[81] R. J. Niffenegger, J. Stuart, C. Sorace-Agaskar, D. Kharas, S. Bramhavar, C. D. Bruzewicz, W. Loh, R. T. Maxson, R. McConnell, D. Reens, G. N. West, J. M. Sage, and J. Chiaverini, Integrated multi-wavelength control of an ion qubit, Nature (London) **586**, 538 (2020).

[82] K. K. Mehta, C. Zhang, M. Malinowski, T.-L. Nguyen, M. Stadler, and J. P. Home, Integrated optical multi-ion quantum logic, Nature (London) **586**, 533 (2020).

[83] J. Kwon, W. J. Setzer, M. Gehl, N. Karl, J. Van Der Wall, R. Law, D. Stick, and H. J. McGuinness, Multi-site integrated optical addressing of trapped ions, Nat Commun **15**, 3709 (2024).

[84] C. Mordini, A. R. Vasquez, Y. Motohashi, M. Müller, M. Malinowski, C. Zhang, K. K. Mehta, D. Kienzler, and J. P. Home, Multi-zone trapped-ion qubit control in an integrated photonics QCCD device, arXiv:2401.18056.

[85] F. Lindenfelser, M. Marinelli, V. Negnevitsky, S. Ragg, and J. P. Home, Cooling atomic ions with visible and infra-red light, New J. Phys. **19**, 063041 (2017).

[86] R. J. Hendricks, J. L. Sørensen, C. Champenois, M. Knoop, and M. Drewsen, Doppler cooling of calcium ions using a dipole-forbidden transition, Phys. Rev. A **77**, 021401(R) (2008).

[87] J. M. Pino, J. M. Dreiling, C. Figgatt, J. P. Gaebler, S. A. Moses, M. S. Allman, C. H. Baldwin, M. Foss-Feig, D. Hayes, K. Mayer, C. Ryan-Anderson, and B. Neyenhuis, Demonstration of the trapped-ion quantum CCD computer architecture, Nature (London) **592**, 209 (2021).

[88] K. A. Landsman, Y. Wu, P. H. Leung, D. Zhu, N. M. Linke, K. R. Brown, L. Duan, and C. Monroe, Two-qubit entangling gates within arbitrarily long chains of trapped ions, Phys. Rev. A **100**, 022332 (2019).

[89] Y. Wang, S. Crain, C. Fang, B. Zhang, S. Huang, Q. Liang, P. H. Leung, K. R. Brown, and J. Kim, High-fidelity two-qubit gates using a microelectromechanical-system-based beam steering system for individual qubit addressing, Phys. Rev. Lett. **125**, 150505 (2020).

[90] C. Monroe, R. Raussendorf, A. Ruthven, K. R. Brown, P. Maunz, L.-M. Duan, and J. Kim, Large-scale modular quantum-computer architecture with atomic memory and photonic interconnects, Phys. Rev. A **89**, 022317 (2014).

[91] L. J. Stephenson, D. P. Nadlinger, B. C. Nichol, S. An, P. Drmota, T. G. Ballance, K. Thirumalai, J. F. Goodwin, D. M. Lucas, and C. J. Ballance, High-rate, high-fidelity entanglement of qubits across an elementary quantum network, Phys. Rev. Lett. **124**, 110501 (2020).

[92] V. Krutyanskiy, M. Meraner, J. Schupp, V. Krcmarsky, H. Hainzer, and B. P. Lanyon, Light-matter entanglement over 50 km of optical fibre, npj Quantum Inf. **5**, 72 (2019).

[93] E. T. Campbell, Distributed quantum-information processing with minimal local resources, Phys. Rev. A **76**, 040302(R) (2007).

[94] A. Browaeys and T. Lahaye, Many-body physics with individually controlled Rydberg atoms, Nat. Phys. **16**, 132 (2020).

[95] P. Scholl, M. Schuler, H. J. Williams, A. A. Eberharter, D. Barredo, K.-N. Schymik, V. Lienhard, L.-P. Henry, T. C. Lang, T. Lahaye *et al.*, Quantum simulation of 2D antiferromagnets with hundreds of Rydberg atoms, Nature (London) **595**, 233 (2021).

[96] S. Ebadi, T. T. Wang, H. Levine, A. Keesling, G. Semeghini, A. Omran, D. Bluvstein, R. Samajdar, H. Pichler, W. W. Ho *et al.*, Quantum phases of matter on a 256-atom programmable quantum simulator, Nature (London) **595**, 227 (2021).

[97] A. W. Young, S. Geller, W. J. Eckner, N. Schine, S. Glancy, E. Knill, and A. M. Kaufman, An atomic boson sampler, Nature **629**, 311 (2024).

[98] R. Tao, M. Ammenwerth, F. Gyger, I. Bloch, and J. Zeiher, High-fidelity detection of large-scale atom arrays in an optical lattice, Phys. Rev. Lett. **133**, 013401 (2024).

[99] W. J. Eckner, N. Darkwah Oppong, A. Cao, A. W. Young, W. R. Milner, J. M. Robinson, J. Ye, and A. M. Kaufman, Realizing spin squeezing with Rydberg interactions in an optical clock, Nature **621**, 734 (2023).

[100] S. Trotzky, P. Cheinet, S Folling, M. Feld, U. Schnorrberger, A. M. Rey, A. Polkovnikov, E. A. Demler, M. D. Lukin, and I. Bloch, Time-resolved observation and control of superexchange interactions with ultracold atoms in optical lattices, Science **319**, 295 (2008).

[101] A. M. Kaufman, B. J. Lester, M Foss-Feig, M. L. Wall, A. M. Rey, and C. A. Regal, Entangling two transportable neutral atoms via local spin exchange, Nature (London) **527**, 208 (2015).

[102] D. Barredo, S. De Léséleuc, V. Lienhard, T. Lahaye, and A. Browaeys, An atom-by-atom assembler of defect-free arbitrary two-dimensional atomic arrays, Science **354**, 1021 (2016).

[103] M. Endres, H. Bernien, A. Keesling, H. Levine, E. R. Anschuetz, A. Krajenbrink, C. Senko, V. Vuletic, M. Greiner, and M. D. Lukin, Atom-by-atom assembly of defect-free one-dimensional cold atom arrays, Science **354**, 1024 (2016).

[104] K. Singh, C. E. Bradley, S. Anand, V. Ramesh, R. White, and H. Bernien, Mid-circuit correction of correlated phase errors using an array of spectator qubits, Science **380**, 1265 (2023).

[105] J. W. Lis, A. Senoo, W. F. McGrew, F. Rönchen, A. Jenkins, and A. M. Kaufman, Mid-circuit operations using the omg-architecture in neutral atom arrays, arXiv:2305.19266.

[106] M. A. Norcia, W. B. Cairncross, K. Barnes, P. Battaglino, A. Brown, M. O. Brown, K. Cassella, C.-A. Chen, R. Coxe, D. Crow *et al.*, Mid-circuit qubit measurement and rearrangement in a ^{171}Yb atomic array, Phys. Rev. X **13**, 041034 (2023).

[107] P. Scholl, A. L. Shaw, R. B.-S. Tsai, R. Finkelstein, J. Choi, and M. Endres, Erasure conversion in a high-fidelity Rydberg quantum simulator, Nature **622**, 273 (2023).

[108] L. Pause, T. Preuschoff, D. Schäffner, M. Schlosser, and G. Birkl, Reservoir-based deterministic loading of single-atom tweezer arrays, Phys. Rev. Res. **5**, L032009 (2023).

[109] T. Xia, M. Lichtman, K. Maller, A. W. Carr, M. J. Piotrowicz, L. Isenhower, and M. Saffman, Randomized benchmarking of single-qubit gates in a 2D array of neutral-atom qubits, Phys. Rev. Lett. **114**, 100503 (2015).

[110] Y. Wang, A. Kumar, T.-Y. Wu, and D. S. Weiss, Single-qubit gates based on targeted phase shifts in a 3D neutral atom array, Science **352**, 1562 (2016).

[111] C. Sheng, X. He, P. Xu, R. Guo, K. Wang, Z. Xiong, M. Liu, J. Wang, and M. Zhan, High-fidelity single-qubit gates on neutral atoms in a two-dimensional magic-intensity optical dipole trap array, Phys. Rev. Lett. **121**, 240501 (2018).

[112] A. Jenkins, J. W. Lis, A. Senoo, W. F. McGrew, and A. M. Kaufman, Ytterbium nuclear-spin qubits in an optical tweezer array, Phys. Rev. X **12**, 021027 (2022).

[113] S. Ma, A. P. Burgers, G. Liu, J. Wilson, B. Zhang, and J. D. Thompson, Universal gate operations on nuclear spin qubits in an optical tweezer array of ^{171}Yb atoms, Phys. Rev. X **12**, 021028 (2022).

[114] M. Suchara, A. W. Cross, and J. M. Gambetta, Leakage suppression in the toric code, in *2015 IEEE International Symposium on Information Theory (ISIT)* (IEEE, Piscataway, NJ, 2015), pp. 1119–1123.

[115] N. Berthusen and D. Gottesman, Partial syndrome measurement for hypergraph product codes, Quantum **8**, 1345 (2024).

[116] N. Delfosse and N. H. Nickerson, Almost-linear time decoding algorithm for topological codes, Quantum **5**, 595 (2021).

[117] H. Bombín, Single-shot fault-tolerant quantum error correction, Phys. Rev. X **5**, 031043 (2015).

[118] A. Kubica and M. Vasmer, Single-shot quantum error correction with the three-dimensional subsystem toric code, Nat. Commun. **13**, 6272 (2022).

[119] C. Stahl, Single-shot quantum error correction in intertwined toric codes, arXiv:2307.08118.

[120] T. Richardson and R. Urbanke, *Modern Coding Theory* (Cambridge University Press, Cambridge, 2008).

[121] J. Roffe, D. R. White, S. Burton, and E. Campbell, Decoding across the quantum low-density parity-check code landscape, Phys. Rev. Res. **2**, 043423 (2020).

[122] A. Leverrier and G. Zémor, Efficient decoding up to a constant fraction of the code length for asymptotically good quantum codes, *ACM-SIAM Symposium on Discrete Algorithms, SODA 2023* (ACM, 2022).

[123] A. Leverrier, Mapping the toric code to the rotated toric code, https://github.com/errorcorrectionzoo/eczoo_data/files/9210173/rotated.pdf (2022).

[124] A. G. Manes and J. Claes, Distance-preserving stabilizer measurements in hypergraph product codes, arXiv:2308.15520.

[125] A. O. Quintavalle, M. Vasmer, J. Roffe, and E. T. Campbell, Single-shot error correction of three-dimensional homological product codes, PRX Quantum **2**, 020340 (2021).

[126] M. Sipser and D. A. Spielman, Expander codes, IEEE Trans. Inf. Theor. **42**, 1710 (1996).

[127] A. Leverrier, J.-P. Tillich, and G. Zemor, Quantum expander codes, in *2015 IEEE 56th Annual Symposium on Foundations of Computer Science* (IEEE, Piscataway, NJ, 2015).

[128] O. Fawzi, A. Grospellier, and A. Leverrier, Constant overhead quantum fault tolerance with quantum expander codes, Commun. ACM **64**, 106 (2021).

[129] A. Grospellier and A. Krishna, Numerical study of hypergraph product codes, arXiv:1810.03681.