Universal Fault-Tolerant Quantum Computation in 2D without Getting Tied in Knots

Margarita Davydova,[1,2,3] Andreas Bauer,[4] Julio C. Magdalena de la Fuente,[5] Mark Webster,[6,7] Dominic J. Williamson,[7,8] and Benjamin J. Brown[2,9]

[1]*Walter Burke Institute for Theoretical Physics and Institute for Quantum Information and Matter, California Institute of Technology, Pasadena, California 91125, USA*
[2]*IBM Quantum, T. J. Watson Research Center, Yorktown Heights, New York 10598, USA*
[3]*Department of Physics, Massachusetts Institute of Technology, Cambridge, Massachusetts 02139, USA*
[4]*Department of Mechanical Engineering, Massachusetts Institute of Technology, Cambridge, Massachusetts 02139, USA*
[5]*Dahlem Center for Complex Quantum Systems, Freie Universität Berlin, 14195 Berlin, Germany*
[6]*Department of Physics and Astronomy, University College London, London, WC1E 6BT, United Kingdom*
[7]*School of Physics, The University of Sydney, New South Wales 2006, Australia*
[8]*IBM Quantum, IBM Almaden Research Center, San Jose, California 95120, USA*
[9]*IBM Denmark, Sundkrogsgade 11, 2100 Copenhagen, Denmark*

(Received 15 May 2025; revised 9 December 2025; accepted 26 March 2026; published 7 July 2026)

We show how to perform scalable fault-tolerant non-Clifford gates in two dimensions by introducing domain walls between the surface code and a non-Abelian topological code whose code space is stabilized by Clifford operators. We formulate a path-integral framework which provides both a macroscopic picture for different logical gates as well as a way to derive the associated microscopic circuits. We also show an equivalence between our approach and prior proposals where a 2D array of qubits reproduces the action of a transversal gate in a 3D stabilizer code over time, thus establishing a connection between 3D codes and 2D non-Abelian topological phases. We prove a threshold theorem for our protocols under local stochastic circuit noise using a just-in-time decoder to correct the non-Abelian code.

DOI: 10.1103/vrty-qs5h

Subject Areas: Quantum Information

I. INTRODUCTION

A scalable quantum computer must have a universal set of fault-tolerant quantum logic gates. These gates require a practical scheme for their execution with quantum error-correcting codes that have low-depth circuits for syndrome readout as well as an efficient decoding strategy [1]. Topological codes [2–4] are promising in this regard thanks to their local connectivity and efficient high-threshold decoding algorithms. Furthermore, their rich underlying physics makes them an excellent playground for designing new fault-tolerant logic gates [2,5,6].

In this work, we show how to perform non-Clifford logic gates in copies of the surface code by interfacing them with a topological code stabilized by Clifford operators. The code space of the latter code is the ground-state space of the type-III twisted quantum double model [7,8], which represents a *non-Abelian* topological phase [9,10]. We derive circuits for syndrome readout for this non-Abelian code, together with an efficient decoder that demonstrates a threshold. The logical information at the beginning and end of the protocol is encoded in copies of the surface code or an equivalent code in the same topological phase. A universal set of logic gates for the surface codes can then be completed using standard methods to perform Clifford gates in two dimensions [3,11–21].

Our results can be interpreted using a number of different perspectives, each with its own advantages. One point of view treats our logic gates as a code deformation between the surface code and the non-Abelian code and presents an instance of a gauging logical measurement. We discuss the code deformation and the gauging perspectives in Sec. III. More generally, we can design a multitude of non-Clifford logic gates more systematically by changing boundary and domain wall configurations in spacetime using the path-integral formalism. We present this picture in Sec. IV. Examples include logical CCZ and T gates as well as magic CZ- and T-state preparation. In Sec. V, we show how to use the path-integral approach to derive microscopic circuits for the logical protocols. Finally, our gates are closely related to prior work on two-dimensional implementations of non-Clifford gates [22,23]. There, the transversal non-Clifford gates of a three-dimensional code are turned into linear-depth protocols on a two-dimensional

Published by the American Physical Society under the terms of the Creative Commons Attribution 4.0 International license. Further distribution of this work must maintain attribution to the author(s) and the published article's title, journal citation, and DOI.

array of qubits. In Sec. VI, we show that this approach is a special instance following from our framework. Thus, both of the examples in Refs. [22,23] realize the same non-Abelian code during their intermediate steps, which reveals a new connection between 3D codes with non-Clifford transversal gates and 2D non-Abelian topological phases.

Unlike conventional stabilizer codes corresponding to Abelian phases, decoding non-Abelian anyon models has not been addressed as satisfactorily in the literature. Existing work typically includes numerical results with simplified noise models [24–26]. To the best of our knowledge, analytical results include fault tolerance of nilpotent anyon models [27] under general phenomenological noise [28] and the Ising anyon model under a phenomenological noise model that assumes perfect measurements. In this paper, we show that one can use the just-in-time decoder [22,23,29,30] to decode the non-Abelian quantum double appearing in our protocols. Unlike usual matching algorithms for Abelian topological codes, the decoder for the non-Abelian phases must apply corrections on the fly (hence the relation to the "just-in-time" concept), using only the spacetime history of the error syndromes available up to the present time. We formulate an renormalization group (RG)-based just-in-time decoding algorithm in the context of fault tolerance of a non-Abelian code and prove a threshold theorem under the local stochastic noise model in Sec. VII. We also introduce an efficient minimum-weight perfect-matching version of this decoder that we expect to perform even better (i.e., achieve a logical error rate that is exponentially suppressed with the code distance).

As a means to implement non-Clifford gates and magic states, our results offer an alternative to magic state distillation [31–34]. Non-Clifford gates and magic states are widely regarded to be the most resource-intensive components in two-dimensional quantum computing architectures if one desires to achieve extremely low logical error rates, as well as in a number of other settings [34–37].

Because of the relation to non-Abelian phases, it is also instructive to contrast our results with existing schemes for topological quantum computation by braiding anyons or holes in a non-Abelian phase [2,5,38,39]. The microscopic circuits for braiding-universal phases, such as the Fibonacci model, usually require deep syndrome extraction circuits [25,40–42]. In contrast, a round of syndrome extraction in our protocols has comparatively low depth. Furthermore, the circuits can be expressed using CNOTs, single-qubit Pauli measurements, and few-qubit non-Clifford gates only. Specific non-Clifford gates that are required depend on the choice of the microscopic realization of the protocol; some of the examples we present only require non-Clifford $T/T^\dagger$ gates; others require CCZ and $CS/CS^\dagger$ (however, we note that CCZ and $CS/CS^\dagger$ gates can always be resolved into a circuit of Clifford gates and $T/T^\dagger$ rotations, at the expense of extra overhead). Additionally, braiding-universal topological phases require sophisticated strategies for decoding and correction that have not been studied in full [26]. The nilpotency of the twisted quantum double topological phase used in our protocols allows us to design especially straightforward decoding strategies using a just-in-time decoder [22,23]. This gives our scheme a clear simplicity advantage over proposals that require more complex phases.

We can also compare our scheme to proposals that propose preparing magic states by measuring macroscopic logical operators directly [43–46]. In our work, we infer the value of high-weight logical operators using low-weight checks. This gives us a scalable way to identify and correct errors as the gate is conducted. In this context, our results can be viewed as a generalization of logic gates by lattice surgery [12,47–51], where the Clifford logical operators are measured using local operations to complete the universal gate set.

Our results offer a new perspective on universal quantum computation in two dimensions using non-Abelian topological phases. We present several approaches to designing fault-tolerant non-Clifford logical protocols and circuits, and we clarify how these methods are related to one another. This may serve as a useful starting point for the development of efficient circuits with specific architectures in mind and for benchmarking them under realistic assumptions. We hope that further development of our approach will help reduce the resource costs of scalable quantum computing.

Additionally, we argue that our approach can be extended to other non-Abelian phases and more general quantum low-density parity check codes (qLDPC). This provides a new avenue to realize universal fault-tolerant quantum computation under geometric locality restrictions that are inherited from physical architectures.

II. SKETCH OF THE IDEA

The protocols in this paper all use a non-Abelian topological phase to perform non-Clifford gates on copies of the surface or toric code. Here, we give a high-level overview of the main idea with two expository examples.

We use the *type-III twisted quantum double* (TQD), which is related to three copies of the toric code phase by a 3-cocycle twist [7,9]. More specifically, it is the twisted quantum double model associated with the type-III 3-cocycle $[\omega_{\mathrm{III}}] \in H^3[\mathbb{Z}_2 \times \mathbb{Z}_2 \times \mathbb{Z}_2, U(1)]$ given by $\omega_{\mathrm{III}}[(a,b,c),(a',b',c'),(a'',b'',c'')] = (-1)^{ab'c''}$. This model represents the same topological phase as the quantum double of D_4 [2,10,52]. We show how to use domain walls to transfer logical information encoded in toric or surface codes [2,3,11,53] into the TQD and back again to perform non-Clifford gates. While information is encoded in the twisted quantum double model, we take advantage of its twist to introduce a relative phase factor in the encoded state to implement non-Clifford gates.

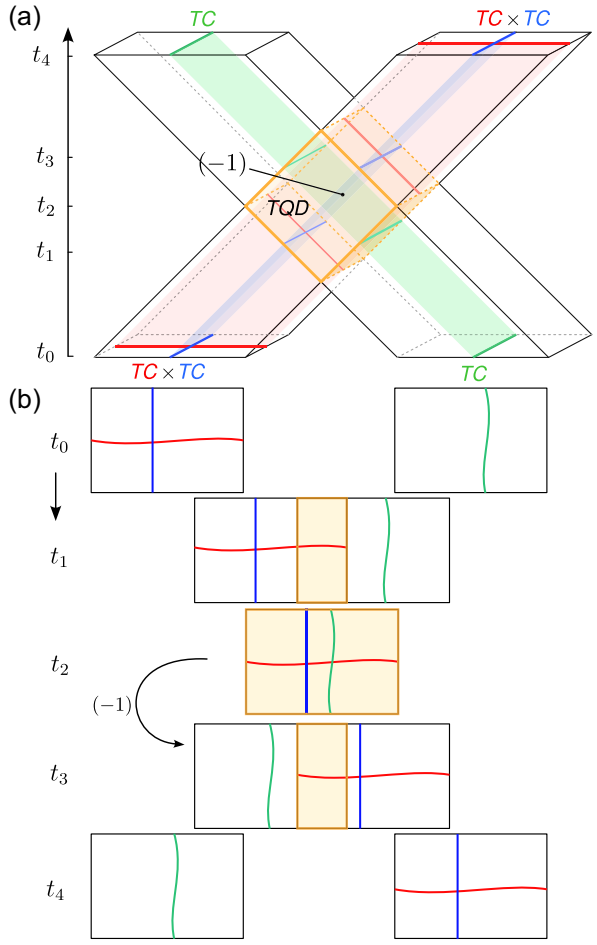


FIG. 1. (a) The spacetime picture of the protocol that fault-tolerantly implements the non-Clifford $\overline{CCZ}$ gate on three copies of the surface code in $O(d)$ time. The spacetime region containing the non-Abelian twisted quantum double phase is shaded in yellow. A particular boundary configuration of three surface codes is initialized at time t_0. Upon completion of the protocol, the state acquires a -1 phase only if all three surface codes are simultaneously in the $|1\rangle$ logical state, corresponding to $\overline{CCZ}_{rgb}$. (b) A step-by-step depiction of the protocol, which can be interpreted as "sweeping" one copy of the surface code past two others. The copies are coupled by a twist in the yellow-shaded region, where the twisted quantum double is realized.

The first expository example, shown in Fig. 1, realizes the non-Clifford unitary $\overline{CCZ}$ gate. The spacetime layout is based on that in Ref. [23]; however, we derive the action of the gate by considering the twisted quantum double as an intermediate two-dimensional phase. Figure 1(b) shows how the gate can be viewed as "sweeping" the third copy of the surface code (labeled green) past the other two (labeled red and blue). At the points in time where all three of the surface code regions overlap, we realize the twisted quantum double, which is responsible for a relative phase over the course of the operation. Specifically, if and only if all three codes start in the logical-1 state, a -1 phase is acquired. As a result, the protocol applies the phase $(-1)^{rgb}$, which is the same as applying the logical $\overline{CCZ}_{rgb}$ gate.

Throughout our work, it is helpful to view our logic operations using a $(2+1)$-dimensional spacetime perspective, which we explore systematically in Sec. IV. In spacetime, the relative phase responsible for the logical action appears as a triple intersection between the membranes associated with the logical states. The logical states of the toric codes are in correspondence with continuous nontrivial loops in space, which turn into continuous membranes when viewed in spacetime. Likewise, the states in the non-Abelian phase are defined by continuous loops (membranes in spacetime) that we label with their three respective colors. However, unlike in the toric or surface codes phase, every point in spacetime where all three types of membranes intersect (we call this a "triple intersection point") acquires a -1 phase due to the 3-cocycle twist. The spacetime configuration of topological codes and their domain walls and boundaries shown in Fig. 1(a) is chosen such that, for the all-1 input logical state, there is strictly an odd number of logical membrane intersections, which gives rise to the $\overline{CCZ}$ logic gate. We explain this gate in detail in Sec. IV. In Appendix A, we give more details on the loop-sum picture.

We now turn to the second example, which is explained together with microscopic details in Sec. III. It simply projects two copies of the toric code onto the twisted quantum double model on the torus before reversing the operation, as shown in Fig. 2. The resulting operation can

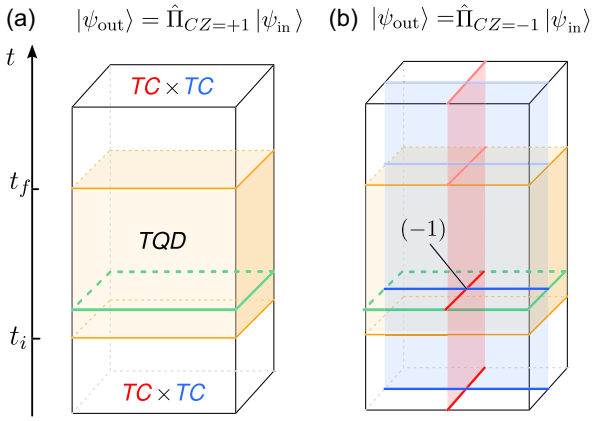


FIG. 2. Schematic spacetime diagram of the protocol that projects an input logical four-qubit state onto an eigenstate of the $\overline{CZ}_{rHbV}\overline{CZ}_{rVbH}$ operator. The protocol transitions from two copies of the toric code to the non-Abelian twisted quantum double phase and back. The spacetime region with the twisted quantum double phase is shaded in yellow. The projection occurs because, inside the twisted quantum double phase, the product of all vertex stabilizers of the green color (in one time slice) counts the number of red-blue string intersections modulo 2. If the toric codes are initialized in the logical $|++00\rangle$ state, the protocol probabilistically prepares the "CZ" magic state.

be used to prepare a magic state known as the CZ state [54,55].

The protocol is defined with two copies of the toric codes as an input, where each code encodes two logical qubits. We label the two copies red and blue and index the encoded qubits by the color and the orientation of their respective logical Pauli-X string operator, i.e., $\{rH, bV, rV, bH\}$. Then, the protocol transforms from the toric code phase to the twisted quantum double phase and back, which performs a logical measurement, projecting the input state onto an eigenstate of the logical $\overline{CZ}_{rH,bV}\overline{CZ}_{rV,bH}$ operator. Suppose that we start with the input state $|+\rangle_{rH}|+\rangle_{bV}|0\rangle_r V|0\rangle_{bH}$; then this measurement probabilistically prepares the state $|CZ\rangle_{rH,bV}|0\rangle_r V|0\rangle_{bH}$ where the CZ-magic state $|CZ\rangle$ is an equal-weight superposition of the $+1$ eigenstates of the $\overline{CZ}$ operator: $|\overline{CZ}\rangle \propto |00\rangle + |01\rangle + |10\rangle$.

As with the first example, we can use the spacetime picture to see the action of the gate. First of all, we notice that, with our choice of boundary conditions, a green membrane is supported at the 2D interface between the toric codes and the twisted quantum double phase, which gives rise to two possible nontrivial triple intersections. One is between the qubits rH and bV and the green membrane degree of freedom which we label g, while the second is between rV, bH, and g. One such intersection is shown in Fig. 2. Once the state is in the twisted phase, the product of all the green vertex stabilizers on any time slice produces the green membrane. Hence, measuring these stabilizers results in the projective measurement of the total number of triple intersections on a time slice modulo 2. This number is given by the red-blue membrane intersections in the toric code state that enters the twisted phase, due to the geometry. The green membrane insertion at each step of the twisted quantum double phase can be thought of as initializing the auxiliary degree of freedom in the $|+\rangle_g$ state, after which a nontrivial phase factor $\overline{CCZ}_{rV,bH,g}\overline{CCZ}_{rH,bV,g}$ counting the parity of triple intersections is implemented; then the extra degree of freedom is measured in the logical Pauli-X basis. The effect of this is the measurement of the logical operator $\overline{CZ}_{rH,bV}\overline{CZ}_{rV,bH}$.

Another way to understand this logical action is by noticing that the domain wall occurring at time t_i when the protocol switches from two copies of the toric code to the twisted quantum double corresponds to *gauging* a global symmetry of the red and blue copies of the toric codes by measurement. This symmetry has the logical unitary action of $\overline{CZ}_{rHbV}\overline{CZ}_{rVbH}$. Gauging by measurements results in a projection onto the eigenstate of this symmetry. The second transition at t_f is an ungauging operation that reverts the system to the two copies of the toric code [48].

An important feature of our protocols is that the logical operations implemented by a certain spacetime configuration depend solely on the topological phase of each volume, as well as the phases of spacetime defects such as boundaries, domain walls, and corners, which are sometimes referred to as "universality classes" or "superselection sectors." This can be used to derive various microscopic implementations of the same logic gates, as these phases can have many different lattice realizations. In this work, we focus on the description of the non-Abelian model as the twisted quantum double, which we use in the continuum closed-membrane picture for computing the logical action implemented by different spacetime configurations. Various microscopic implementations representing the same topological phase are discussed in Secs. V and VI. In Appendix D, we present more microscopic examples derived from gauging Clifford symmetries.

III. PREPARING A MAGIC STATE ON A TORUS

In this section, we discuss the protocol that is shown in Fig. 2. We provide a succinct microscopic explanation that is centered around stabilizer and logical operators of the quantum error-correcting code that is transformed throughout the protocol. We then explain how this protocol can be viewed as a gauging logical measurement, which is a type of code deformation. In Secs. IV and V, we introduce a comprehensive method that generalizes the example presented in this section to construct other logic-gate protocols and the associated circuits based on the topological path integral. We also show a planar version of this protocol that operates on surface as opposed to toric codes in Sec. IV.

A. Stabilizers of the twisted quantum double

In this example, we use a microscopic lattice realization of the non-Abelian topological phase that is needed for the protocol. This model shares the code space (the ground-state subspace) with a variant [8,56] of the type-III non-Abelian twisted quantum double [7]. We use this known microscopic realization as our simplest illustrative example; the novelty here is in using this model as an intermediate non-Abelian phase in a non-Clifford logical protocol. In the rest of the paper, we refer to this model simply as the "twisted quantum double," as it is the only twisted quantum double model that appears in this work. We discuss alternative microscopic realizations of the same non-Abelian phase in Secs. V and VI, which are used for alternative circuits that have the same logical action.

We start with a single copy of the (untwisted) toric code. We place qubits on the edges e of a triangular lattice. The stabilizer group is generated by star and plaquette stabilizers associated to vertices v and faces f as follows:

$$A_v = \prod_{e:\, v\in\partial e} X_e, \qquad B_f = \prod_{e\in\partial f} Z_e, \tag{1}$$

where X_e and Z_e are the standard Pauli matrices acting on a qubit at location e and ∂ is the boundary map. The code

states of the toric code are the simultaneous $+1$ eigenstates of all stabilizer operators. The logical Pauli operators of the toric code are strings of Pauli operators that run along noncontractible cycles of the manifold, on the primal lattice for Z and on the dual lattice for X. On a torus, we label them by the horizontal or vertical direction of winding, i.e., $\bar{Z}_{H/V}$ and $\bar{X}_{H/V}$. With this definition, the nontrivial commutation relations are as follows: $\bar{Z}_H\bar{X}_V = -\bar{X}_V\bar{Z}_H$ and $\bar{Z}_V\bar{X}_H = -\bar{X}_H\bar{Z}_V$.

We now introduce the noncommuting stabilizer model that realizes the TQD topological phase. Informally, this model can be understood as three copies of the toric code that are coupled via a "twist." When picturing the qubit configurations that appear in the ground state as closed-loop patterns, the twist can be understood as counting the number of triple intersections, as we discuss in Sec. IV and Appendix A. Consider three overlapping triangular sublattices, which we color red (r), green (g), and blue (b), as shown in Fig. 3. Accordingly, there is one qubit at each edge of each sublattice, marked by a dot. For the sublattice of color $c \in \{r, g, b\}$, we label the vertices as (v, c), edges as (e, c), and plaquettes as (f, c). The plaquette operators are the same as those of three individual toric codes placed each on a sublattice of an associated color. We denote them $\mathcal{B}_{f,c} = B_{f,c}$. The vertex terms are modified by the twist and become Clifford stabilizers:

$$\begin{aligned}\mathcal{A}_{v,c} &= \prod_{\partial e \ni (v,c)} X_e \prod_{i \in \langle\langle (v,c) \rangle\rangle} CZ_{i,i+1} \\ &\equiv A_{v,c} \prod_{i \in \langle\langle (v,c) \rangle\rangle} CZ_{i,i+1}, \end{aligned} \tag{2}$$

where the product of the CZ phase operators is applied to the qubits of two complementary colors belonging to a hexagon surrounding the vertex (v, c). Both types of stabilizers are depicted in Fig. 3 for the blue sublattice, and those of the red and green colors are defined analogously.

In noncommuting stabilizer codes [57–59], similarly to the usual stabilizer codes, the code states can be defined as the $+1$ eigenstates of all the star (vertex) and plaquette stabilizers. In other words, the corresponding Hamiltonian

$$\mathcal{H} = -\sum_{v,c} \mathcal{A}_{v,c} - \sum_{f,c} \mathcal{B}_{f,c} \tag{3}$$

is frustration-free. However, these "stabilizers" do not commute in the full Hilbert space; rather, they commute only in the $+1$ subspace of $\mathcal{B}_{f,c}$ operators, which we make more precise below. The algebra of the stabilizers has a different structure to the analogous algebras that occur in commuting stabilizer models. This also applies to relations between the logical operators, which are discussed in the next subsection.

More specifically, the group commutator $[V, W] \coloneqq VWV^{-1}W^{-1}$ between Clifford vertex stabilizers $\mathcal{A}_{v,c}$ always lies in the subgroup $\mathcal{S}^Z$ of the stabilizer group generated by the $\mathcal{B}_f$ plaquette terms:

$$[\mathcal{A}_{v,c}, \mathcal{A}_{v',c'}] = \mathcal{B}_{f,c''}\mathcal{B}_{f',c''}, \qquad c'' \neq c, c', \tag{4}$$

if $c \neq c' \in \{r, g, b\}$ and the operators $\mathcal{A}_{v,c}$ and $\mathcal{A}_{v',c'}$ overlap. Otherwise, the commutator is trivial. Here, (f, c'') and f' are the plaquettes of the third color whose centers coincide with (v, c) and (v', c'), respectively. As we can see, the vertex operators commute in the subspace where all the plaquette stabilizers are $+1$ and anticommute in the presence of nearby excitations (associated with violated plaquettes).

Finally, violations of stabilizer terms take the state out of the code space. Similar to the toric code, we call the pointlike excitations associated with violations of the vertex stabilizers charges (which are Abelian) and those associated with the plaquette stabilizers fluxes (which are non-Abelian). The three charge (flux) generators are of red, green, and blue color in correspondence with the color of the violated stabilizer, and we denote them as e_c (m_c), where $c \in \{r, g, b\}$. We direct the reader to Secs. IV and V, where this is discussed in more detail, as well as to Appendix B for a brief review of anyons of the TQD model.

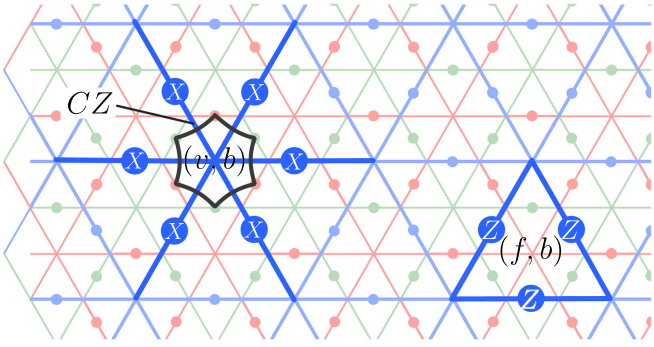


FIG. 3. Clifford vertex (left) and Pauli plaquette (right) stabilizers on the blue sublattice of the twisted quantum double. The stabilizers on the red and green sublattices are defined analogously.

B. Logical string operators

We now discuss the logical string operators of the twisted quantum double model, which are the operators that can generate rotations within the code space. The explicit form of these operators, and the algebra they generate, is useful for understanding the logical action of the protocol. Like the stabilizers, we find it convenient to write down the logical operators in terms of standard toric code logical operators that are decorated with a twist. The Pauli Z-type operators are the same as in the untwisted case: $\bar{\mathcal{Z}}^c_{H/V} \equiv \bar{Z}^c_{H/V}$. Physically, we can think of the unitary action of these operators as transporting electric charges of their corresponding color around noncontractible loops of the torus.

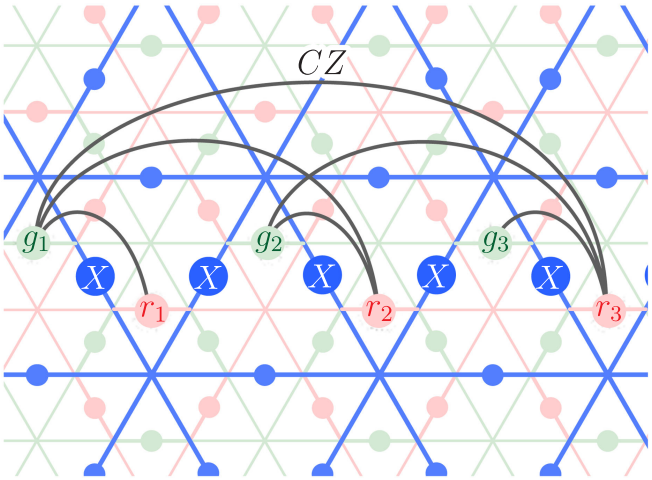


FIG. 4. A logical operator $\hat{X}^b_H$ for the twisted quantum double model. The gray lines depict CZ gates applied between the qubits at their end points. The blue string operator is decorated with controlled-phase gates connecting qubits on the red and green sublattices.

We obtain complementary logical operators for the TQD by supplementing the logical operators $\bar{X}^c_{V/H}$ of the standard toric code with additional CZ terms [56,60]. An example of one such operator, of color blue, is shown in Fig. 4. In order to specify the CZ interaction terms of this operator, we must order the qubits along a path around the torus. We choose an arbitrary starting point (in this case, we set it to be the green qubit g_1; see Fig. 4) and enumerate the green and red vertices in increasing order following the blue path around the torus. This allows us to write

$$\bar{\mathcal{X}}^b_H = \bar{X}^b_H \left(\prod_{r_i} \prod_{g_j:\, j \le i} CZ_{g_j, r_i} \right). \tag{5}$$

We find other logical operators by changing the color and orientation of this example. This operator is implemented by a unitary circuit of depth linear in the length of the string, consisting of CZ gates that are applied from each red qubit to all the green qubits to its left.

The logical operators of the twisted quantum double generate a non-Pauli (and non-Abelian) algebra. In fact, the algebra generated by all $\bar{\mathcal{X}}$ operators does not preserve the code space of the TQD; i.e., some combinations of these operators map to states out of the code space. We consider the commutator between an X-type logical operator and a vertex stabilizer with overlapping support. For example, for a blue logical operator and a red stabilizer, we find

$$\left[\bar{\mathcal{X}}^b_{H/V}, \mathcal{A}_{v,r} \right] = \mathcal{B}_{f,g}, \tag{6}$$

where the plaquette (f, g) is located within the star associated with the vertex (v, r). For each logical operator representative, there exists one vertex for which this commutation relation is different, namely,

$$\left[\bar{\mathcal{X}}^b_{H/V}, \mathcal{A}_{v,r} \right] = \mathcal{B}_{f,g} \bar{\mathcal{Z}}^g_{H/V}, \tag{7}$$

where the representative of the logical $\bar{\mathcal{Z}}^g_{H/V}$ is supported on the same lattice cells as the corresponding $\bar{\mathcal{X}}^b_{H/V}$. The special vertex exists due to the non-translation-invariant logical operator shown in Fig. 4. In our example, the special vertex appears at the "start point" of the loop. For example, for the blue logical operator in Fig. 4, the special red vertex operator is the one acting on qubits r_1 and r_3. The remainder of the commutation relations are the same under color permutations. Before concluding this subsection, let us define the charge-parity operator that appears frequently throughout the remainder of this section:

$$\hat{C}_c = \prod_v \mathcal{A}_{v,c}. \tag{8}$$

This is an operator that counts the parity of charges of color c on the torus. It follows from Eq. (6) that

$$\left[\bar{\mathcal{X}}^c_{H/V}, \hat{C}_{c'} \right] = \bar{\mathcal{Z}}^{c''}_{H/V}, \tag{9}$$

where $c'' \neq c, c' \in \{r, g, b\}$.

C. Code states and charge sectors

The code states of the TQD model are simultaneous $+1$ eigenstates of all stabilizer operators. Because of the non-Pauli algebra formed by the logical operators, the code space of the TQD on a torus (and the ground-state space of the associated Hamiltonian model) is 22-dimensional, which reflects the nonadditive nature of the code. A basis for the code space can be generated by applying certain combinations of the $\bar{\mathcal{X}}$-type logical operators to the all-0 ("vacuum") logical state $|\mathbf{0}\rangle$. The $|\mathbf{0}\rangle$ state is defined to be a simultaneous $+1$ eigenstate of all $\mathcal{Z}^c_{H/V}$ operators in addition to the stabilizers. Up to normalization, it can be written as

$$|\mathbf{0}\rangle = \prod_{v,c} \frac{1 + \mathcal{A}_{v,c}}{2} |0\rangle^{\otimes n}, \tag{10}$$

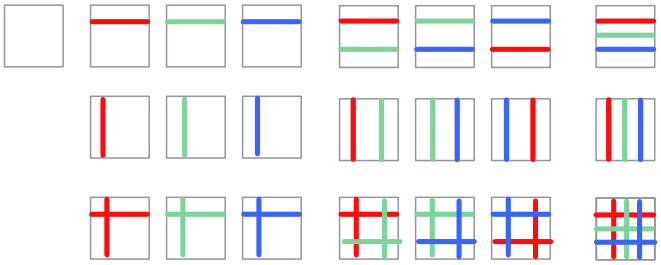
where $(1 + \mathcal{A}_{v,c})/2$, $c \in \{r, g, b\}$, is the projector onto the $+1$ eigenstate of the associated vertex operator. Of the 64 combinations of $\bar{\mathcal{X}}$-type string operators of three colors and two orientations, only 22 result in an eigenstate of all the vertex operators. To see this, we recast the product of all the vertex operators as

$$\hat{C}_c = (-1)^{(1-\bar{\mathcal{Z}}^{c'}_H)(1-\bar{\mathcal{Z}}^{c''}_V)/4} (-1)^{(1-\bar{\mathcal{Z}}^{c'}_V)(1-\bar{\mathcal{Z}}^{c''}_H)/4}. \tag{11}$$

The product of all vertex operators of color c is $+1$ (-1) when the total parity of intersections of the $\bar{\mathcal{X}}$-type strings of two complementary colors is even (odd). This has a clear physical interpretation, as we recall that $\hat{C}_c$ counts the total parity of charges of color c.

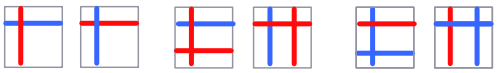
To obtain an odd parity of charges on a manifold with periodic boundary conditions, we use the fact that an intersection between two $\bar{\mathcal{X}}$ operators of different colors contains an unpaired charge of the third color [56]. An odd total number of charges is, thus, possible in the states that are obtained from $|\mathbf{0}\rangle$ by applying a combination of $\bar{\mathcal{X}}$ operators of different colors and orientations such that the resulting total parity of intersections is odd. The right-hand side of Eq. (11) counts precisely the parity of these intersections.

The 22 basis states for the code space of the TQD on a torus are depicted below:



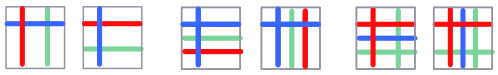
Here, each colored line denotes the application of an $\bar{\mathcal{X}}$ operator of corresponding color and orientation. All of the above states contain an even number of intersections between strings of any pair of distinct colors, and there are 22 such combinations.

In what follows, we must also consider the case when our protocol obtains an odd charge-parity subspace when we transform between the toric code model and the TQD. For the specific protocol that we consider in this section, odd parity of green charges can occur. There are six basis states containing a single unpaired green charge e_g, depicted below:



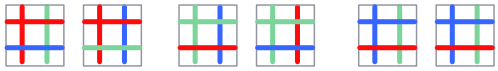
These states have an odd total number of intersections between strings of red and blue color. This subspace is characterized by the -1 eigenvalue of the $\hat{C}_g$ operator. The states with an odd number of e_r or e_b charges are obtained by appropriate color permutations.

For completeness, we characterize all types of charge-parity sector that occur in the TQD model. There are six states that simultaneously support unpaired charges of two different colors. For example, the subspace with unpaired e_r and e_g is spanned by the states depicted below:



Similar states occur for (e_g, e_b) as well as (e_b, e_r).

Finally, there are six states in the subspace with simultaneously unpaired e_g, e_r, and e_b charges, depicted below:



In total, the classes above result in

$$\underbrace{22}_{1} + \underbrace{3\times 6}_{e_c} + \underbrace{3\times 6}_{(e_c,e_{c'})} + \underbrace{6}_{(e_r,e_g,e_b)} = 64$$

combinations, where the braces indicate if there is an odd parity of certain number of charge types.

D. The $\overline{CZ}$ measurement protocol

We now describe the implementation of the simple measurement protocol and its logical action. We postpone the derivation of the protocol to Sec. IV. For simplicity, we assume noiseless operations; fault tolerance is discussed in Sec. VII.

The protocol is implemented as follows. We start with encoded states of the red and blue copies of the toric code on triangular sublattices of the corresponding color, as shown in Fig. 3. The qubits on the green sublattice are initialized in the $|0\rangle$ state. The initial stabilizer group on all qubits can be written as

$$\mathcal{S}_1 = \langle A_{v,r}, B_{f,r}, A_{v,b}, B_{f,b}, Z_{e,g}, B_{f,g}, \bar{Z}^g_H, \bar{Z}^g_V \rangle, \tag{12}$$

where we have listed redundant operators to highlight that the state is a $+1$ eigenstate of the $B^g_f \equiv \mathcal{B}^g_f$ plaquettes, as well as the $\bar{Z}^g_{H,V}$ operators.

With this initialization, we find that the code is stabilized by the red and blue vertex terms of the TQD $\mathcal{A}_{v,r}$ and $\mathcal{A}_{v,b}$, because the CZ operators in these vertex terms always act on at least one green qubit in the $|0\rangle$ state. Thus, we present the stabilizer group $\mathcal{S}_1$ alternatively as follows:

$$\mathcal{S}_1 = \langle \mathcal{A}_{v,r}, \mathcal{B}_{f,r}, \mathcal{A}_{v,b}, \mathcal{B}_{f,b}, Z_{e,g}, \mathcal{B}_{f,g}, \bar{Z}^g_H, \bar{Z}^g_V \rangle. \tag{13}$$

Thus, the only TQD stabilizers that have not been prepared are the $\mathcal{A}_{v,g}$ operators.

Next, we implement a transition to the TQD model by measuring its green vertex operators $\mathcal{A}_{v,g}$. We explain why this measurement prepares the twisted quantum double in greater depth in the next section. After this measurement, the stabilizer group becomes

$$\mathcal{S}_2 = \langle \mathcal{A}_{v,r}, \mathcal{B}_{f,r}, \mathcal{A}_{v,b}, \mathcal{B}_{f,b}, (-1)^{m_v}\mathcal{A}_{v,g}, \mathcal{B}_{f,g}, \bar{\mathcal{Z}}_H^g, \bar{\mathcal{Z}}_V^g \rangle. \tag{14}$$

Here, we have included the ± 1 measurement outcomes of $\mathcal{A}_{v,g}$ which are captured by $\mathbb{F}_2$-valued variables m_v. These random measurement outcomes specify a distribution of green Abelian charges. Upon adding the newly measured operators $(-1)^{m_v}\mathcal{A}_{v,g}$ to the stabilizer group, we keep only the Z_g-type operators from $\mathcal{S}_1$ that commute with all the newly measured terms, which are green plaquette terms and green logical-Z operators. We emphasize that the stabilizer group $\mathcal{S}_2$ we have prepared describes the TQD in a $+1$ subspace of green $\bar{\mathcal{Z}}$ logical operators. Additionally, the product of all green vertex operators is the charge-parity operator

$$\hat{C}_g = (-1)^{[(1-\bar{\mathcal{Z}}_{r,H})/2][(1-\bar{\mathcal{Z}}_{b,V})/2]} \times (-1)^{[(1-\bar{\mathcal{Z}}_{r,V})/2][(1-\bar{\mathcal{Z}}_{b,H})/2]} \tag{15}$$

that was introduced in Eq. (11).

Finally, the transition back to the red and blue toric code copies is achieved by decoupling the green qubits into Z eigenstates by measuring on-site Z operators, giving

$$\mathcal{S}_3 = \Big\langle \mathcal{A}_{v,r}, \mathcal{B}_{f,r}, \mathcal{A}_{v,b}, \mathcal{B}_{f,b}, (-1)^{m'_e} Z_{e,g}, \mathcal{B}_{f,g}, \bar{\mathcal{Z}}_H^g, \bar{\mathcal{Z}}_V^g, \prod_v (-1)^{m_v}\hat{C}_g \Big\rangle. \tag{16}$$

Here, we have added the newly measured operators to the stabilizer group and "kicked out" the stabilizers that do not commute with these measurements. The operator $\prod_v (-1)^{m_v}\hat{C}_g$ remains in the stabilizer group, as it commutes with the individual Pauli-Z operators (the Pauli-X component of $\hat{C}_g = \prod_v \mathcal{A}_{v,g}$ cancels upon multiplication).

We have included redundant operators $\mathcal{B}_{f,g}$, $\bar{\mathcal{Z}}_H^g$, and $\bar{\mathcal{Z}}_V^g$ in the stabilizer group $\mathcal{S}_3$ to explicitly show the constraints that measurement outcomes m'_e of single-qubit $Z_{e,g}$ operators on green qubits have to obey. In fact, the syndromes must form closed loops that can be corrected by applying products of $\mathcal{A}_{g,v}$ operators to fill the regions enclosed by the loops. Any correction that removes the closed loops on the green sublattice, but also commutes with the existing vertex and plaquette stabilizers, is valid. Products of operators $\mathcal{A}_{v,g}$ meet both of these requirements. Once the correction is performed, the resulting state is stabilized by $\mathcal{A}_{v,r/b}$ and $A_{v,r/b}$ simultaneously, similar to the beginning of the protocol. In addition, the state on the green sublattice completely decouples from the state on the red and blue sublattices. This recovers the original pair of red and blue toric codes with stabilizer group

$$\mathcal{S}_3 = \Big\langle A_{v,r}, B_{f,r}, A_{v,b}, B_{f,b}, Z_{e,g}, \prod_v (-1)^{m_v}\hat{C}_g \Big\rangle, \tag{17}$$

which includes the initial stabilizer group $\mathcal{S}_1$ along with the $\prod_v (-1)^{m_v}\hat{C}_g$ operator.

We now determine the logical action of the protocol. It is convenient to work in an eigenbasis of the logical Pauli-Z operators $\{\bar{Z}_V^r, \bar{Z}_H^b, \bar{Z}_H^r, \bar{Z}_V^b\}$, as these operators are preserved over the transformation from the Abelian phase onto the TQD and back again. We label the associate states as $\{rH, bV, rV, bH\}$. On the other hand, upon completing the operation, the operator $\prod_v (-1)^{m_v}\hat{C}_g$ is included in the stabilizer group of the code. The inclusion of this operator in the stabilizer group represents a projection onto the $+1$ eigenvalue eigenspace of $\pm\hat{C}_g$. Importantly, the expression in Eq. (15) can be written as the product of logical $\overline{CZ}$ operators

$$\hat{C}_g \equiv \overline{CZ}_{rH,bV}\overline{CZ}_{rV,bH}. \tag{18}$$

Thus, the protocol measuring $\hat{C}_g$ projects the state onto the $\prod_v (-1)^{m_v}$ eigenstate of a product of $\overline{CZ}$ operators.

This logical measurement can be used to prepare magic states if we choose the input logical state to be $|+_{rH}+_{bV}+_{rV}+_{bH}\rangle$. The protocol prepares a magic state of the following form:

$$|C_g = +1\rangle_{\text{out}} = \frac{3}{\sqrt{10}}|\overline{CZ}_{HV}\rangle|\overline{CZ}_{VH}\rangle + \frac{1}{\sqrt{10}}|1111\rangle, \tag{19}$$

$$|C_g = -1\rangle_{\text{out}} = \frac{1}{\sqrt{2}}|\overline{CZ}_{HV}\rangle|11\rangle + \frac{1}{\sqrt{2}}|11\rangle|\overline{CZ}_{VH}\rangle, \tag{20}$$

where

$$|\overline{CZ}\rangle = \frac{1}{\sqrt{3}}(|00\rangle + |01\rangle + |10\rangle) \tag{21}$$

is the CZ-magic state [54,55]. Both of the output states of this preparation protocol, Eqs. (19) and (20), are magic (nonstabilizer) states. In particular, one can use either of them to probabilistically recover the CZ state in its original form by measuring the pair of qubits (rH,bV) in the Pauli-Z basis and verifying we obtain the correct outcome such that the remaining state is a CZ state.

We verify the magic state preparation protocol in a code in the Jupyter notebook [61] by representing the stabilizers of the twisted quantum double phase as XS operators using the embedded code technique from Ref. [62] and simulating measurements of the XS operators using the technique set out in Ref. [59].

The protocols we present in this paper can be seen as generalized code deformations [23,47,63] beyond Pauli stabilizer codes [59,64]. Such a code deformation can be

formulated as a projective measurement on a stabilizer code $\mathcal{S}$ that is induced by measuring into the stabilizers of a new code $\mathcal{S}'$ whose stabilizers include a logical operator of $\mathcal{S}$. In the example above, measuring a Clifford operator prepares a magic state. This is followed by a reverse code deformation that restores the original code in a magic state, as the measurements that implement this reverse transformation commute with the logical operator that was measured.

E. Relation to gauging

The transitions between the Abelian and the non-Abelian topological codes in the protocol in this section correspond to *gauging* and *ungauging* the $\mathbb{Z}_2$ symmetry implemented by the operator in Eq. (15) on a pair of toric codes. This point of view allows us to explain the protocol purely in terms of universal, emergent anyon data. Here, we briefly discuss how to understand this in terms of the gauged symmetry group and how to identify the associated logical operator. This is discussed further in Appendixes C and D, including microscopic lattice model details that go beyond the example presented in this section.

Both gauging and ungauging transitions can be described by gapped domain walls between a "gauged" phase and the original "ungauged" phase [65,66]. They can be described as *condensation transitions* in which certain nontrivial defects before condensation are identified with the "vacuum" after condensation, meaning that after condensation these defects can be freely removed or inserted into the state via local operators [67]. Gauging a finite global symmetry group G corresponds to the condensation of G-domain walls, which are one-dimensional defects separating regions in which the symmetry was applied from regions where it was not applied. Ungauging refers to the opposite process in which Rep(G) bosons, pointlike defects at the end points of deconfined strings, are condensed [6].

When G is an Abelian group, gauging G can be implemented via local adaptive quantum circuits [68,69]. For this reason, the subclass of domain walls that are induced by gauging an Abelian symmetry automatically yield microscopic circuits for the associated logical gates. For example, in this section, $G \simeq \mathbb{Z}_2$ is the anyon-permuting symmetry of a pair of toric codes (see Appendix C) that corresponds to a finite-depth logical unitary $\overline{CZ}_{rH,bV}\overline{CZ}_{rV,bH}$ in the code. Gauging this symmetry implements a projective measurement of the corresponding Clifford logical operator, which we then use to prepare magic states. More generally, all protocols in this work in which the domain wall between the TQD phase and toric code phases lies perpendicular to the time direction can be regarded as gauging domain walls.

The approach of gauging symmetries of a code to obtain a logical action has the advantage of being readily applicable to general qLDPC codes [70] with symmetries. Gauging measurements have been discussed in this context in Ref. [48], where they were applied to symmetries of quantum codes to implement measurements of logical Pauli operators. However, when gauging a Clifford symmetry, the stabilizers after gauging are Clifford operators and are not guaranteed to commute outside the code space. In this work, we establish fault tolerance for a class of Clifford gauging measurements on topological codes; see Sec. VII. A general analysis of the fault tolerance of deformed qLDPC codes whose description is beyond the Pauli stabilizer formalism is an open problem.

IV. NON-CLIFFORD LOGICAL OPERATIONS FROM THE TQD PATH INTEGRAL

In this section, we present a systematic approach to non-Clifford logical operations from interfacing Abelian and non-Abelian topological phases via configurations of topological defects in spacetime, using a path-integral description of topological phases. We first review the TQD model, including its Euclidean spacetime path integral in the presence of defects, and then derive a whole family of protocols that operate similarly to the one introduced in Sec. III. These include $\overline{CZ}$ logical measurement and $\overline{XS}$ logical measurement (which prepares the T-magic state) as well as $\overline{CCZ}$ and $\bar{T}$ unitary gates. Among the protocols discussed in our work, the $\overline{CCZ}$ and $\bar{T}$ unitary gates have close predecessors in Refs. [22,23]. However, our method enables one to generate alternative realizations of these gates, and our other logical protocols are new.

This section aims at deriving the global, logical properties of the protocols which follow from the universal features of topological phases and their defects. To calculate these global properties, we first discuss microscopic continuum descriptions of the relevant phases and their properties. In the following Sec. V, we derive the microscopic circuit implementations of the protocols discussed in this section.

A. Global topology vs microscopic implementation

We start by discussing the difference between two distinct but related aspects of the protocols. The *global topology* of a protocol corresponds to a configuration in spacetime where each region of spacetime is filled by a topological phase (which can be trivial) and the interfaces between these regions host domain walls between adjacent topological phases (this further extends to interfaces of interfaces and so on; see Refs. [71,72]). For example, an interface to the trivial phase corresponds to a gapped boundary. Examples of distinct global topologies have already appeared in the text, in Figs. 1 and 2.

In this work, we focus on protocols that specifically involve copies of the toric code and TQD phases in three-dimensional spacetime volumes, gapped boundaries, and domain walls between these phases, as well as line and point defects. The global topology alone suffices to determine the logical operation performed by a protocol.

Given a fixed global topology, the overall logical action is independent of the microscopic details.

On the other hand, we define a *microscopic lattice implementation* to be a specific recipe used to implement a global topology. It specifies an implementation of the topological phases, boundaries, domain walls, and their interfaces, by an explicit, geometrically local quantum circuit with global classical communication and feedforward. There are two remarks regarding different microscopic implementations of a given global topology. First, there are different microscopic implementations of the same emergent topological phase in the continuum. For example, the Dijkgraaf-Witten gauge theories based on D_4 with no twist and $\mathbb{Z}_2^3$ with a type-III cocycle twist are equivalent. Second, there are different ways to realize a microscopic description of a topological phase on a lattice of qubits (for example, the TQD phase can be implemented via the model either in Ref. [7] or in Ref. [8]). Such microscopic implementations can be obtained from fixed-point path-integral representations of the topological phases, which we discuss in Sec. V A. Alternatively, microscopic implementations of protocols that implement logical measurements can be obtained via the gauging logical measurement procedure; see Sec. III E and Appendix C.

B. Circuits, path integrals, and the closed-membrane picture in spacetime

1. Fixed-point path integral

Here, we formulate a spacetime picture of the relevant topological phases. The well-known loop-sum picture for the relevant ground states (see Appendix A) follows naturally from this point of view, as we discuss below. We then show how to use the path-integral approach for the calculation of the logical action implemented by a given global topology.

Any nonchiral topological phase can be described in spacetime through a *fixed-point path integral* [9,73–78]. We define the path integral using a cellulation of spacetime where we place a variable on each of its cells. The path integral is a sum over discrete configurations $\vec{c}$ of these variables where each term in the sum is given by a product over local weights ω. Schematically,

$$Z = \sum_{\vec{c}} \prod_i \omega_i(\vec{c}), \tag{22}$$

where i is a local label in spacetime (usually labeling 3-cells) and $\omega_i(\vec{c})$ depends only on the configuration $\vec{c}$ in the vicinity of location i. For the path integrals considered here, the weight $\omega_i(\vec{c})$ is either a complex phase or equal to zero to impose specific local constraints on the configuration, such as, for example, closed-membrane constraints.

The value Z has an important meaning when it is evaluated on a lattice with an *input or output* state boundary. The input state boundary is usually placed at the earliest time of the protocol, and the output one is placed at the latest time. At such boundaries, there is no summation over the variables. For each configuration of variables on the state boundary $\vec{c}_\mathrm{b}$, we obtain a separate amplitude $Z(c_\mathrm{b})$. This type of boundary is distinct from a *physical boundary* which is discussed in Sec. IV C. We use the evaluation of the path integral with an input or output state boundary b to define a *boundary state* $|\psi_\mathrm{b}\rangle$ that is an input or output state for the corresponding boundary type. The computational-basis coefficients are the amplitudes, namely, $\langle \vec{c}_\mathrm{b}|\psi_\mathrm{b}\rangle = Z(\vec{c}_\mathrm{b})$.

To motivate the path-integral approach, we briefly explain how the topological fixed-point path integrals are related to circuits consisting of unitary gates and measurements [77,78]. The full discussion follows in Sec. V. It is relatively straightforward to turn a path integral with input and output state boundaries into a circuit of unitary gates and $+1$-postselected measurements, as the latter are simply projection operators. For this, we assign a time direction in the spacetime cellulation for the path integral. We turn the path-integral variables located at the same time, but different space, coordinates into the configurations of qubits at a corresponding time in the circuit. We then implement the local constraints and phases in the path integral given by weights $\omega_i(\vec{c})$ through projectors onto configurations that satisfy these constraints, as well as gates whose matrix elements are equal to the associated complex phases, respectively. Finally, we can replace the projectors (or the postselected measurements) with actual measurements. For this, we observe that the circuit with some of the measurement outcomes equal to -1 corresponds to a path integral with insertion of linelike defects [77,78] at appropriate locations.

We now describe the fixed-point path integrals that are used in this work. The first path integral is the one for the toric code phase. A $\mathbb{Z}_2 \simeq \{0,1\}$-valued variable is associated to every edge of a three-dimensional spacetime cellulation. The toric code path integral is an equal-weight sum over all configurations with an even number of 1 variables around each face (that is, there are weights that are nonzero only for configurations that satisfy this constraint). This constraint is, in fact, the *closed-membrane constraint*: Because of it, the Poincaré dual faces on each 1 variable form a closed-membrane pattern on the Poincaré dual cellulation. The path integral is, thus,

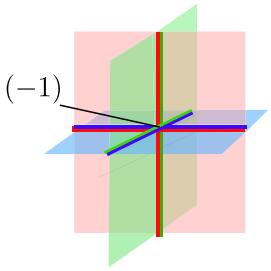
$$Z = \sum_{\vec{c}} \prod_f \delta\left(\sum_{\partial f} \vec{c} = 0 \bmod 2\right) \tag{23}$$

$$= \sum_{\vec{c}:\ \text{closed membranes}} 1, \tag{24}$$

where the second line is a schematic representation. Any input and output state for this path integral is an

equal-weight superposition of all closed-loop patterns, giving rise to the picture presented in Appendix A.

The second path integral is that of the TQD phase [7]. It is a sum over all closed-membrane configurations of three different colors, r, g, and b, where a -1 weight is associated to each triple intersection of three different colored membranes in spacetime, visually depicted below.



(25)

In the figure above and in the rest of this section, it suffices to use continuous pictures to represent the membranes and lines in spacetime, since we are focusing on global topology and global properties of the protocols. Schematically, the path integral is of the form [78]

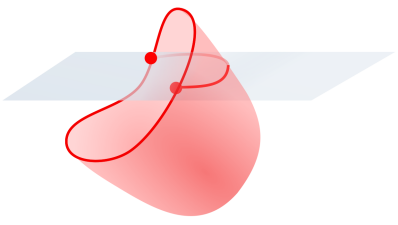
$$Z_{\mathrm{TQD}} = \sum_{\vec{c}_r,\vec{c}_g,\vec{c}_b:\ \text{closed membranes}} (-1)^{\#\ rgb\ \text{triple intersections}}. \quad (26)$$

For the input and output states, one obtains the three-colored closed-loop configurations forming the ground states of the TQD discussed in Appendix A. The history of these spatial closed-loop patterns evolving through time results in three-colored closed-membrane patterns in spacetime. Similarly, a spatial triple-crossing event, as shown in Fig. 21(c), corresponds to a triple-membrane intersection in spacetime, as shown in Eq. (25).

2. Charge and flux defects

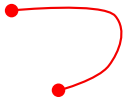
In order to relate the path-integral picture to circuits, we need to consider the path integrals for relevant phases after the insertion of *defects*. The defects in the toric code and the TQD phases correspond to either *charge* or *flux* worldlines.

We start by discussing flux defects, in the absence of any charge defects. In spacetime, flux defects are worldlines where the membranes of the path integral terminate and, accordingly, come in red, green, and blue colors. Below, we show such a worldline of red color as well as a section at a fixed time t:



(27)

From the time slice section, depicted above in gray, we obtain the picture in space. In space, the fluxes are points where the loops of one color are not closed but terminate:



(28)

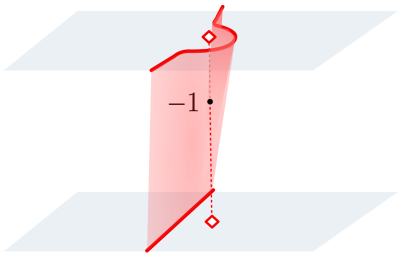
The path integral with flux defects inserted is modified in comparison to the usual Z_{TQD} by summing over membranes that terminate at the flux worldlines instead of only the closed ones. For example, the path integral with red flux worldline f in the absence of any charge worldlines is

$$Z_{\mathrm{TQD,\,flux}\ f} = \sum_{\substack{\text{closed } b,g \text{ membr.} \\ r \text{ membr. terminating at } f}} (-1)^{\#\ rgb\ \text{triple int.}}. \quad (29)$$

Microscopically, fluxes are points or lines on the Poincaré dual lattice in space as well as in spacetime; that is, they pierce perpendicularly through the faces of the primal lattice.

The insertion of a flux worldline in the TQD phase qualitatively alters the properties of the path integral: After insertion, the number of triple intersection points is no longer invariant under local deformations of the closed-membrane pattern (technically, the resulting action with a defect is not gauge invariant; i.e., it is not a function of the cohomology class). When a blue-green intersection line is deformed across a red flux worldline, this creates or removes a triple intersection; i.e., a local deformation yields a factor of -1. This implies that the path integral may depend on the geometric shape of the flux worldlines rather than their topology (more precisely, their cohomology class) alone, as it should normally be for topological phases. Hence, the continuously deformable closed-loop or membrane picture is not applicable in the presence of nontrivial flux worldline configurations.

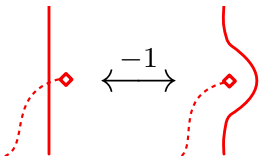
We next discuss what happens in the presence of charge defects with no flux defects. Charges also come in red, green, and blue color. In spacetime, the charge defects correspond to closed worldlines, where an additional -1 weight appears in the path integral for every intersection of a worldline with membranes of the same color:



(30)

In the spatial loop-sum picture, charge defects are points (shown as hollow diamonds) such that the states in the

superposition have a relative -1 for configurations related by a loop crossing this point:



(31)

The only modification to the path integral that is needed in order to add charge defects is to account for the possible -1 factor from charge-membrane crossings along every charge worldline. For example, adding a red charge worldline ℓ leads to the following path integral:

$$Z_{\mathrm{TQD,charge}\,\ell} = \sum_{\text{closed } rgb \text{ membr.}} (-1)^{\#\, rgb \text{ triple int.}} \times (-1)^{\#\,\text{red charge } \ell\text{-membrane int}}. \tag{32}$$

Charge defects of other colors are described similarly. Since the number of charge-membrane intersections is invariant under local deformations (when the fluxes are absent), the entire path integral is still gauge invariant when only charge defects are present. In fact, the charge worldlines correspond to the Abelian anyons of the TQD. See Appendix B for a short review of the anyons of the TQD.

Finally, we discuss what happens if we include both fluxes and charges into the path integral. As mentioned earlier, the presence of flux worldlines breaks the gauge invariance of the path integral (i.e., its invariance under local deformations of the closed-membrane pattern). Gauge invariance, however, ensures that charge worldlines must always be closed. Otherwise, the path integral evaluates to 0. As a consequence, if there are flux worldlines, then the charge worldlines *can now terminate on the flux worldlines*. More precisely, a charge worldline of color c can terminate on the flux worldlines of the other two colors $c' \neq c$. In addition, in a circuit realization, any configuration of such charge worldlines that terminate on the associated flux worldlines is equally probable. This has important consequences for decoding the protocols in the presence of noise; see Sec. VII.

C. Boundaries, domain walls, and corners of the TQD

We now briefly discuss the boundaries and the corners of the TQD phase as well as the domain walls between the TQD phase and the toric code. All of these spacetime defects are important for constructing the desired logical actions.

The most important type of defects in our protocols are *domain walls* that interface the TQD model with copies of the toric code. In our fault-tolerant circuits that implement logical protocols, these domain walls can, for example, appear on a timelike slice such as in Fig. 2, on a diagonal plane in spacetime such as in Fig. 1, or at a fixed spatial location. Several of the global topologies we consider also use domain walls between the TQD model and the vacuum, i.e., *gapped boundaries*. A number of the global spacetime topologies we consider include *corners*, which are one-dimensional interfaces between domain walls or boundaries. We present specific realizations of these defects in the closed-loop and closed-membrane picture below, without explicitly referring to a microscopic lattice. Explicit circuits that include corner defects can be derived from this picture by mapping loop configurations onto lattice cocycles.

1. Boundaries of the TQD

The boundaries of nonchiral topological orders in $(2+1)$ dimensions are well understood. They can be classified into topological superselection sectors, labeled by a finite number of inequivalent Lagrangian algebra objects [79–82]. For example, the toric code has two different boundary phases. The rough and smooth boundaries introduced in Ref. [53] are specific microscopic implementations of the distinct boundary phases. For the non-Abelian type-III TQD considered here, there are 11 distinct topological boundaries [83,84]. We briefly describe all 11 boundaries in the closed-membrane picture without reviewing their full classification.

Each boundary is determined by the way membranes of different colors are allowed to terminate and the boundary amplitudes of the path integral. The terminating membranes form a proper subgroup of $\mathbb{Z}_2^3$, and we label the associated boundary by angle brackets containing the generators of this subgroup that are allowed and also determine the charge anyon worldlines that are allowed to terminate on each boundary. Because of the nontrivial 3-cocycle, not all subgroups are allowed.

The 11 distinct boundaries of the TQD are as follows.

(i) $\langle \varnothing \rangle \equiv \langle \rangle$ boundary, where no membranes are allowed to terminate. All three generating charge anyon worldlines, and their combinations, are allowed to terminate on this boundary.

This is similar to the rough boundary of each of the three copies of the toric code.

[(ii)–(iv)] $\langle c \rangle$ boundary, where only the membranes of color $c \in \{r, g, b\}$ are allowed to terminate. Only charge anyon worldlines of complementary colors are allowed to terminate on this boundary (if we allowed a c-colored line to terminate at the boundary, the parity of line-membrane intersections of color c would not be invariant under local deformations).

For three copies of toric codes, this is similar to the boundary that is a smooth for the copy of color c and rough for the copies labeled by other colors.

[(v)–(vii)] $\langle c_1, c_2 \rangle$ boundary, where membranes of colors $c_1 \neq c_2 \in \{r, g, b\}$ are allowed to terminate. Only charge anyon worldlines of the third color are allowed to terminate on this boundary.

This is similar to a rough-smooth-smooth boundary for three copies of the toric code. In fact, for three copies of the toric code, there is one additional boundary for this subgroup that includes nontrivial boundary weights. However, for the TQD, these boundaries with and without the additional weights are phase equivalent.

[(viii)–(x)] $\langle c_1 c_2 \rangle$ boundary, where only the membranes of colors $c_1 \neq c_2 \in \{r, g, b\}$ must terminate simultaneously along the same line (and cannot terminate individually). Charge worldlines of colors c_1 and c_2 are allowed to terminate simultaneously on the same point on this boundary, as well as individual anyon worldlines of the third color.

For three copies of the toric code, this corresponds to a folding boundary between the copies of colors c_1 and c_2 and a rough boundary for the third copy.

(xi) $\langle rg, rb \rangle$ boundary, where any pair of membranes of different colors is allowed to terminate simultaneously but none of them individually. The charge worldlines of all three colors are allowed to terminate simultaneously (but no individual worldlines or pairs are allowed to).

This boundary requires additional weights, namely, a factor of i at each intersection between an rg and an rb termination line on the boundary. In space, this weight is associated to a process of exchanging two pair-termination-points on the boundary. This is illustrated and explained in more detail in Appendix F.

2. Domain walls between the TQD and toric codes

Our protocols act on logical states of toric and surface codes, and, hence, we require interfaces between the toric code and the TQD phases to transfer the logical information into and out of the non-Abelian phase. Similar to gapped boundaries, domain walls are described by ways in which membranes can terminate, as well as additional weights on the domain wall.

The domain walls we consider are equivalent to gapped boundaries of a stack consisting of the TQD and toric codes, as the toric codes can be folded onto the same side of the domain wall as the TQD. These domain walls are fully classified [79,83,84]; however, the complete list of possibilities is too numerous to discuss in full here, and so we restrict our attention to domain walls that are relevant for our purposes.

First, we consider the relevant domain walls between a single toric code (which we label with the color purple, denoted by p) and the TQD. We focus exclusively on domain walls that we can use to transfer all logical information from the toric codes to the TQD. For the types of domain walls used in our protocols, this is ensured by the fact that each type of toric code membrane is forced to terminate simultaneously with some nontrivial TQD membrane. We note that such domain walls exist between one single toric code, or a pair of toric codes, and the TQD but not between three toric codes and the TQD.

The termination pattern is given by a subgroup of $\mathbb{Z}_2^4$, where the first three $\mathbb{Z}_2$ factors correspond to the TQD and the last $\mathbb{Z}_2$ factor corresponds to the toric code. In this paper, we use $\mathbb{Z}_2$ to refer to the group $\{0, 1\}$ with addition modulo two and, therefore, use the notation interchangeably with $\mathbb{F}_2$ in several places. The domain walls that we use in our protocols are the following:

(i) $\langle cp \rangle$ domain wall where $c \in \{r, g, b\}$. Only the membrane of color c on the TQD side can terminate simultaneously with the toric code membrane p along a common termination line.

(ii) $\langle c_1 p, c_2 \rangle$ domain wall with $c_1 \neq c_2 \in \{r, g, b\}$, where the membrane of color c_1 from TQD side can terminate only simultaneously with the p membrane of the toric code, and, additionally, the membrane of color c_2 can freely terminate.

Our protocols also use the domain walls between the TQD and two copies of the toric code. In this case, we color the membranes of the copies of toric code purple (p) and yellow (y). The relevant domain walls are as follows.

(i) $\langle c_1 p, c_2 y \rangle$ domain wall with $c_1 \neq c_2 \in \{r, g, b\}$, where TQD membranes of color c_1 can terminate simultaneously with the purple toric code membranes, and TQD membranes of color c_2 can terminate simultaneously with yellow toric code membranes.

(ii) $\langle rgp, rby \rangle$ domain wall, which is derived from the $\langle rg, rb \rangle$ boundary of the TQD by forcing the rg and rb termination lines to coincide with the termination lines of the p and y toric codes, respectively.

3. Corners of the TQD

In some of the logical protocols, the types of corners used play an important role. For this reason, we now review the types of corners that can occur in the TQD phase.

In a global topology with multiple different boundary conditions, there are also corners, which are lines in spacetime that correspond to interfaces between boundaries. For each pair of boundaries, the corners can be fully classified into topologically distinct superselection sectors. The distinct corners are the irreducible blocks of a *-algebra corresponding to a two-dimensional path integral obtained by compactifying the bulk into a thin slab with the two boundaries at the top and bottom [79,81,85–87].

The corners we consider simply interface termination lines on the boundaries on either side. For example, a corner between the $\langle rg, rb \rangle$ and the $\langle r, g \rangle$ boundary may interface an rg termination line on one side with a separate r and g termination line on the other side [see Eq. (F9) in Appendix F and the surrounding discussion]. In general, there exist different superselection sectors of corners between the same boundary types, which are distinguished by assigning different weights to the points on the corner where termination lines are interfaced. For the configurations we consider, there are only two different

weights that differ by a factor of -1, so the choice of corner changes the protocols by only an unimportant single-qubit logical Z operator. Specifically, the weights are always ± 1 (and we choose the "trivial" $+1$ weight), except for the corners between the $\langle rg, rb\rangle$ and $\langle g, b\rangle$ boundaries, where the weights are $\pm\omega$ with $\omega = e^{-i\pi/4}$ (we choose the $+\omega$ weight). Note that the ω weight of this corner is the source of the "non-Cliffordness" (magic) in the protocols for the logical $\overline{XS}$ measurement (that can be used for magic state preparation) and the $\bar{T}$ gate.

Note that the TQD phase also allows for more intricate corners where we sum over additional closed-loop configurations inside the corner worldline, but we do not make use of these. We refer the reader to Appendix F for a more complete discussion of the corners.

Finally, the zero-dimensional defects in spacetime (such as the end points or meeting points of the corners) can be identified with the ground states for the spatial configuration obtained by taking the intersection of the spacetime with a (sufficiently small) two-sphere surrounding the point. If the according ground-state space has dimension 2 or higher, then this means that part of the logical information is not encoded globally but can be accessed by operators acting locally at this point. For the global protocol to be fault tolerant, the ground space dimension, thus, must be one for every spacetime point, in which case local operators can give rise to only irrelevant global prefactors.

D. Logical operations

In this section, we construct different global topologies that result in specific logical operations. For each global topology, we label each 3-volume by the topological phases that are either the TQD, a single toric code (TC), or two copies of the toric code (TC $\times$ TC). We further specify the types of domain walls and boundaries in a given geometry using the notation introduced above. We also designate the state boundaries with input and output states, which we label as logical input (i) and output (o) states of the protocol and ensure that they always belong to the toric code phase. We also ensure that the distance between any two points in spacetime, where flux or charge defect termination can affect the logical state, is at least d—the desired distance of the code. An example of such a pair of points is any pair of points on two distinct boundaries where the same type of flux or charge can terminate. For each labeled global topology, we then derive its logical action using the closed-membrane picture for the path integrals.

1. Computing the logical action

First, we explain how to compute the logical operation corresponding to a global topology, in general, and then we consider specific examples. We start by determining the generating cohomology classes of string configurations on the input and output states of the input toric or surface codes, which are in one-to-one correspondence with the logical states in the Pauli-Z basis. We label the ith input or output state boundary as I_i or O_i, respectively. The total input and output cohomology classes are then of the form

$$\sum_i \alpha_i I_i \quad \text{and} \quad \sum_i \beta_i O_i,$$

for some $\alpha_i, \beta_i \in \mathbb{Z}_2$, respectively. They form a basis for the input and output vector spaces for the logic operation, which we denote by $|\alpha\rangle$ and $\langle\beta|$, respectively. Here, we have defined $\mathbb{Z}_2$-valued vectors $\alpha = (\alpha_1, \alpha_2, \ldots)$ and $\beta = (\beta_1, \beta_2, \ldots)$. The input and output states that we use in our protocols are placed on either tori (containing the toric code copies), rectangular surface code patches, or surface code patches that are folded into triangles, which are equivalent to the color code [88].

We use the path integral to define an operator F whose matrix elements $\langle\beta|F|\alpha\rangle$ determine the logical operation applied between input and output states. To compute these matrix elements, we first use the global topology to determine a set of generating cohomology classes of "bulk" membranes in the TQD and toric code phases, which we label B_j. These can be thought of as the path along which the logical information can continuously flow between the input and output states. The bulk cohomology classes are, thus, $\sum_j \gamma_j B_j$ for some $\gamma_j \in \mathbb{Z}_2$, and we keep track of them via an $\mathbb{Z}_2$-valued vector $\gamma = (\gamma_1, \gamma_2, \ldots)$. By restricting B_j to the ith input state boundary, we obtain $\mathcal{I}_{i,j} \in \mathbb{Z}_2$. We define $\mathcal{O}_{i,j}$ analogously for the restriction of B_j to the ith output state boundary. Both $\mathcal{I}$ and $\mathcal{O}$ are, thus, $\mathbb{Z}_2$-valued matrices.

Finally, for every bulk cohomology class, we further determine the weight associated to this closed-membrane configuration by the path integral which we denote $Z(\gamma)$. We then use these weights to evaluate the desired matrix elements:

$$\langle\beta|F|\alpha\rangle = \sum_{\gamma:\mathcal{I}\gamma=\alpha,\mathcal{O}\gamma=\beta} Z(\gamma). \tag{33}$$

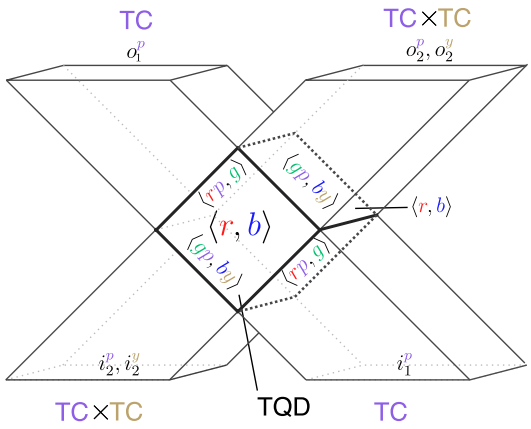
In some cases, the operator F is not unitary but rather a partial isometry. This occurs when it is not possible to find a correction that maps the corresponding circuit with an arbitrary set of observed measurement outcomes to the $+1$-postselected circuit. In this case, the classical measurement outcomes contain information about the logical input states. The resulting logical operation is a quantum channel rather than a unitary. In our protocols, this scenario occurs if there exist nontrivial cohomology class generators C_i for charge worldlines that go between input and output state boundaries. The different charge cohomology classes are $\sum_i \epsilon_i C_i$ for some $\epsilon_i \in \mathbb{Z}_2$ and $\epsilon = (\epsilon_1, \epsilon_2, \ldots)$. In this case, the path-integral weights also depend on ϵ, and we

denote them $Z^\epsilon(\gamma)$. The resulting logical operation is a measurement defined by a collection of operators F^ϵ, where ϵ corresponds to the logical measurement outcome. The matrix elements of these operators are

$$\langle\beta|F^\epsilon|\alpha\rangle = \sum_{\gamma:\mathcal{I}\gamma=\alpha,\mathcal{O}\gamma=\beta} Z^\epsilon(\gamma). \tag{34}$$

2. $\overline{CCZ}$ unitary gate

For our first example, we consider the protocol realizing the unitary $\overline{CCZ}$ gate that is briefly described in Sec. II (and Appendix D). The corresponding global topology is shown below:

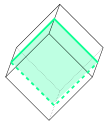
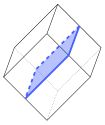
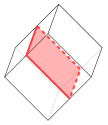


(35)

Here, and in all the protocols in this section, we orient the figures such that the time direction used to derive a circuit flows up the page. The global topology above is built from a cube containing the TQD phase and regions containing either one or two copies of the toric code phase. Two of the opposing faces of the TQD cube are interfaces with cubes that contain a single copy of the toric code phase. Another pair of opposing faces of the TQD cube are interfaces with cubes containing two copies of the toric code. The other two faces of the TQD cube correspond to gapped boundaries.

Each region containing (copies of) the toric code has a face that is an input or an output state boundary labeled by i or o, correspondingly. There is a subscript on the input or output label enumerating the inputs and outputs and a superscript corresponding to the copy of the toric code phase that is labeled by the color p or y. We omit the boundary labels on the faces of the regions containing (copies of) the toric code phase. These are either $\langle\rangle$, $\langle p\rangle$, or $\langle y\rangle$, corresponding to rough and smooth surface code boundary conditions in the associated copy. The boundary labels are chosen such that all spatial slices define surface codes with rough and smooth boundaries that produce logical operators consistent with the figures below.

There are three generating input classes and three generating output cohomology classes, corresponding to logical states of the individual surface code patches. These are labeled I_2^p, I_2^y, I_1^p and O_2^p, O_2^y, O_1^p, following the labeling in Eq. (35). There are three bulk cohomology class generators:

$B_g =$ , $B_b =$ , $B_r =$ . (36)

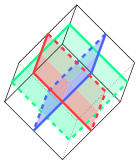
In this example, there are no nontrivial cohomology classes of charge anyon worldlines. Hence, the resulting logical operation is unitary. From the above figures, we find the following compatibility matrices between bulk and boundary cohomology classes to be

$$\mathcal{I} = \begin{pmatrix} & I_1^p & I_2^p & I_2^y \\ B_g & 0 & 1 & 0 \\ B_b & 0 & 0 & 1 \\ B_r & 1 & 0 & 0 \end{pmatrix},$$
$$\mathcal{O} = \begin{pmatrix} & O_1^p & O_2^p & O_2^y \\ B_g & 0 & 1 & 0 \\ B_b & 0 & 0 & 1 \\ B_r & 1 & 0 & 0 \end{pmatrix}. \tag{37}$$

The weight obtained from the path integral as a function of bulk cohomology classes is

$$Z(\gamma) = (-1)^{\gamma_g\cdot\gamma_b\cdot\gamma_r}. \tag{38}$$

The nontrivial weight is depicted below:

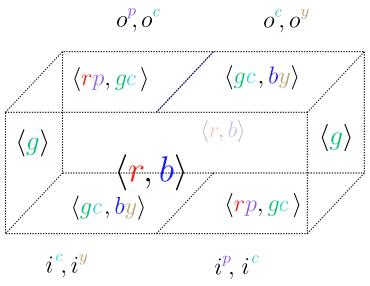
$$Z((1,1,1)) = Z\Big($$  $$\Big) = -1\,, \tag{39}$$

where membranes of all three colors are present with different orientations, which results in an odd number of triple intersection points. At this point, we have described the necessary data. We now evaluate the sum in Eq. (33). For $\alpha \neq \beta$, the sum is over the empty set and yields zero, and if $\alpha = \beta$, the sum is over a single bulk cohomology class determined by $\gamma = \mathcal{I}^{-1}\alpha = \mathcal{O}^{-1}\beta$. This gives the expression

$$\langle\beta|F|\alpha\rangle = \delta_{\alpha_1^p,\beta_1^p}\delta_{\alpha_2^p,\beta_2^p}\delta_{\alpha_2^y,\beta_2^y}(-1)^{\alpha_1^p\cdot\alpha_2^p\cdot\alpha_2^y}, \tag{40}$$

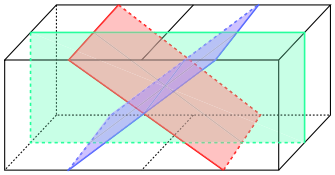
which coincides with the CCZ logic gate.

Below, we depict a deformation of this protocol:



(41)

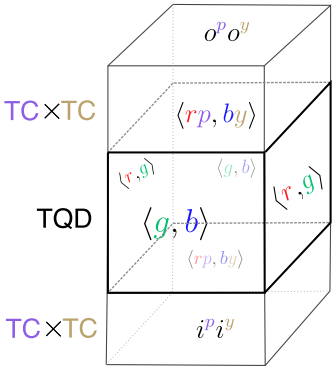
where we did not show the regions containing the toric code copies [we show only the labels of the corresponding input and output state boundaries; the regions can be added by analogy with Eq. (35)]. We added a c-colored copy of the of the toric code, where "c" stands for "cyan." Compared to Eq. (35), we introduced additional $\langle g\rangle$ boundaries such that the domain walls between the TQD and toric code are purely spatial. These additional boundary sections do not change the logical action of the protocol, and the analysis is the same as in Eq. (35). For example, the bulk cohomology class $\gamma_g = \gamma_b = \gamma_r = 1$ with $Z(\gamma) = -1$ is shown below:



(42)

3. Logical $\overline{CZ}$ measurement

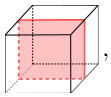
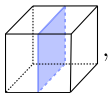
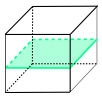
The next example we consider corresponds to a global topology that yields a logical CZ measurement. This is a variant of the example in Secs. II and III, with open spatial boundary conditions. See also the example in Appendix D.



(43)

The cube in the middle of this configuration hosts the TQD phase which interfaces two copies of the toric code above and below it. The spatial sections of the regions hosting the toric code phase can be chosen to either be tori (as in Sec. III) or rectangles (in which case the initial and final codes are the surface codes). In the latter case, we choose the spatial boundary conditions such that they are consistent with the figures below. We consider only the case with input and output surface codes here; the case with input and output toric codes can be calculated analogously.

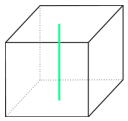
There are two generating input and output cohomology classes I^p, I^y and O^p, O^y, which we label as in Eq. (43). There are three generating bulk cohomology classes:

$B_r =$ , $B_b =$ , $B_g =$ . (44)

The restriction of the bulk cohomology classes to the input and output is described by

$$\mathcal{I} = \begin{pmatrix} & I^p & I^y \\ B_g & 0 & 0 \\ B_b & 0 & 1 \\ B_r & 1 & 0 \end{pmatrix}, \quad \mathcal{O} = \begin{pmatrix} & O^p & O^y \\ B_g & 0 & 0 \\ B_b & 0 & 1 \\ B_r & 1 & 0 \end{pmatrix}. \tag{45}$$

There is also a single nontrivial bulk charge-line cohomology class generator:

$C_g =$ , (46)

so the cohomology classes are of the form $\epsilon_g \cdot C_g$ for $\epsilon_g \in \mathbb{Z}_2$. The weights of the bulk cohomology classes are

$$Z^{\epsilon_g}(\gamma) = (-1)^{\gamma_r \cdot \gamma_b \cdot \gamma_g + \epsilon_g \cdot \gamma_g}. \tag{47}$$

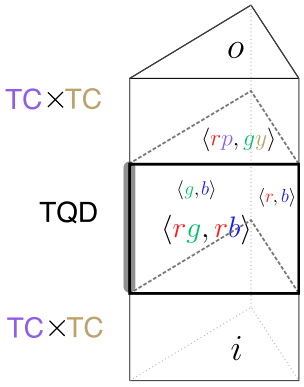
We now evaluate the sum in Eq. (34). For $\alpha \neq \beta$, there is no γ such that $\mathcal{I}\gamma = \alpha$ and $\mathcal{O}\gamma = \beta$. So the sum is over the empty set and yields zero. On the other hand, if $\alpha = \beta$, the sum is over two cohomology classes, namely, $\gamma_r = \alpha^p = \beta^p$, $\gamma_b = \alpha^y = \beta^y$, and $\gamma_g \in \{0, 1\}$. For $\epsilon_g = 0$ and $\alpha^p = \alpha^y = 1$, the two summands cancel. The same happens for $\epsilon_g = 1$ and $\alpha^p = 0$ or $\alpha^y = 0$. In all other cases, both summands are $+1$. Finally, we obtain

$$\langle \beta | F^{\epsilon_g} | \alpha \rangle = \delta_{\alpha^p, \beta^p} \delta_{\alpha^y, \beta^y} \delta_{\epsilon, \alpha^p \cdot \alpha^y}. \tag{48}$$

For $\epsilon_g = 0$ and $\epsilon_g = 1$, these are the matrix elements of the two projection operators associated to a projective measurement of the observable $\overline{CZ}$ with measurement outcome $(-1)^{\epsilon_g}$.

4. $\bar{T}$-state preparation

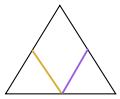
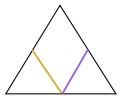
The global topology that we use to implement a nonunitary logical "XS measurement" operation is shown below. See also the example in Appendix D.



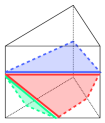
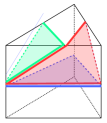
(49)

Here, the triangular prism in the middle supports the TQD, with the regions below and above containing two copies of the toric code phase each. The "$+\omega$" corner between $\langle rg, rb\rangle$ and $\langle g, b\rangle$ boundaries is shown highlighted in dark gray. The surface codes are placed on a triangle with the "fold" boundary conditions (the fold corresponds to the $\langle py\rangle$ boundary placed on the front face), equivalent to a single copy of a surface code folded into a triangle. This is analogous to the color code triangle [88]. The boundary conditions are otherwise chosen to be consistent with both the membranes that can pass into the TQD and condense on the boundaries there and the figures below.

There is a single generating input, and output, cohomology class, which we label I, and O, respectively:

$I =$  , $O =$  . (50)

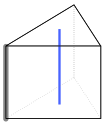
There are two generating bulk cohomology classes:

$B_0 =$  , $B_1 =$  . (51)

The restriction maps from the bulk cohomology classes to the input and output state cohomology classes are

$$\mathcal{I} = \begin{pmatrix} & I \\ B_0 & 1 \\ B_1 & 0 \end{pmatrix}, \qquad \mathcal{O} = \begin{pmatrix} & O \\ B_0 & 0 \\ B_1 & 1 \end{pmatrix}. \tag{52}$$

There is a single nontrivial charge worldline cohomology class generator in the bulk:

$C_b =$  . (53)

The weights of the bulk cohomology classes are

$$Z^{\epsilon_b}(\gamma) = \omega^{\gamma_0}\bar{\omega}^{\gamma_1}(-1)^{\epsilon_b\cdot(\gamma_0+\gamma_1)}. \tag{54}$$

The factor ω^{γ_0} ($\bar{\omega}^{\gamma_1}$) appears due to the termination of the membranes corresponding to B_0 (B_1) at the $+\omega$ corner between $\langle rg, rb\rangle$ and $\langle g, b\rangle$ boundaries (which is highlighted in dark gray in the figure), following the discussion in Sec. IV C 3. We now evaluate the sum in Eq. (34). For every α and β, we sum over only one bulk cohomology class, $\gamma_0 = \alpha, \gamma_1 = \beta$. Hence, we obtain the matrix elements

$$\langle\beta|F^{\epsilon_b}|\alpha\rangle = \omega^{\alpha}\bar{\omega}^{\beta}(-1)^{\epsilon_b\cdot(\alpha+\beta)}, \tag{55}$$

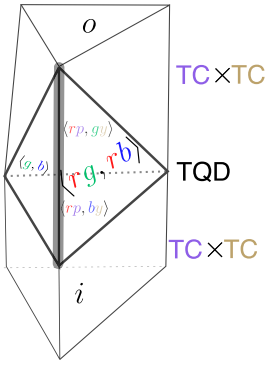
or in matrix notation

$$F^0 = \begin{pmatrix} 1 & \omega \\ \bar{\omega} & 1 \end{pmatrix}, \qquad F^1 = \begin{pmatrix} 1 & -\omega \\ -\bar{\omega} & 1 \end{pmatrix}. \tag{56}$$

For $\epsilon_b = 0, 1$, these are precisely the projection operators onto the eigenstates of the $\overline{XS}$ operator, which are the $T^\dagger$-magic states with additional relative phase $(-1)^{\epsilon}_b$. Therefore, we colloquially call this action the "logical $\overline{XS}$ measurement."

5. $\bar{T}$ unitary gate

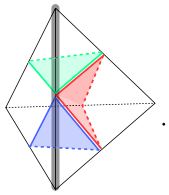
The final example we present in this section corresponds to a global topology that implements the $\bar{T}$ gate. This protocol is closely related to the $(2+1)$-dimensional non-Clifford gate in Ref. [22].



(57)

The tetrahedron in the middle contains the TQD phase, which is connected to two double-toric code phases across two of its triangular faces. The $+\omega$ corner between $\langle rg, rb\rangle$ and $\langle g, b\rangle$ boundaries is shown highlighted in dark color. The toric codes have color-code-like boundary conditions, consistent with the membranes that pass into the TQD and condense on boundaries there. There is a single generating

input, and output, cohomology class which we denote by I and O, respectively. There is also a single generating bulk cohomology class:

$B =$  . (58)

The restriction of the bulk cohomology class to the state boundary is simply

$$\mathcal{I} = \begin{pmatrix} & I \\ B & 1 \end{pmatrix}, \qquad \mathcal{O} = \begin{pmatrix} & O \\ B & 1 \end{pmatrix}. \tag{59}$$

There are no cohomologically nontrivial charge worldlines. The weights of the bulk cohomology classes are determined by whether there is a membrane termination at the $+\omega$ corner between $\langle rg, rb\rangle$ and $\langle g, b\rangle$ boundaries (highlighted in dark gray in the figure):

$$Z(\gamma) = \omega^{\gamma}. \tag{60}$$

We now evaluate the sum in Eq. (34). If $\alpha \neq \beta$, due to Eq. (59) the sum is over the empty set and, thus, is zero. If $\alpha = \beta$, the sum is over a single bulk cohomology class, $\gamma = \alpha = \beta$. Finally, we obtain the matrix elements

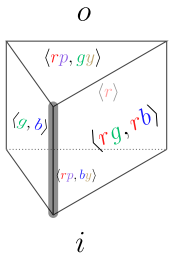
$$\langle\beta|F|\alpha\rangle = \delta_{\alpha,\beta}\omega^{\alpha}, \tag{61}$$

which can be represented as the matrix

$$F = \begin{pmatrix} 1 & 0 \\ 0 & \omega \end{pmatrix}. \tag{62}$$

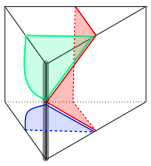
Hence, the logical operation is the unitary $\bar{T}$ gate.

A simpler variation of this configuration with the same topology is shown below:



(63)

Here, we do not explicitly show the regions hosting copies of the toric code phase. This geometry is obtained by deforming the geometry shown in Eq. (57) and splitting the back edge into a pair of edges separated by the additional $\langle r\rangle$ boundary in the back, followed by shifting one to the bottom and one to the top. The single generating bulk cohomology class $\gamma = 1$ is shown below:

$B =$  (64)

Following the analysis above, this results in a logical $\bar{T}$ gate.

V. CIRCUIT IMPLEMENTATIONS FROM THE PATH INTEGRAL

In this section, we discuss microscopic circuits for spacetime geometries that result in non-Clifford logical operations. We discuss a systematic approach to derive circuits realizing logical protocols on arbitrary lattices (or spacetime cellulations) that can be used to obtain various microscopic implementations of the same protocol. The approach naturally gives rise to specific circuits with a given ordering of unitary and measurement operations. The circuits we describe in this section can be brought into a form where they consist of repeated measurements of the Clifford and Pauli stabilizers (for example, such as the circuit considered in Sec. III) by applying local circuit equivalences and grouping operations. However, the same method can also be used to derive Floquet-code or measurement-based-quantum-computing–like circuits as opposed to ones that repeatedly measure the same stabilizers. For example, one can obtain a circuit for the TQD phase that looks like three copies of the honeycomb Floquet code coupled with additional CCZ gates; see Ref. [78].

We first explain how to derive circuits from the path-integral formalism using a specific example, which we refer to as the "cup-product" implementation. We then briefly discuss other examples (for example, making the connection to the model in Sec. III) and alternative ways to derive and understand these circuits. In addition, Appendix D shows more microscopic examples which are discussed in a language similar to Sec. III.

A. Cup-product microscopic lattice realization of the TQD

We first explain how to derive the circuit from the path-integral description for the simplest protocol that has trivial logical action, namely, the one that simply realizes the TQD phase without boundaries or domain walls (i.e., on a torus). After that, we explain how to derive the circuits for nontrivial logical protocols corresponding to specific spacetime geometries.

1. Preliminaries: Circuits for the toric code phase

We start by briefly reviewing how to obtain the circuit for the toric code from the path-integral framework (see Sec. VA 1 as well as Refs. [77,89] for a more comprehensive discussion). Recall that the path integral for the toric code can be written as

$$Z = \sum_{\text{confs. } \vec{c}} \prod_{\text{faces } f} \omega_f(\vec{c}) = \sum_{\vec{c}} \prod_f \delta\left(\sum_{\partial f} \vec{c} = 0 \bmod 2\right). \tag{65}$$

That is, the weights of the path integral $\omega_f(\vec{c}) = \delta(\sum_{\partial f} \vec{c} = 0 \bmod 2)$ enforce the constraint that the variables on the edges ∂f around the boundary of each plaquette f must have even total parity. Here, we describe the circuit on a square lattice obtained from the path integral on a cubic lattice. The approach, however, can be used for arbitrary spacetime cellulations and for other topological phases [78].

We first turn a path integral with input and output state boundaries into a circuit of unitaries and projector operators (i.e., $+1$-postselected measurements). Define the time direction to be along one of the axes of the cubic lattice. We call the edges that are parallel to the time direction *timelike* and the other edges *spacelike*. Then, applying the projectors $\frac{1}{2}(1 + XXXX)$ for every vertex v and $\frac{1}{2}(1 + ZZZZ)$ for every horizontal face f at each time step realizes the same action between input and output states as the expression for the path integral, upon appropriate identification between $\mathbb{Z}_2$ variables and qubits (shown in Appendix E).

Next, we replace the projectors with measurements. The -1 measurement outcome of an $XXXX$ operator at vertex v at time t corresponds to a factor of $(-1)^{\text{variable on a } t \text{ edge}}$ in the path integral, which is the same as inserting the charge worldline at the timelike edge (t edge). The -1 measurement outcome of the $ZZZZ$-plaquette operator replaces the parity-even constraint with the parity-odd constraint on that plaquette in the path integral. This means that there is a flux defect worldline going through this plaquette in the path integral. In practice, for every vertex (plaquette) measurement, we add an ancilla qubit at the associated vertex (plaquette). At each time step, this ancilla is initialized in the $|+\rangle$ ($|0\rangle$) state, then coupled to the qubits participating in the measurements via CNOT gates, and finally measured in the Pauli-X (Pauli-Z) basis. The -1 outcomes then record the locations of the worldlines of respective defects in spacetime.

Finally, we can also derive the circuit for the slanted version of the cubic cellulation (shown in Fig. 5). This circuit is obtained by splitting the time step t into an appropriate number of substeps. At each substep, the gates in the circuit described above can implement only those weights (here, the parity constraints) of the path integral that involve variables that are currently "active," that is, represented by qubits. Finally, the ancilla qubits are measured, which completes the full period. The circuit is periodic in time, and each period is subdivided into five steps. The step t_0 corresponds to single-qubit m_X (m_Z) ancilla measurements followed by $|+\rangle$ ($|0\rangle$) initialization. The circuit is equivalent to the one shown in Fig. 6 if one skips steps t_2 and t_5, assuming that there is one data qubit per edge and an ancilla qubit per vertex and per plaquette shown as white dots.

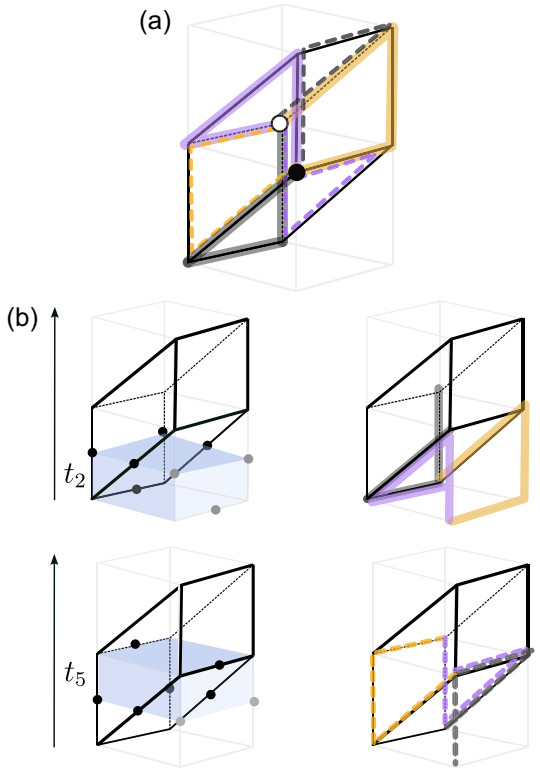


FIG. 5. (a) Figure used to define the cup product on a slanted cube in spacetime that appears in the path integral of the TQD phase. The time arrow points upward. (b) Two time steps labeled t_2 and t_5 of the circuit during which we implement the cup-product weights of the TQD path integral. The spacetime regions on the left that are shaded in blue denote the degrees of freedom that correspond to the qubits that are "live" in the circuit at the corresponding time step. On the right, the paths that are supported only on the live degrees of freedom at a given time step are shown. Some of the paths belong to neighboring slanted cubes.

2. Circuits implementing the TQD phase

A very natural microscopic representation of the type-III TQD path integral can be obtained from the action of the according Dijkgraaf-Witten model [9] in terms of *cup products*. Informally speaking, the cup-product expression precisely counts the number (modulo 2) of triple

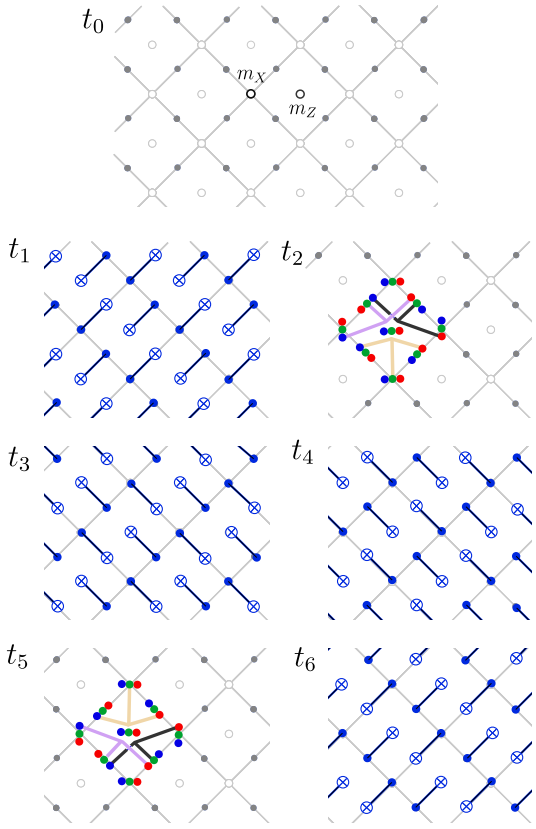


FIG. 6. Seven-step circuit that implements the TQD phase on a torus while detecting the presence of charge and flux excitations. At time t_0 all ancilla qubits, shown as hollow circles, are measured in either single-qubit X (labeled as m_X) or Z basis (labeled as m_Z) and then reset. Yellow, purple, and black-colored hyperedges at times t_2 and t_5 show the CCZ gates; each hyperedge ends on a triple of qubits of three different colors on which the physical CCZ gates act. The colors are chosen in correspondence with the paths in Fig. 5. This action is repeated on each square plaquette in a translationally invariant way. CNOT gates are shown at times t_1, t_3, t_4, and t_6 in blue and are depicted in the standard way.

intersections of membranes of three different types by evaluating this number in each 3-cell and summing over the entire spacetime.

This microscopic representation is defined on a 3-cellulation of spacetime with three $\mathbb{Z}_2 = \{0, 1\}$ variables at each edge, which can host three copies of the toric code path integral in Eq. (65), which we label in red, green, and blue. For the TQD phase, we add a nontrivial weight (or "twist") to the path integral using the cup product. Let a_r, a_g, and a_b denote the configurations of red, green, and blue toric code variables, respectively. For a given configuration, the expression

$$\prod_{3-\text{cells } v} (-1)^{(a_r \cup a_g \cup a_b)(v)} \tag{66}$$

counts the total parity of triple intersections between all three colors (see Refs. [9,90] and Appendix A in Ref. [78]). Then, the path integral can be written as

$$Z = \sum_{\substack{\text{closed membranes} \\ a_r, a_g, a_b}} \prod_v (-1)^{(a_r \cup a_g \cup a_b)(v)}. \tag{67}$$

Similar to the toric code above, we again consider a skewed cubic spacetime lattice as shown in Fig. 5. Note that the skew is not strictly needed, but it makes the resulting circuit more uniform and better parallelized. Then, for a given cube with coordinate v, we can evaluate the expression $(a_r \cup a_g \cup a_b)(v)$ following Fig. 5(a): In each cube, there are six different paths of edges from the vertex marked in black to the vertex marked in white, shown as black, purple, and yellow, dashed or solid. For each path p, we number the edges $(0_p, 1_p, 2_p)$. Then, we have

$$(a_r \cup a_g \cup a_b)(v) = \sum_p a_r(0_p) a_g(1_p) a_b(2_p). \tag{68}$$

Intuitively, formulas for the cup product can be obtained by shifting three sublattices slightly into a generic position with respect to another, such that the intersection between the membranes becomes unambiguous. Note that, in the example in Sec. III, the three toric codes are already defined on three separate lattices, such that the intersections are unambiguous. The vertices marked black and white in Fig. 5 correspond to a choice of a *branching structure* on the cube, that is, a choice of local coordinate system that is needed to define the shift direction. For further details about microscopic formulas for cup products, we refer the reader to Appendix A in Ref. [78], as well as Refs. [90,91].

Finally, we briefly explain the derivation of the circuit for the TQD phase (see Ref. [78] for a more in-depth discussion). The circuit is shown in Fig. 6 and is obtained by considering sections of the spacetime lattice at each time step, as illustrated in Fig. 5(b). The correspondence between the data and ancilla qubits and the path-integral variables is analogous to that for the toric code example in the previous subsection. There are three data qubits per edge (labeled by red, green, and blue colors) shown as a single dark dot at t_0 and three ancilla qubits per vertex and plaquette shown as white dots. If we exclude the steps t_2 and t_5, the resulting circuit implements precisely three decoupled toric code copies on the square lattice, where the step t_0 corresponds to single-qubit m_X or m_Z ancilla measurements. The additional steps t_2 and t_5 implement the nontrivial phases in the path integral of the twisted phase [Eq. (67)].

Figure 5(b) illustrates how steps t_2 and t_5 realize Eq. (67). The left-hand side shows the time step and the "active" path-integral variables that are represented by data and ancilla qubits at this point in time. The right-hand side shows the paths entering the cup product in Eq. (68) that are

supported only on these active variables. Note that the paths shown do not all correspond to the marked cube, but some also originate from the cube below. Each of these paths p corresponds to a multiplicative term of the form $(-1)^{a_r(0_p)a_g(1_p)a_b(2_p)}$ in the path integral. The presence or absence of a membrane at a given edge corresponds to the qubit being in $|0\rangle$ or $|1\rangle$ state, respectively. Therefore, the phase $(-1)^{a_r(0_p)a_g(1_p)a_b(2_p)}$ can be implemented by a $CCZ_{0_p,1_p,2_p}$ gate acting between red, green, and blue qubits at specified locations. Finally, each such CCZ gate for a single square plaquette is shown in Fig. 6 at corresponding time steps. Each gate is colored in correspondence to the path it was obtained from [i.e., as the path in Fig. 5(b) at the corresponding time]. The qubits that each of these gates couples are shown explicitly for clarity. Each square plaquette contains six paths for each period of circuit operation, three of which are realized at time step t_2 and the other three at time step t_5.

B. Full circuits for logical protocols

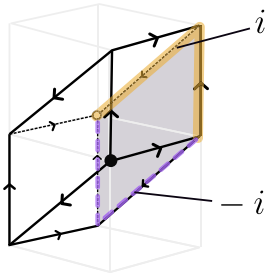
Having discussed the derivation of the circuit realizing the TQD phase, we are now ready to discuss the derivation of circuits for the protocols that implement nontrivial logical operations. These are the circuits corresponding to the global topologies presented in Sec. IV D. For this, we need to discuss how to implement the boundaries of the TQD phase and its domain walls with the toric code. These elements are then combined to realize a specific spacetime protocol. We explain this by considering an example of the $\overline{CZ}$ measurement protocol, which is an open boundary condition variant of the example in Sec. III.

We first discuss boundaries. For each copy of the toric code phase, there is only the rough ($\langle\rangle$) and the smooth ($\langle p\rangle$ or $\langle y\rangle$) boundary. There is also a "fold" boundary that joins two copies of the toric code ($\langle py\rangle$). A rough boundary is realized by defining a continuous boundary line on the primal lattice and removing the qubits on the "vacuum" side, including the boundary itself. The circuit with a boundary is obtained by removing the measurement and unitary operations that formerly involved the removed qubits. A smooth boundary is realized similarly, except that the boundary is now defined on the dual lattice. For the $\langle py\rangle$ boundary, we start with a pair of smooth boundaries instead and add a ZZ measurement that "forces" termination of a matching type of membrane on each boundary site whenever a membrane terminates there.

For the TQD phase, the only boundaries that appear in the protocols are of $\langle r, g\rangle$ and $\langle rg, rb\rangle$ type (or the analogous boundaries under color permutation). The $\langle r, g\rangle$ boundary is a smooth boundary on the red and green sublattices and rough boundary on the blue sublattice. The $\langle rg, rb\rangle$ boundary is realized by first considering a smooth boundary of each color and then adding a ZZZ measurement on each boundary site that "glues" the termination lines of any pair of membranes. In addition, the path integral includes a complex-phase weight for crossings of termination lines as shown in Eq. (F10) in Appendix F. In the microscopic model, this additional weight can be implemented via a cup product as

$$\prod_f i^{(a_r \cup a_b)(f)}. \tag{69}$$

Here, f runs over all boundary faces, and a_r and a_b are the configurations of green and blue path-integral variables, respectively, assigned to the boundary edges. The microscopic expression for this cup product is similar to the one for the bulk in Fig. 5(a): We again consider the two different paths p_1 and p_2 of edges inside the boundary face that go from one vertex to an opposite vertex, as shown in yellow and purple in the following picture for the boundary face shaded gray:



(70)

The cup product on f is then given by

$$(a_r \cup a_b)(f) = a_r(0_{p1})a_b(1_{p1}) - a_r(0_{p2})a_b(1_{p2});$$

that is, the purple path p_2 gets an extra factor of -1. Note that -1 signs are also present in the cup-product formula on the cube in Eq. (68), but they do not matter there since $(-1)^{-a} = (-1)^a$.

Thus, the additional path-integral weights are of the form $i^{a_r(0_{p1})a_b(1_{p1})}$ and can be implemented in the circuit by adding physical CS and $CS^\dagger$ gates at steps t_2 and t_5, acting on the green and blue qubits which represent the corresponding spacetime edges.

Next, we discuss the domain walls between the toric code copies and TQD phases that appear in the protocols. These are the $\langle rp, by\rangle$ domain wall and its color permutations. Two different orientations of this domain wall have appeared in the global topologies above. One is timelike, meaning that it occurs at a fixed instant of time which corresponds to globally switching from one topological phase to the other. The second orientation is tilted, which corresponds to a domain wall that moves in space as time progresses. This happens, for example, in the $\overline{CCZ}$ and $\bar{T}$-gate protocols. We first address the timelike $\langle rp, by\rangle$ domain wall between a pair of toric codes and the TQD phase. In spacetime, this corresponds to the purple and yellow lattices of the toric codes becoming red and blue,

which is simply a relabeling of colors. For this reason, in many examples we use the color assignment from the TQD phase to label the toric codes as well. For the third (green) color, the domain wall is the same as the rough boundary. To realize it in the circuit, we simply prepare the green data and plaquette ancilla qubits in the $|0\rangle$ state and the green vertex qubits in the $|+\rangle$ state at the end of the circuit realizing the toric code phases and continue to run the circuit for the TQD phase. Similarly, to realize such a domain wall in reverse, between the TQD phase and two copies of the toric code phase, we simply measure the green data qubits in the Z basis and perform the usual measurements of the green ancilla qubits. Next, we discuss the realization of the tilted domain wall. To move such a domain wall from the toric codes phase to the TQD phase, we initialize a new set of green data qubits of the third color in state $|0\rangle$ in a thin slice and implement the toric codes circuit on one side of the new domain wall and the TQD circuit on the other side. Similarly, to move such a domain wall from the TQD to the side of the toric codes phase side, we instead measure out green data qubits in Z basis.

We now discuss the corners. The majority of corners appearing in our protocols are not associated with adding any new weights in the path integral. Microscopically, such corners can be obtained by naively letting two types of microscopic boundary conditions meet at the corner. An exception is the "ω" boundary between the $\langle rg, rb\rangle$ and the $\langle g, b\rangle$ boundaries, which requires the addition of $\omega = e^{-i\pi/4}$ phase weights. These ω weights are implemented in the circuit by additional T gates. Apart from these weights, a straightforward blending between the two boundary conditions produces the correct circuit for this type of corner.

This completes our summary of how to derive circuit implementations for all types of spacetime defects that appear in the global geometries presented in Sec. IV D. When appropriately combined together, this produces circuits implementing the associated logical operations by design. The measurement outcomes of vertex ancilla qubits are stored, while those of plaquette ancillas are used for the operation of the just-in-time decoder, discussed in Sec. VII.

Finally, we present an example of the circuit for the logical $\overline{CZ}$ measurement introduced in Sec. IV D 3, where the bulk of the TQD phase is implemented as in Fig. 7. The entire lattice is shown in Fig. 7(a), where every black dot represents a triple of an r, g, and b qubit, and every colored dot represents an individual qubit of that color. First, the syndrome extraction circuit is run for the r and b copies of the toric codes phase (we have relabeled p to r and y to b) with open boundary conditions, which implement rotated surface code patches with distance 4. This circuit is analogous to the one in Fig. 6 with steps t_2 and t_5 omitted, except that the application of CNOT gates is appropriately truncated at the boundary as shown for one time step in Fig. 7(c). Then, the domain wall from two copies of the toric code to the TQD is realized by initializing the green qubits as shown in Fig. 7(b) (all locations involved in the initialization are highlighted in green). Next, we run the circuit realizing the TQD phase with boundary conditions shown in Fig. 7(a) according to the appropriately truncated circuit from Fig. 6 for a number of rounds. Finally, we switch from the TQD phase to the pair of surface codes by measuring out the green vertex qubits in the X basis and the green data and plaquette qubits in the Z basis, similar to Fig. 7(b). Overall, this performs a logical $\overline{CZ}$ measurement between the logical qubits of the red and blue copy of the surface code.

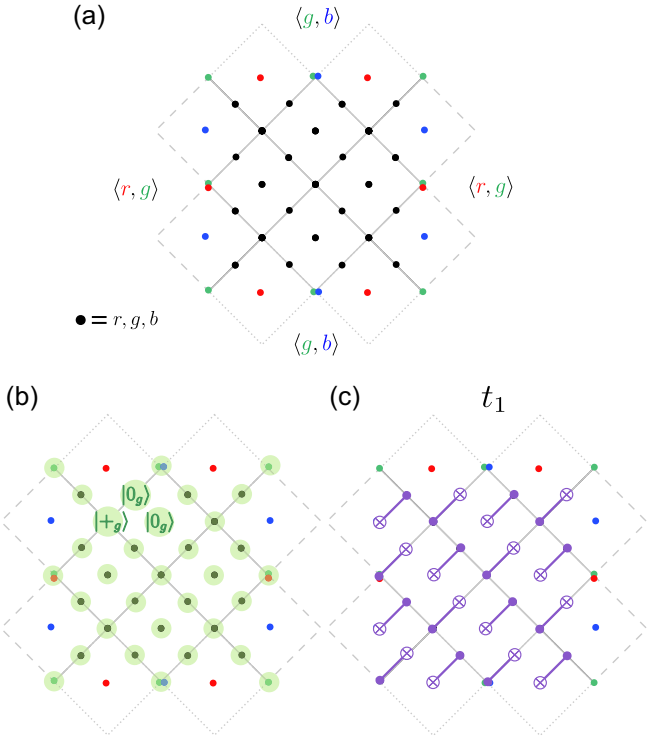


FIG. 7. (a) Qubit layout for the logical $\overline{CZ}$ measurement on a pair of surface codes of distance 4, which matches the transient TQD phase and its boundary conditions shown in Eq. (43). (b) The locations of green qubits that are initialized or measured out to realize the domain wall between the TQD and the toric codes phases. (c) An example demonstrating how the circuit shown in Fig. 6 without boundaries is altered near the boundaries.

C. Other microscopic circuit examples

1. *Transversal counting of triple intersections*

Defining the three toric code path integrals on different 3-cellulations, it is possible to obtain a circuit where CCZ operations act on triples of qubits at the same location. For example, three such superimposed 3-cellulations are described in Ref. [92]. In this case, while the three cellulations are different, the centers of the edges of each cellulation coincide; in the end, we have three qubits per

site that are labeled three different colors. In this case, there is one triple intersection weight at every edge center c which takes the simple form

$$(-1)^{a_r(c)a_g(c)a_b(c)}. \tag{71}$$

Using the approach described in this section, one can turn this path integral into a circuit, which results in one where the CCZ gates act transversally. The associated stabilizer group appears in Appendix G. Additionally, the protocol in Ref. [23] where the original $\overline{CCZ}$ protocol was introduced (where the TQD phase is featured only implicitly) can be viewed as a circuit realizing this path integral.

2. Microscopic lattice from Sec. III and its variations

It is possible to also relate the protocol presented in Sec. III to the TQD path integral. Namely, the circuit of alternating $+1$-postselected plaquette and vertex measurements yields a TQD path integral on three superimposed spacetime lattices. The plaquette stabilizers implement the parity constraints at the faces of the spacetime lattices, whereas the vertex terms implement the triple-intersection weights at the volumes. These volumes are diamond shaped, with a "bottom" and "top" half that both consist of the triangles surrounding the according vertex in the spatial triangulation. If we replace projectors with measurements, these measurements, in addition, detect charge and flux defects. Reinterpreting this model through the lens of the path-integral framework is useful, because the path-integral approach makes the fault tolerance of the protocol much more apparent and gives systematic ways to construct full logical circuits, which requires deriving boundaries, domain walls, and corners appearing in them. This can be used to derive circuits for all the logical protocols in Sec. IV, where the TQD phase is implemented via the model from Sec. III.

We now turn to a similar, but simpler, version of the Clifford stabilizer model from Sec. III. In Fig. 8(a), we show its stabilizers on the same lattice; the only difference to the lattice in Fig. 3 is that the Clifford circuit attached to the vertex stabilizers now consists of two CZ gates only. In Fig. 8(b), we show why the new vertex stabilizer counts the same loop statistics. For this, we make use of the dual lattice, which is depicted along with the original vertex stabilizer in Fig. 22. By slightly shifting the vertices of the red and green dual lattice with respect to one another, we see that there are precisely two locations with possible red-green intersections inside a given blue loop. For each location, a CZ gate detects the presence of such an intersection. This model has the same properties as the one in Sec. III. However, the stabilizers have weights 3 and 8 as opposed to 3 and 12, respectively, which could be beneficial for practical implementations.

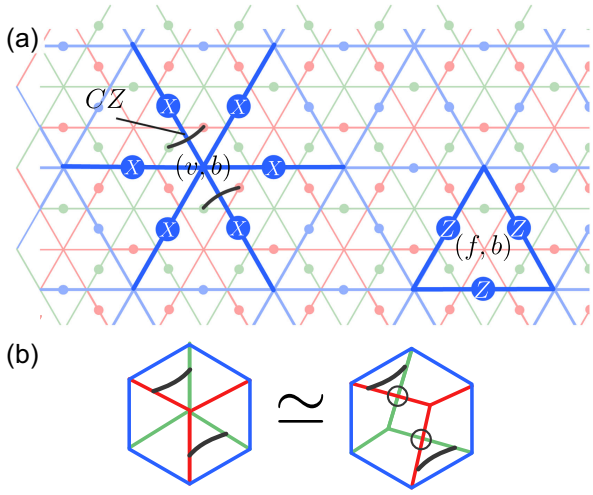


FIG. 8. (a) A version of Clifford vertex (left) and Pauli plaquette (right) stabilizers on the blue sublattice of the twisted quantum double on three triangular lattices. The stabilizers on the red and green sublattices are defined analogously. (b) Detection of red-green loop crossings inside a blue loop on a dual lattice and CZ operators in the associated vertex stabilizer. The possible intersections of red and green loops are resolved on the right and are circled.

VI. THE NON-ABELIAN PHASE AND 3D STABILIZER CODES

This section also highlights the connection between twisted quantum doubles of an Abelian group in 2D and an Abelian 3D gauge theory, both conceptually and showing several lattice examples.

Previous constructions of 2D protocols for non-Clifford gates relied on specific 3D codes with transversal non-Clifford logic gates [22,23]. In this section, we explain that these protocols in fact realize the same TQD phase and are, hence, special instances of protocols constructed in Sec. IV. Specifically, we show that, while performing the non-Clifford gate, the protocols based on a dimensional jump realize states that can be associated with the TQD phase as an intermediate step. We start by discussing a general connection between $(2+1)$D twisted quantum double models and topological boundary conditions of an associated $(3+1)$D untwisted topological gauge theory. For concreteness, we consider the gauge theory realized by the 3D color code, a $\mathbb{Z}_2^{\times 3}$ gauge theory. In a microscopic model, physical T rotations and local Pauli measurements on a slab of the 3D color-code model can be used to prepare a 2D state in the TQD phase [in fact, the resulting *color path integrals* are also described by a cohomology theory that is equivalent to, but not the same, as cellular (co)homology [93]]. In Appendix G, we present an analogous protocol based on three copies of the 3D toric code and physical controlled-controlled-Z gates. This leads to a different lattice realization of the same TQD phase. In general, this procedure can be performed for any

microscopic Pauli stabilizer realization of the $\mathbb{Z}_2^{\times_3}$ phase in three dimensions.

The arguments in this section are general enough that they should extend to a 3D Abelian topological gauge theory with a gauge group G^m (where G is Abelian and $m = 2, 3$) and a non-Abelian phase of G^m with a 3-cocycle twist; however, for simplicity and lack of unnecessary generality, we focus on the case of $\mathbb{Z}_2^{\times 3}$.

A. (2+1)D circuits from (3+0)D measurement-based protocols

References [22,23] introduced two-dimensional protocols that fault-tolerantly realize non-Clifford gates via code switching between a 3D code, that admits a transversal non-Clifford gate, and a 2D code. To ensure the resulting protocols are fault tolerant, the *dimensional jumping* procedure [94,95] that transforms between the two codes must be performed repeatedly over $O(d)$ time steps, where d is the spatial code distance of the 2D code if only a thin slab of 3D code is used. In this section, and in Appendix G, we show that the non-Abelian TQD model is, in fact, realized as an intermediate state during both proposals. This illustrates the equivalence between these examples and the framework we have developed to perform non-Clifford gates using the TQD as an intermediate phase.

Topological stabilizer codes in three dimensions can be prepared in constant time using single-shot error correction [95,96]. This property can be used to perform single-shot magic-state preparation on the two-dimensional boundaries of the three-dimensional code. For example, we summarize below the procedure to prepare a CCZ magic state in constant time using three copies of the three-dimensional toric code.

(1) Prepare the $|{+}{+}{+}\rangle$ logical state of three copies of 3D toric codes in a cube with open boundary conditions such that they admit a transversal $\overline{CCZ}$ gate [92]. For this, first initialize the qubits on the lattice that hosts three copies of the 3D toric code in the $|+\rangle^n$ state.
(2) Measure the Z (face) stabilizers of the 3D toric code copies and correct for errors. This is done fault tolerantly with only a single round of measurements using single-shot error correction [96]. This results in a logical $|{+}{+}{+}\rangle$ state.
(3) Apply the constant-depth transversal CCZ gate between the three 3D toric code copies. This prepares a logical magic state $CCZ|{+}{+}{+}\rangle$ of the total 3D code.
(4) For each of the copies of the 3D toric code, measure all qubits in the X basis, except those on designated boundaries such that a planar surface code remains. We can use these measurements to reliably correct Z errors in the resulting two-dimensional code such that the residual error is small and local [95].

The constant depth execution of the protocol means that the example is readily expressed as a postselected three-dimensional tensor network. Furthermore, we can interpret the evaluation of this tensor network as summing over all "qubit configurations" on internal edges that fulfill the constraints imposed by the local circuit elements and adding weights, depending on the local configuration. It can, therefore, be interpreted as a *path integral*, in the sense of Eq. (22). The fact that we obtain a scalable family of such networks with macroscopic fault distance renders this tensor network *topological*. More specifically, any error that is supported only on a 3D ball-like region in spacetime does not affect the logical action of the postselected network [78]. Using local equivalence rules of the tensors involved, we can interpret the tensor network either as a $(3+0)$D protocol or a $(2+1)$D protocol. This can be understood as trading one spatial dimension in the 3D code involved in the above protocol for a time direction. Moreover, the tensor network can be seen as representing a $(2+1)$D topological phase that is equivalent to the phase of the non-Abelian TQD.

In the remainder of this section, we explore the equivalence between 3D codes and the non-Abelian TQD further, invoking a 2D perspective. We show how a 3D code that admits a transversal non-Clifford gate in the bulk can be used to prepare a non-Abelian state on its 2D boundary. These types of states appear as intermediate states in the protocols from Refs. [22,23], hence explaining that these protocols, in fact, involve a non-Abelian phase.

B. Overview: The non-Abelian TQD from a 3D Abelian code

Here, we discuss the relationship between the non-Abelian TQD model in two dimensions and the 3D color code. The 3D color code, like three copies of the 3D toric code [88], can be viewed as realizing a $\mathbb{Z}_2^{\times 3} = \mathbb{Z}_2 \times \mathbb{Z}_2 \times \mathbb{Z}_2$ topological gauge theory. The logical operators in these codes are related to the nontrivial winding of fluxes and charges of the underlying gauge theory. Symmetries among these topological defects can be implemented by locality-preserving unitaries and correspond to fault-tolerant logical gates of the associated codes. The $\mathbb{Z}_2^{\times 3}$ gauge theory in three spatial dimensions admits a global symmetry that implements a logical non-Clifford gate. For the 3D color code, this symmetry is realized by a transversal $T^{(\dagger)}$ gate, and in three copies of toric or surface code it is realized by a transversal CCZ gate acting between the individual codes [92,97].

To explain the relationship between a three-dimensional slab and a two-dimensional model, we consider a slab of 3D code whose two-dimensional section is a torus, i.e., a manifold with $\mathbb{T}^2 \times [0, 1]$ topology. The slab is shown in Fig. 9, and it has two boundaries which we label *top* and *bottom*. We start with smooth boundary conditions on both boundaries. This results in a model that is equivalent to

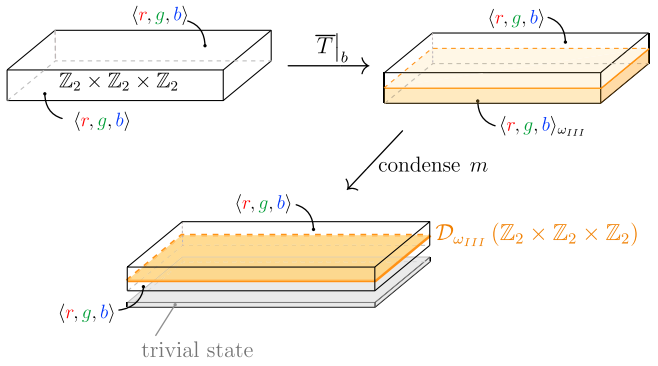


FIG. 9. The topological quantum field theory picture for the initialization of a non-Abelian TQD from a $(3+1)$-dimensional topological $\mathbb{Z}_2^{\times 3}$ gauge theory defined on a slab with appropriate boundary conditions. The outcome is shown at the bottom, displaying a thin slab with the defect configuration that leads to a 2D state equivalent to the TQD model. The top and bottom boundaries are smooth; i.e., flux membranes of all three colors can condense on them (labeled $\langle r, g, b\rangle$), and there is a domain wall defect associated to a non-Clifford symmetry of the bulk indicated in yellow. In the limit where the thickness of the slab is kept constant but the other two directions are scaled, this becomes equivalent to a two-dimensional state of the non-Abelian TQD. Proceeding from the top left to the bottom, we show how to create this configuration by transversal gates and measurements: we start with a slab of a 3D code with two opposing smooth boundaries. Applying the non-Clifford symmetry $\bar{T}|_b$ of the model to the bottom half of the slab creates a domain wall and also "twists" the bottom boundary condition, which now becomes $\langle r, g, b\rangle_{\omega_{\text{III}}}$. Finally, we enforce smooth boundary condition again on the bottom boundary. In the stabilizer models that realize this topological theory, this can be achieved by local measurements and adaptive corrections.

three copies of the 2D toric code when the third dimension is compactified using these boundary conditions. Applying the non-Clifford symmetry to the lower half of the state, including the bottom boundary, creates a two-dimensional domain wall defect between the two boundaries [8,18]. At the same time, it "twists" boundary conditions of the bottom boundary. As a final step, we enforce smooth boundary conditions on the bottom boundary by measuring out the bottom qubits in a suitable single-qubit basis. This condenses bare fluxes below the domain wall defect. This creates a model that, when regarded as a two-dimensional model, is equivalent to a non-Abelian TQD.

We now turn the above procedure into an explicit microscopic protocol that prepares a non-Abelian code associated with the TQD. The same procedure occurs in the bulk during the $(2+1)$D protocol obtained from a $(3+0)$D circuit, as we discussed above. The example presented in this section is based on the 3D color code. We present another example based on three copies of the 3D toric code in Appendix G.

The protocol is summarized in the following steps.

(1) Initialize a slab of 3D code that admits a transversal non-Clifford gate in the bulk, with appropriate boundary conditions, "smooth" on both the top and bottom boundary.
(2) Apply the non-Clifford gate to the lower half of the slab including the bottom boundary.
(3) Measure out the qubits along the bottom boundary in a suitable single-qubit basis, enforcing smooth boundary conditions again.

These steps are depicted in Fig. 9.

After these steps, a layer of qubits around the top boundary realizes a two-dimensional code that, as we show, is equivalent to the non-Abelian TQD considered in this paper.

If step 2 (applying the non-Clifford gate in half of the slab) is skipped, we obtain an untwisted version of the 2D model which corresponds to three copies of the 2D toric code. We start by considering this example, as it helps to determine the topological phase of the twisted model below.

C. Non-Abelian TQD preparation from the 3D color code

We define the three-dimensional color code [97] with qubits on the vertices of a cubic lattice. The color code has two types of Pauli stabilizers associated to the four-colorable brickwork cells shown in Fig. 10: volume stabilizers S_c^X and face operators S_f^Z. Volume stabilizers are the product of Pauli-X operators acting on the vertices of cell c, i.e., $S_c^X = \prod_{v\in c} X_v$. Faces f interface pairs of adjacent cells, $f = c' \cap c$, and are associated with pairs of colors. We define $S_f^Z = \prod_{v\in f} Z_v$.

The 3D color code admits a transversal non-Clifford gate of the form

$$\bar{T} = \prod_{\text{even } v} T_v \prod_{\text{odd } v} T_v^\dagger, \tag{72}$$

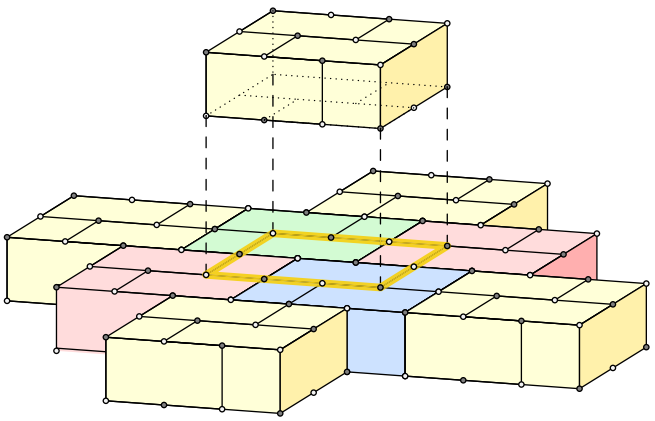


FIG. 10. Four-colorable 3-cells of the color code brickwork lattice organized with qubits on the vertices of a cubic lattice.

where $T_v = \exp(i\pi Z_v/4)$ and $v = (x, y, z)$ is even (odd), if their canonical coordinates on the cubic lattice sum to an even (odd) number, i.e., $x + y + z$ is even (odd) [19,98]. As a consequence, the code also admits Clifford stabilizers obtained from conjugating the Pauli stabilizers with the non-Clifford logical. First, we define the operator

$$W_v = \bar{T} X_v \bar{T}^\dagger, \tag{73}$$

which is equal to $e^{i\pi/4} X_v S^\dagger$ if v is on the even sublattice and $e^{-i\pi/4} X_v S$ for the odd sublattice. Then, we define

$$S_c^W = \bar{T} S_c^X \bar{T}^\dagger. \tag{74}$$

They obey the group commutation relation

$$[S_t^W, S_b^X] = S^Z_{f=t\cap b}, \tag{75}$$

showing that they commute on the code space. Outside of the "flux-free" subspace, defined by $S_f^Z = +1$, the (Clifford) stabilizers do not necessarily commute.

1. Initializing an Abelian 2D state from 3D color code

We start with a 3D slab of thickness 1 with all-smooth boundary conditions (i.e., flux membranes of all three colors can terminate on the top and bottom boundaries). A slab with such boundary conditions is supported on a single layer of bricks as shown in Fig. 10, and the qubits on the bottom and top of the layer of bricks are naturally split into the *bottom layer b* and the qubits on the *top layer t*. This model can be obtained by considering a 3D color code supported in infinite space and then measuring out the qubits everywhere, apart from one layer of bricks, in the single-qubit X basis.

First, consider what happens if we "collapse" the state in the three-dimensional slab onto the top boundary without introducing the non-Clifford domain wall defect. For this, we measure out the bottom layer in the single-qubit Pauli-X basis, which leaves us with a state supported on the top layer only. The stabilizers for the two-dimensional state that we obtain this way are shown in Fig. 11.

The postmeasurement state is in the phase of three copies of the 2D toric code. To see this, we explicitly construct the basis of excitations for this model and verify their statistics. We assign colors to the three copies r, g, and b. The excitations of a single toric code are known as electric charges e and magnetic fluxes m. We assign the e_u excitations a single color that corresponds to their layer $u = r, g, b$. Here, an excitation of a given color lies at a violated brick stabilizer. In Fig. 11, we show an operator that creates a pair of e_r charges. A yellow electric charge $e_y = e_r \times e_g \times e_b$ is the fusion product of the electric charges.

Magnetic fluxes are assigned a color pair m_{uv}. Electric charges e_w braid nontrivially with magnetic charges m_{uv} if either $w = u$ or $w = v$. We can associate the magnetic charge uy to the u-colored toric code, since m_{uy} braids nontrivially with the electric charge of the corresponding toric code e_u but trivially with electric charges e_v with $v \neq u, y$ (recalling that e_y is the fusion product of the electric charges of the three toric code layers). Magnetic charges have a nontrivial fusion product $m_{uv} = m_{uw} \times m_{vw}$ where all colors u, v, and w are distinct.

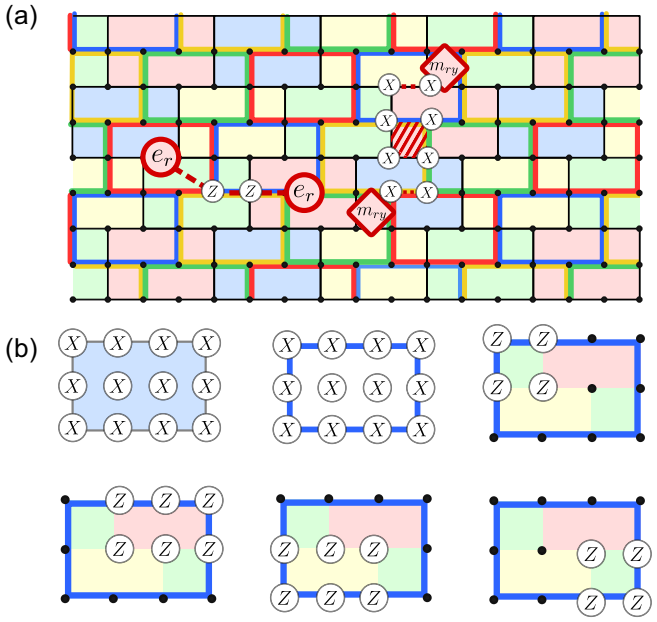


FIG. 11. (a) The four-colorable (r, g, b, and y) brickwork lattice obtained by "collapsing" a thin three-dimensional slab of 3D color code onto its top boundary. Figure 10 shows two layers of 3-cells in 3D whose intersections with one of the boundaries give two types of bricks that we label "bottom" (shaded in solid colors) and "top" (shown with bold colored outlines) that come in four colors. Qubits are shown as black dots and form a square lattice. The faces of this lattice are weight-6 and weight-4 rectangles that are colored in pairs of colors corresponding to the intersecting top and bottom bricks. The cell and face stabilizers are shown in (b). Additionally, (a) shows red charge and flux excitations and their hopping operators.

2. Initializing a non-Abelian state from 3D color code

For this particular microscopic model, we are free to choose whether to apply the $\bar{T}$ symmetry to the top or bottom half in our non-Abelian phase preparation protocol before measuring out the bottom qubits. We choose to apply it to the bottom half of the slab, denoting it $\bar{T}|_b$. It acts nontrivially on the bottom layer of qubits before we measure them out in the X basis. The resulting effect on the top qubits is the same as if we instead measured the bottom qubits directly in the basis defined by W_v.

Assuming that all measurement outcomes are $+1$, the postmeasurement stabilizer group follows directly from the stabilizer group of the thin 3D code. For other measurement

outcomes, the stabilizers of the state differ only by their signs. The Z stabilizers that overlap the bottom qubits are removed from the stabilizer group. The two-dimensional state after measurement is stabilized by S_f^Z operators on faces f supported on the top boundary. To determine the operators that stabilize the state, consider first how the volume operators of a thin slab of the 3D color code are transformed by $\bar{T}|_b$. Conjugating the S_c^X and S_c^W stabilizers transforms them to "mixed" Pauli and Clifford stabilizers

$$S_c^X \overset{\bar{T}|_b}{\mapsto} \tilde{S}_c^X = \prod_{v\in c\cap b} W_v \prod_{v\in c\cap t} X_v, \tag{76a}$$

$$S_c^W \overset{\bar{T}|_b}{\mapsto} \tilde{S}_c^W = \prod_{v\in c\cap b} X_v \prod_{v\in c\cap t} W_v. \tag{76b}$$

The measurement in the X_q basis commutes only with $\tilde{S}_c^W$ and the X-type volume stabilizers of the 3D color code that are supported on the top boundary only. Therefore, the postmeasurement state is supported by the latter as well as the volume stabilizers of type $\prod_{v\in c\cap t} W_q$.

The resulting stabilizers are shown in Fig. 12. The lattice shows two types of bricks: "top-layer" bricks and "bottom-layer" bricks where, in the figure, top-layer bricks are outlined in bold and the bottom-layer bricks are shaded. One can see that the lattice of top-layer bricks is a spatial translation from the lattice of bottom-layer bricks on the square lattice of vertices. The top-layer bricks host the X-type top boundary cell stabilizers of the 3D color code; the bottom-layer bricks support the volume stabilizers of the $\prod_{v\in c\cap t} W_q$ type. Each face supports a Z-type stabilizer.

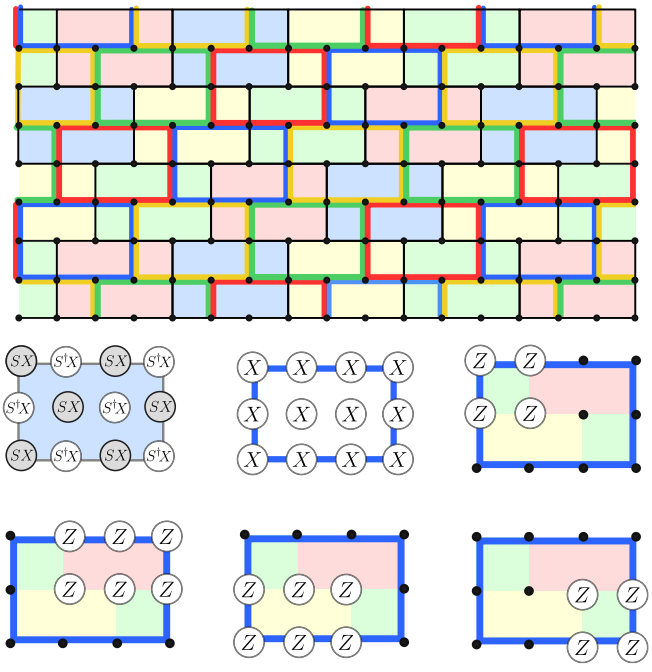


FIG. 12. Noncommuting Clifford stabilizers for the color code-based microscopic lattice realization of the TQD phase. Below the figure, we show an example of blue brick stabilizers S_b^X and S_b^W, as well as some of the Pauli-Z face stabilizers.

3. The phase of the non-Abelian code

We now identify the non-Abelian phase of the Clifford stabilizer model obtained in the previous section by viewing it as a model obtained from gauging a particular anyon-permuting $\mathbb{Z}_2$ symmetry in two copies of the toric code. This shows that the model hosts non-Abelian anyons precisely equivalent to those of the type-III twisted quantum double of $\mathbb{Z}_2^{\otimes 3}$; see Appendix C 2.

First, we construct a Pauli model with a suitable $\mathbb{Z}_2$ symmetry. We start with the two-dimensional Abelian version of the model discussed in Sec. VI C 1, which is equivalent to three copies of the toric code. Then, we condense a single type of boson [99] to obtain a model equivalent to two copies of the toric code. We choose to condense the e_g anyons by projecting onto the $+1$ eigenspace of $Z\otimes Z$ operators on green edges—edges that connect green bricks. The stabilizer group for the resulting state is then obtained by adding $Z\otimes Z$ hopping operators to it and removing the operators that do not commute with them. The remaining operators are generated by the Z stabilizers of the uncondensed model and all X stabilizers except for the green cells, since these anticommute with the enforced hopping operators. We illustrate the condensed model in Fig. 13.

The condensed model has a global Clifford symmetry of the form

$$\overline{XS} = \prod_{c\ \text{green}} S_c, \tag{77}$$

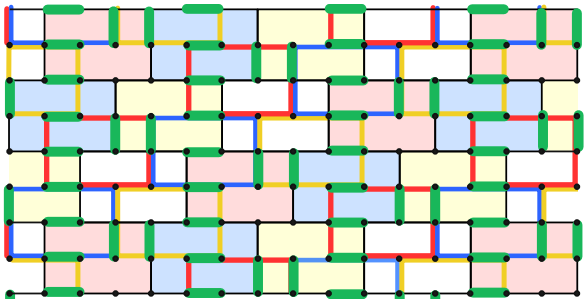


FIG. 13. The lattice supporting the stabilizer code is locally equivalent to the color-code model. For each red, blue, and yellow colored brick, it has the same Pauli-X volume stabilizers as in the uncondensed model shown in Fig. 11. Similarly, it has all the Pauli-Z face stabilizers of the uncondensed model. The green edges shown in the picture support $Z\otimes Z$ stabilizers. A transversal implementation of the logical S gate is achieved by applying $S = i^{|1\rangle\langle 1|}$ to all the qubits on the even sublattice and $S^\dagger$ to all the qubits on the odd sublattice, which is the same as the operator shown in Eq. (77).

where S_c are both the Pauli and Clifford green volume stabilizers of the uncondensed model. To see that this symmetry preserves the code space, we recall that if two qubits are stabilized by $Z \otimes Z$, they are also stabilized by any operator of the form $D \otimes D^\dagger$, where D is a diagonal single-qubit unitary. We use $D = S = i^{|1\rangle\langle 1|}$ (or its conjugate on the odd sublattice). In addition, one can explicitly verify that $\overline{XS}$ commutes with the X stabilizers up to a Z stabilizer. Taken together, we find that $\overline{XS}$ preserves the code space and acts as a nontrivial logical Clifford operator of the order of 2 on the condensed code and, hence, realizes a nontrivial $\mathbb{Z}_2$ symmetry of the condensed code.

Following the general procedure of gauging Abelian on-site symmetries (which we review in more detail in Appendix C 3 and C 4), we find that this $\mathbb{Z}_2$ symmetry is gauged by projecting the code space onto the $+1$ eigenspace of each of the (Clifford) stabilizers $\mathcal{S}_c$ associated to the green bricks. Alternatively, we can measure these operators after ensuring that all Z stabilizers have $+1$ eigenvalues and then applying a suitable correction. By construction, this gauging procedure produces a state that hosts non-Abelian anyons characterized by the twisted quantum double, as explained in Appendix C.

D. Linking charges in terms of non-Abelian statistics

Having established this connection between 3D codes and the TQD in spacetime, it becomes instructive to investigate the common features that emerge in these two perspectives. In fact, the triple intersection of membranes in spacetime that is instrumental for our protocols (see Sec. IV) is analogous to the triple intersection resulting in non-Clifford gates in 3D topological codes [35,98,100,101]. Let us explore the analogy more closely at the level of error propagation in 3D codes that occurs upon the application of the non-Clifford gate [100].

Applying a transversal non-Clifford gate to a 3D code with a flux loop excitation gives rise to a superposition of electric charges running along the flux loop [100,102]. Because of this, the *linking charge* phenomenon can occur in such 3D codes [100,103], wherein two nontrivially interlinked flux loops of appropriate color must exchange an odd number of charges upon the application of a transversal gate.

There is a close connection between 3D codes and $(2+1)$D phases, and we find that, indeed, the linking charge phenomenon is mirrored in the anyon data of the non-Abelian TQD phase. The flux worldlines measured in $(2+1)$D can be understood as certain non-Abelian anyon worldlines on which charge worldlines can terminate. Note that the flux measurement in our protocols is not a measurement of *simple* anyons, but observing a -1 outcome projects onto a sector of the Hilbert space that hosts a combination of non-Abelian anyons.

On this basis, the fact that the total charge parity among linked flux anyons is fixed when the non-Clifford gate is applied can be described from the fusion and anyon data of the non-Abelian anyon theory. From the $(2+1)$D perspective, the charge exchange between linked flux anyon worldlines is related to the braiding data of the TQD anyon model: The S-matrix entry for two non-Abelian flux anyons of different types (see Appendix A in Ref. [56]), corresponding to two linked flux loops that are not connected by any charge strings, is 0. As a consequence, if we braid two fluxes, they must exchange an odd number of Abelian charge anyons during the process. In Fig. 14, we show an explicit example demonstrating the analogy between the charge excitations in the 3D code and a $(2+1)$-dimensional braiding diagram of the non-Abelian anyons in the TQD.

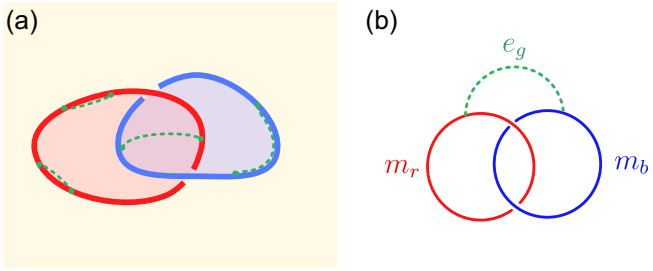


FIG. 14. (a) Linking charge phenomenon as defined in Ref. [100] in 3D code equivalent to three copies of the toric code or the color code. The red and blue flux loops are linked. When the transversal non-Clifford gate is applied to the three-dimensional volume (depicted by the yellow shading), a superposition of charges of the complementary colors along each flux loop is generated. Furthermore, an odd number of charges of the third (green) color are exchanged between the linked flux loops with certainty. This is analogous to the anyons of the non-Abelian TQD shown in (b), where braiding red and blue non-Abelian anyons m_r and m_b exchanges a green charge e_g.

VII. FAULT TOLERANCE AND JUST-IN-TIME DECODING

In this section, we discuss how to ensure that the protocols are fault tolerant by introducing just-in-time decoding and proving the existence of a threshold under the general local stochastic noise model. Our analytical proof applies to an RG version of the just-in-time decoding algorithm (Algorithm 1); we then discuss end-to-end decoding near the boundaries and domain walls (Algorithm 2) and a minimum-weight matching version of the just-in-time decoding (Algorithm 3), which we conjecture achieves distance of the code, i.e., logical error rate suppression that is exponential in the distance of the protocol. All of the protocols that we discuss are efficient (polynomial time) and reasonably practical.

A. Informal explanation

The measurements in the circuit that implements the TQD phase are divided into two classes, one of them

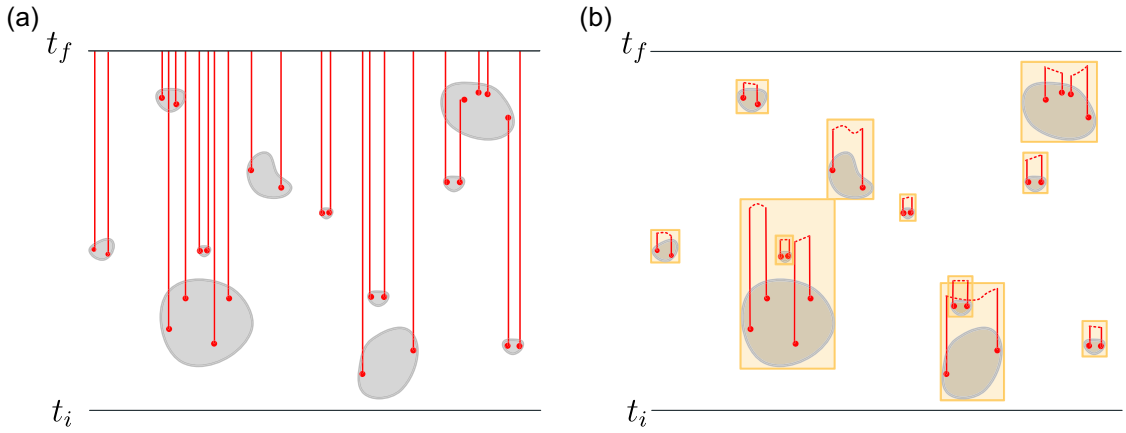


FIG. 15. (a) Schematic illustration of the worldlines of flux defects (red lines) created due to noise (contained in gray regions) in the circuit implementation of the non-Abelian TQD, not to scale. (b) The same spacetime history when operating with the just-in-time decoder, which pairs up the defects as soon as the correction can be deduced reliably. Each error cluster together with the associated flux worldlines is contained in some box (in yellow), and these boxes form a sparse distribution.

corresponding to the charge worldlines (which are associated with those of the Abelian charge anyons) and the other corresponding to the flux worldlines. Both charges and fluxes come in red, green, and blue colors. For simplicity, in our initial discussion we assume that the syndrome-extraction circuit has the property that, starting from a noise-free initial state, no -1 measurement outcomes would appear in the absence of the noise. In addition, if we start in a state with some distribution of fluxes and run a noiseless circuit, the flux worldlines continue indefinitely in the time direction (unless error correction is performed). All examples considered in Sec. V have this property, but it does not apply to $(3+0)$D, measurement-based quantum computing (MBQC)-like [94] and Floquet-like circuits [77,104,105]. While the decoding strategy and the fault tolerance proof remain the same for these other kinds of protocols, they possess an extra element of complexity. We defer a discussion of decoding such protocols to Sec. VII D.

A subtle point is that configurations of fluxes in our circuits do not correspond to specific configurations of non-Abelian excitations. Instead, a state containing some configuration of fluxes contains some superposition of non-Abelian excitations. We focus specifically on correcting flux defects (as opposed to non-Abelian anyons themselves) in our decoding strategy, as these describe more naturally what happens in the circuits.

The just-in-time decoding in the flux sector is motivated as follows. If we simply collect the measurement data for an extended period of time while in the TQD phase, assuming local stochastic noise as well as measurement errors, a schematic picture of a typical scenario is shown in Fig. 15(a). Because of the noise (occurring in gray regions), fluxes (shown in red) appear in the spacetime picture. Their worldlines continue in a timelike direction (shown by vertical red lines) and continue indefinitely in the absence of correction or unless new noise occurs in their path. After a long time (e.g., by t_f), many flux worldlines accumulate. For Abelian models, this would not be a problem: By considering the spacetime history of measurements, we determine the most-likely homology class of the flux worldlines in the resulting state and apply the correction at the end; only the resulting homology class matters. This is because in Abelian models all defect worldlines are freely deformable. However, in the non-Abelian TQD this is not the case. On a flux worldline of a given color, the charge worldlines of two other colors can freely terminate (see Sec. IV B 2). Therefore, in the presence of a large density of fluxes (which happens if we simply leave the fluxes be without the corrections), we also end up with an arbitrary configuration of charge worldlines in a large spacetime region, which is no longer correctable.

If we instead apply corrections *just in time* (see Fig. 15)—i.e., we attempt to remove the flux worldlines as we carry out the protocol—it is possible to retain fault tolerance. Such a just-in-time decoding algorithm uses the available measurement history up to the current time and applies the best correction possible based on available information. As a result, the effect of the noise is bounded by a set of spacetime boxes of a certain size, each proportional to the size of the noise cluster. This is shown by yellow boxes in Fig. 15(b). Because of this containment, the state outside of these boxes corresponds to the noiseless TQD phase. The protocol fails if the flux configurations are not contained in sufficiently isolated boxes, which, as we show below, happens only at a large enough noise rate, i.e., above the threshold.

Thus, the idea behind an appropriate just-in-time decoding scheme is this: We need to close off the flux worldlines that are created due to the noise as soon and as reliably as possible. However, if the decoder applies the correction as soon as the flux is detected, we risk adding large flux

worldlines that accidentally connect measurement errors. If the correction is applied too late, the flux worldlines can proliferate as in the example in Fig. 15(a). The resolution is to apply the correction as soon as we are confident enough that the detection event (which we also call the noise syndrome) is not a measurement error. As a rule of thumb, a pair of flux worldlines can be closed when the spatial separation between detection events is of the same order as the time these flux worldlines have been observed for.

The just-in-time decoding scheme discussed here is based on the idea first proposed by Bombin [22] and follows more closely the variant proposed by Brown [23] (also see Ref. [29]). Our discussion provides a slightly different explicit proof and makes a more direct relation to topological phases.

When it comes to error correction of charges (that correspond to Abelian anyons), we can simply store the measurement outcomes of associated stabilizers and process them at the end of the protocol. One difference with the usual decoding of Abelian codes is that detection events of charge measurements can be caused not only by noise but also by the flux worldlines themselves. Nevertheless, below we show that the resulting distribution must be correctable below the threshold error rate.

Finally, when it comes to the fault tolerance of the whole protocol, we need to take into account additional factors, such as (i) initialization and readout, (ii) dealing with boundaries and domain walls, and (iii) extracting logical measurement outcomes from the physical measurement outcomes (for example, in the logical $\overline{CZ}$ measurement protocol). In addition, the spacetime geometry has to be chosen such that the shortest nontrivial and undetectable operator has a size $O(d)$, where d is the spatial distance of the codes used in the computation.

In this work, we assume instantaneous and perfect global classical computation and communication. It would be interesting to explore a classically local solution for just-in-time error correction. This would enable classically local universal quantum computation with topological codes in three dimensions or less, which is currently an open problem [106].

B. Fault tolerance proof

1. Proof setting

In this subsection, we assume a circuit whose $+1$-postselected execution corresponds to a fixed-point path integral for the TQD phase on a torus, such as presented in Sec. V. For such a circuit, the -1 measurement outcomes correspond to flux and charge worldlines in the path integral. As discussed above, we consider the simplified case where the measured flux worldlines can propagate only in the time direction and discuss the more general case in Sec. VII D. We assume a noise-free initial state with periodic boundary conditions (i.e., on a spatial torus of a size $L \times L$). We first show the existence of the threshold in this setting and argue the existence of the threshold for complete logical protocols in the following subsection.

We call the end points of charge and flux worldlines in spacetime the noise *syndromes* (there are charge and flux syndromes, correspondingly). This must include the effect of the just-in-time corrections which themselves produce flux worldlines. We assume a general noise model (where we label a noise realization as $\tilde{E}$) that can consist of arbitrary local quantum channels acting on arbitrary qubits at given locations in spacetime as well as faulty gates and measurement outcomes. For the noise realization $\tilde{E}$, we call E the set of all spacetime points that experience error under a given noise realization. The probability distribution determining the error model is specified later on. Here, we use that, for each noise realization and each circuit trajectory, the measurements in the circuit yield a specific distribution of flux and charge worldlines.

To perform error correction in the flux sector, the circuit is run while simultaneously implementing corrections determined by the just-in-time decoder. The charge-detecting measurement outcomes are stored and processed at the end of the protocol. For each time step t, the decoder determines the set of correction strings C_t (which split into red, green, and blue classes for the corresponding sublattices) and passes them to the circuit. The circuit then applies string operators that insert flux worldlines along C_t, at a fixed time t, that close off some of the measured flux worldlines. For all protocols considered in this paper, these string operators are simply strings of Pauli-X operators. From the perspective of Sec. III, as well as circuits in Sec. V, such a Pauli-X string indeed removes the flux defects at its end points, because it anticommutes with the associated plaquette measurements. Furthermore, a Pauli-X string creates a random collection of Abelian charges along its support. These charges are detected by the protocol and can be corrected by the RG decoder at the end of the protocol so long as the clusters containing flux worldlines are sufficiently small (see Sec. IV B 2).

2. Error model and its properties

We assume a noise model wherein different error realizations $\tilde{E}$ occur with different probabilities $P(\tilde{E})$. Each noise realization corresponds to adding or replacing some of the quantum channels or measurements in the noise-free circuit with a faulty channel or measurement. We denote the set of spacetime locations of the faulty channels by E.

Furthermore, we assume that the probability of an error configuration is suppressed in its size or, more precisely, that the probability of a fixed set of channels being faulty decays exponentially with the size of this set.

Definition 1. We say that the noise model is p-bounded if, for any noise realization $\tilde{E}$, the probability of a set of spacetime points A entering E is bounded as

$$\mathbb{P}(A \in E) = \sum_{\tilde{E}: A \subset E} \mathbb{P}(\tilde{E}) \leq p^{|A|}. \tag{78}$$

In the literature, this is also called local stochastic noise with rate p. This model is a worst-case model over faulty locations and is compatible with any underlying local physical noise process including various circuit noise models, amplitude damping, dephasing, and biased noise; in terms of correlations, it covers both independent identically distributed (i.i.d.) and correlated noise, provided that the probability of a fault set decays exponentially in its support.

An important property of this noise model is that there exists a cluster decomposition of the spacetime distribution of noise. We follow the discussion in Ref. [107] (where the cluster decomposition is called a chunk decomposition); see also Refs. [108–110] for the original introduction of the method by Gács. The cluster decomposition is defined as follows. Suppose that we fix some constant $Q > 0$. We call any point in E a level-0 minimal cluster. A subset of E is a level-n minimal cluster if it is a union of two level-$(n-1)$ minimal clusters and its diameter is at most $Q^n/2$; it has precisely 2^n points. We call $E_{\geq n}$ the union of all minimal level-n clusters and define the set of all strictly level-n minimal clusters as $F_n = E_{\geq n} \backslash E_{\geq n+1}$. The minimal cluster decomposition of the error set exists if $E = F_0 \cup F_1 \cup \cdots \cup F_m$. We call an R-connected component of a set E a subset that cannot be further partitioned into two subsets separated in distance for at least R. We use the following statements that have been proven in Proposition 7 in Ref. [107]

Lemma 1 (existence and properties of the cluster decomposition, Ref. [107]). For a p-bounded noise model with $p \leq p_c = 1/(3Q)^6$ and $Q \geq 6$, the following statements hold.

(1) The set of spacetime errors can be decomposed into minimal clusters as

$$E = F_0 \cup F_1 \cup \cdots \cup F_{n_{\max}}, \tag{79}$$

$n_{\max} = \lfloor \log_Q L/2 \rfloor - 1$, with probability

$$\mathbb{P} \geq 1 - C\left(\frac{p}{p_c}\right)^{O(L^\eta)}, \tag{80}$$

where C is some positive constant and $\eta = 1/\log_2 Q$.

(2) We refer to the Q^n-connected components of F_n as *level-n clusters*. Any level-n cluster M of F_n has diameter at most Q^n and is separated from $E_n \backslash M$ (but not from $E \backslash M$) by at least $\frac{1}{3} Q^{n+1}$.

The proof in Ref. [107] is technically written for the i.i.d. noise; however, it directly generalizes to p-bounded noise—see also Refs. [108–110].

3. Just-in-time algorithm

We now introduce a specific implementation of just-in-time decoding, which we call the just-in-time RG decoder. This decoder is not designed to be highly performant in practice; rather it allows us to rigorously prove the existence of a threshold. We discuss how to perform more efficient practical decoding in Sec. VII E. For most of this subsection, we focus on the just-in-time decoding in the flux sector and later address the charge sector. To simplify our proof, we use the usual RG decoder [107] for the charge sector at the end of the protocol.

We assume that the circuit is associated with spacetime lattices corresponding to possible locations of flux worldlines, charge worldlines, and qubits. We use a rectangular coordinate system on the triple of lattices and the box metric ℓ_∞ to determine the distance between points in spacetime: Namely, for $x = (x_1, x_2, x_t)$ and $y = (y_1, y_2, y_t)$, call $d(x, y) = \max(|x_1 - y_1|, |x_2 - y_2|, |x_t - y_t|)$. The flux-detecting measurements performed by the circuit at each time t are stored in an array m_t. At each step, the decoder uses this together with the information about all corrections administered up to time t to determine the syndrome at time t that we call $\mathcal{M}_t$. It includes the measured flux worldline end points (excluding the apparent "termination" of flux worldlines at the current time t) minus the end points of the correction strings.

Below, we present the RG just-in-time decoding algorithm. At each time step t, the input to the decoder is the set of flux measurements in the circuit m_t. The output of the decoder at each time step t is the collection of strings C_t which the circuit turns into a (Pauli-X) correction operator P_t that is then immediately applied. In between two time steps, the decoder memorizes some information, namely, a set of spacetime points Σ_t, which is the set of syndromes that are unmatched at time t. We define ∂C_t to be the end points of the string C_t.

We use the notation $\oplus$ to denote the symmetric difference between sets. Define k_m to be the largest integer satisfying $2^{k_m} < L/2$.

In step 3.2, we consider different u and u' "greedily" in an arbitrary order. The outcome of the decoder depends on the order, but our proof of the threshold for this decoder is order-independent.

We call the index k in the algorithm above the *level* of the operation of the decoder. Any pair of points (u, u') that is removed from Σ_t due to step 3.2 is said to be "removed by pairing." The array of current syndromes Σ_t captures exclusively the noise syndromes up to time t that have not been paired yet.

Informally, the algorithm works as follows. At each point in time, the unpaired syndromes are stored in Σ_t (updated at

Algorithm 1. Just-in-time RG decoder.

At $t = 0$, initialize empty Σ_0. At each time step $t \geq 0$
1: Determine $\mathcal{M}_t$ from m_t and C_{t-1}.
2: Update $\Sigma_t \to \Sigma_t \oplus \mathcal{M}_t$;
3: Initialize C_t to be empty and the set $s = \Sigma_t$.
4: For each $0 \leq k \leq k_m$:
  3.1: remove all points $u = (u_1, u_2, u_t)$ in s, such that $t - u_t < 2^k$;
  3.2: for each $u \in s$, if $u' \in s$ exists such that $d(u, u') \leq 2^k$, find any shortest path for the (Pauli-X) correction string $C_t^{(k),i}$ matching their spatial coordinates. Update $C_t \to C_t \cup C_t^{(k),i}$ and remove u, u' from s and from Σ_t;
5: Pass the string C_t to the circuit for correction.

step 1 of the algorithm). Then, for each scale 2^k, the decoder inspects the history of all unpaired syndromes in spacetime up to the current point in time. The decoder matches a syndrome only if it knows a size-2^k region around it in spacetime. Namely, if a syndrome was detected in less than 2^k time steps in the past, it is set aside and not matched at this time step. Thanks to this property, the measurement errors (which can temporarily violate the parity of syndromes while the time slice is still going through the error cluster) are not incorrectly matched to errors that belong to a different cluster. Instead, the decoder waits to determine which cluster the syndrome must belong to. For all points that pass the test, the decoder proceeds with greedy pairing, as long as the points are close at a given scale. The operation of the decoder illustrating these points is shown schematically in Fig. 16. We call this scheme a "RG decoder" as it operates scale-by-scale similarly to the RG decoder of Bravyi and Haah [107] and matches syndromes greedily whenever it can be done reliably.

4. Fault tolerance of the just-in-time scheme

To prove the existence of a threshold, we assume that we start with a noiseless code state of the TQD model, then run the noisy circuit implementation of the TQD phase together with the just-in-time decoder for T time steps (in the interval of times $[0, T]$), and then run a noiseless version of the same circuit together with the same decoder for another $T_1 \sim O(d)$ time steps (in the interval of times $[T, T + T_1]$), where $d = L$ is the spatial distance of the code. We explain below how the existence of a threshold extends to full circuits implementing logical protocols.

Here, we state the threshold theorem for the just-in-time scheme in our current idealized setting, which says that, under a p-bounded noise model, the logical error rate of a circuit (together with the just-in-time decoder) implementing the TQD phase on a $L \times L$ torus decays exponentially in L^η (for some constant $\eta > 0$) for noise rate below a threshold.

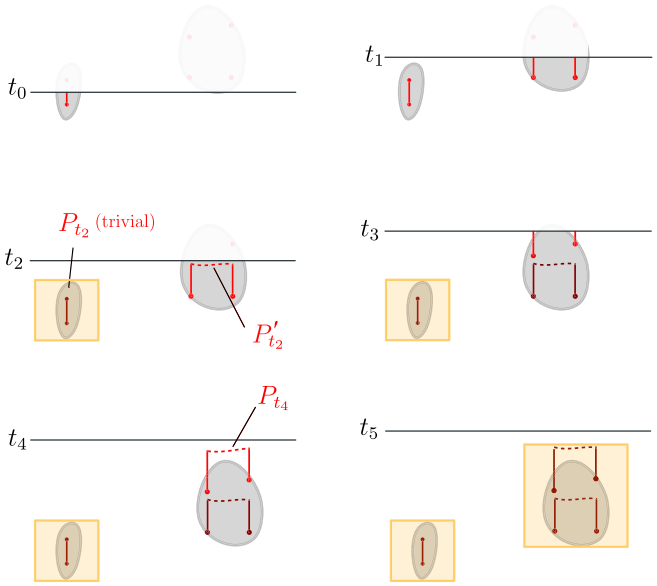


FIG. 16. Schematic example of operation of a circuit with the just-in-time RG decoder in the presence of two noise clusters. The cluster on the left produces a measurement error, corresponding to a short timelike open flux worldline. The decoder avoids determining the correction until time t_2, when both syndromes have been measured and their distance from the current time slice is large enough (larger than their separation but smaller than the separation from other clusters). The correction P_{t_2} is trivial (no operator is applied). The yellow box shows the smallest box containing both the noise and the correction. The cluster on the right produces four syndromes which could be due to a pair of stringlike errors or a pair of measurement errors. One can directly verify that, in either case, the corrections P'_{t_2} and P_{t_4} would close the flux configuration.

Theorem 1. Consider an error-correcting circuit that implements the TQD model on an $L \times L$ torus together with the just-in-time RG decoder under a p-bounded noise model as discussed in this section. Let $\mathcal{C}_{\tilde{E}}$ be this channel implementing the circuit under a given noise realization $\tilde{E}$, let $\mathcal{E}$ be the encoding channel into an error-free logical state for the circuit (where the encoded states belong to the 22-dimensional code state of the TQD model; see Sec. III C), and let $\mathcal{D}$ be the decoding channel consisting of (i) running the just-in-time flux error correction error-free for the time $O(L)$ and (ii) performing the charge correction error-free using regular RG decoder.

Then, there exists a threshold with a lower bound $p_c = [1/(3Q)^6]$ with $Q = 253$, such that for $p < p_c$ we have

$$\sum_{\tilde{E}} \mathbb{P}(\tilde{E}) \mathcal{D} \circ \mathcal{C}_{\tilde{E}} \circ \mathcal{E} = (1 - p_L)\mathcal{E} + p_L \tilde{\mathcal{E}} \qquad (81)$$

for some (trace-preserving) channel $\tilde{\mathcal{E}}$, and the logical error probability is bounded as

$$p_L \leq C\left(\frac{p}{p_c}\right)^{O(L^n)} \tag{82}$$

for some constant $C > 0$.

We prove this theorem at the end of this subsection. The rest of the subsection develops the tools necessary for proving it.

We now look at some fixed error realization $\tilde{E}$. When $\mathcal{D} \circ \mathcal{C}_{\tilde{E}} \circ \mathcal{E} = \mathcal{E}$, the outcome of the circuit with decoder is the same as the initial (clean) encoded state; thus, we say that no logical error has occurred.

We assume that for an error ε contained in a box $\mathcal{B}$ with diameter b, its syndromes obtained after measurements in circuits are generally contained in a γ neighborhood of $\mathcal{B}$ (including $\mathcal{B}$), where $\gamma \geq 0$ is a system-size-independent constant which depends on the details of the circuit. We can always rescale the lattice such that $\gamma = 1$, which we assume below.

The following lemma provides sufficient criteria that ensure that no logical error occurs.

Lemma 2. Consider the channel $\mathcal{C}_{\tilde{E}}^{\vec{f}}$ corresponding to a trajectory of the circuit under a given noise realization $\tilde{E}$ with measurement outcomes for the flux and charge worldlines W_f and W_c, respectively, as well as the union of all flux correction strings C_f and for the charge correction strings C_c, which we collectively denote as $\vec{f} = (W_f, W_c, C_f, C_c)$. The following statements hold with probability 1.

(i) The union of flux worldlines and their corrections terminate in the vicinity of errors: $\partial(W_f + C_f) \subseteq \mathcal{N}_1(E)$, where ∂ denotes the boundary map.

(ii) The union of charge worldlines and their corrections terminate in the vicinity of errors and fluxes: $\partial(W_c + C_c) \subseteq \mathcal{N}_1(E \cup W_f \cup C_f)$.

(iii) Connected components of errors are flux-neutral. Namely, consider any connected component X of $\mathcal{N}_1(E)$. Then, $\partial(W_f + C_f) \cap X$ contains an even number of points.

(iv) A box fully containing one or several connected components of regions that can create charges (i.e., containing noise, flux worldlines, and flux corrections) is charge-neutral. Namely, consider any part Y of $\mathcal{N}_1(E \cap W_f \cap C_f)$ that is contained on the inside of some two-sphere and disconnected from the rest. Then, $\partial(W_c + C_c) \cap Y$ contains an even number of points.

In addition, the following statement is true.

(v) There is no logical fault, namely,

$$\mathcal{D} \circ \mathcal{C}_{\tilde{E}}^{\vec{f}} \circ \mathcal{E} = \mathcal{E}, \tag{83}$$

if the following two conditions hold:

(a) the set $\mathcal{N}_1(E \cup W_f \cup C_f)$ is topologically trivial—i.e. it is contained on the inside of some two-sphere in the 3D spacetime;

(b) for any closure $\tilde{C}$ of $W_c + C_c$ inside of $\mathcal{N}_1(E \cup W_f \cup C_f)$, $\tilde{C} + W_c + C_c$ is homologically trivial.

Note that, in (b), it is always possible to find $\tilde{C}$ due to (iii), and the homology of $\tilde{C} + W_c + C_c$ is independent on the choice of $\tilde{C}$ if (a) already holds.

Intuitively, statement (v) says that a logical fault can occur only if the error together with the correction is topologically (or homologically) nontrivial. The lemma also takes into account the fact that, due to the twist in the path integral, charge worldlines can end on flux worldlines; see Sec. IV B 2. These properties follow from the fixed-point properties (namely, zero correlation length) of the path integral that are inherited by the circuit and of its realization with the flux and charge worldlines, discussed in Sec. IV B 2. We find that they hold for any of the circuits proposed in this paper, but we do not explicitly show this. For explicit proofs of analogous statements, we refer the reader to Refs. [78,89].

We start by showing an auxiliary result; the decoder that runs up to a high enough level fully corrects an error cluster in complete isolation from any other errors while increasing its diameter by at most a certain fraction. A subset $A \in E$ of the error set E is called *isolated* from other errors by distance R if $d(A, E\backslash A) \geq R$. We call the box $\mathcal{B}_{\text{is}}$ an isolation region of set A if it includes the R neighborhood of the smallest box $\mathcal{B}$ containing A and does not include any points from E.

Lemma 3. Assume some fixed $0 < n < n_{\max}$ and $Q > 24$ and that we are given a noise realization that contains an error cluster F that fits in a box $\mathcal{B}$ sized Q^n and is isolated from any other error by at least $\frac{1}{3}Q^{n+1}$. Denote its syndromes by σ_F and assume that $\mathcal{B}$ starts at time t_0. We denote the isolation region of this error in spacetime as $\mathcal{B}_{\text{is}}$.

We continuously run the operation of the circuit just-in-time decoder (Algorithm 1) for all k up to the smallest k such that $2^k \geq Q^n + 2$. We call this value k_n.

(a) At each time step $t \geq t_0$, by the end of operation of level-k_n decoder, the set $s \cap \mathcal{B}_{\text{is}}$ will be empty, and

(b) the syndromes σ_F will be removed from Σ_t by the time $t = t_0 + 3Q^n + 5$. The union of all corrections determined by level-k operation of the just-in-time decoder $C^{(k)}$ with $k \leq k_n$ together with F is contained in a $(2Q^n + 5)$ neighborhood of $\mathcal{B}$.

Proof. First, we notice that the syndromes σ_F must be contained in a 1-neighborhood of $\mathcal{B}$, which we call $\mathcal{B}_{\text{syn}}$. In addition, their number must be even [since any isolated error must be neutral, see Lemma 2(iii)].

As we discussed below Algorithm 1, Σ_t at each time step contains only unpaired noise syndromes. Consider first $t \in [t_0, t_0 + Q^n]$. At every such time step, a given point u in Σ_t must be either left with no action due to step 3.1 (i.e., removed from s and passed directly to Σ_{t+1}) or paired with

another point in σ_F and removed (i.e., removed from s but not passed to Σ_{t+1}). Any point $u \in \sigma_F$ cannot be paired with any point v outside of σ_F because $d(u,v) \geq \frac{1}{3}Q^{n+1} - 2 > 2^{k_n}$ (the latter is true for $Q > 24$). This shows (a) for $t \in [t_0, t_0 + Q^n]$.

The total correction C_t^k for each such time step $t \in [t_0, t_0 + Q^n]$ must be contained in the 1-neighborhood of the box $\mathcal{B}$. This is because, for any $u, u' \in \sigma_F$ that are paired, the shortest path between their spatial coordinates must lie in the spatial shadow of $\mathcal{B}$ by definition of $\mathcal{B}$; this path placed in spacetime at time $t+1$ for $t \in [t_0, t_0 + Q^n]$ must be in 1-neighborhood of $\mathcal{B}$.

Consider now $t > t_0 + Q^n$, i.e., the times later than the latest time coordinate in σ_F. There must be an even number from points in σ_F left in Σ_t at the beginning of each time step. At the time $t = t_0 + Q^n + 2^{k_n}$, any point $u \in \sigma_F \cap \Sigma_t$ is closer to any remaining point $v \neq u$, $v \in \sigma_F \cap \Sigma_t$, since $\max_{u,v\in\sigma_F} d(u,v) \leq Q^n + 2$ and $2^{k_n} \geq Q^n + 2$. In addition, it cannot be removed from s by any level $k < k_n$ of the operation of the decoder. Therefore, the algorithm must remove all the points by pairing at this time at the latest. Because of the shortest matching, the totality of corrections $C^{(k)}$ matching the points in σ_F must belong to a $(2^k + 1)$ neighborhood of $\mathcal{B}$. Because of $2Q^n + 4 \geq 2^k$, we obtain the statement of (b).

In addition, at each time step in the paragraph above, any point in $s \cap \mathcal{B}_{\text{is}}$ must be either removed by step 3.1 or paired up by the time $t = t_0 + Q^n + 2^{k_n}$. This completes statement (a) as well. ∎

As a direct consequence of the lemma above, for any noise realization obeying cluster decomposition, in a circuit that is followed by a noiseless just-in-time decoder, all the flux clusters will be corrected by the time $2Q^{n_{\max}} + 5$. Since $Q^{n_{\max}+1} < L/2$, for large enough Q, $2Q^{n_{\max}} + 5 < L/2$. Thus, if we run noiseless decoding for another $T_1 = O(L)$ time, the state is guaranteed to return to some clean logical state in the flux sector.

Now, we show that the just-in-time decoder is able to correct an error realization in spacetime that obeys cluster decomposition with large enough Q.

Theorem 2 (the effect of the just-in-time RG decoder on noise clusters). Given a noise trajectory E that obeys cluster decomposition from Lemma 1 with $Q > 24$, assume running the circuit with the just-in-time decoder under assumptions of this section. Then, the points in each cluster are paired only with the points within the same cluster (thus, there is a well-defined correction associated with each cluster), and the following holds.

(a) By time $T + T_1$, the syndromes for each cluster $F_{n,i}$ for every $0 \leq n \leq n_{\max}$ are removed by pairing.
(b) The correction associated with each cluster $F_{n,i}$ is contained inside a $(2Q^n + 5)$ neighborhood of $F_{n,i}$.

Proof. We first show that, at each time step t, the syndromes of each error cluster $F_{n,i}$, which we denote $\sigma_{F_{n,i}}$, are operated upon by the just-in-time decoder at levels $k \leq k_n$ (i.e., the smallest k such that $2^k \geq Q^n + 2$) as if this cluster was completely isolated from any other errors by $\frac{1}{3}Q^{n+1}$. Once we show this, we can use statement (b) in Lemma 3 from which the statement of the theorem follows straightforwardly.

We show this by induction. Each level-0 cluster $F_{0,i}$ is just a single error that is isolated from any other error by at least $\frac{1}{3}Q$. In this case, Lemma 3 applies. From Lemma 3(a), we additionally have that, at every time step, by the smallest level of decoder operation k_0: $2^{k_0} \geq 3$, the syndromes in $\sigma_{F_{0,i}}$ that are contained in s must be removed from it. From the proof of the lemma, we also have that any in $\sigma_{F_{0,i}}$ can be paired only with another syndrome in this set. This completes the induction base.

Next, assume that the assumption holds for some n. This means that all syndromes of errors in $\cup_{i,m<n} F_{m,i}$ have been treated according to the assumption and, by Lemma 3(a), their syndromes must be removed independently of the rest of the error clusters. This means that the decoder can be effectively assumed to operate on a decomposition $\cup_{n<s<n_{\max}} F_s$. Consider connected components $F_{n+1,i}$, which we can now assume to be isolated from any other error whose syndromes enter in s by at least $\frac{1}{3}Q^{n+2}$. Applying Lemma 3, we obtain the statement of the theorem. ∎

We denote the R neighborhood of a box A as $\mathcal{N}_R(A)$, which we define to be a set that includes the points in the R neighborhood of A as well as A itself. For any given constant μ, we also define $\mathcal{N}_{\mu,n} \equiv \mathcal{N}_{\mu Q^n}$.

We now show that, if the decoder "spreads" each cluster $F_{n,i}$ of the error by at most some factor αQ^n (from Theorem 2, for the just-in-time decoder in this section, we can choose $\alpha = 7$, which is a conservative overestimate), we can consider the set of all "fattened" boxes and find a decomposition that is effectively sparse if Q is large enough.

Calling $\bar{\mathcal{B}}_{n,i}$ each αQ^n-"fattened" cluster $F_{n,i}$, we now have a set of points in spacetime $\bar{\mathcal{B}} = \cup_{n=0;i}^{n=n_{\max}} \bar{\mathcal{B}}_{n,i}$ to which all flux worldlines in the spacetime volume of the circuit are constrained. We also refer to $\bar{\mathcal{B}}_{n,i}$ as a "box" where it does not cause ambiguity. Some of the fattened clusters can now overlap; see Fig. 17. Nevertheless, we now show that $\bar{\mathcal{B}}$ admits a sparse decomposition. To this end, we first introduce the following definitions.

Definition 2 (linking). Consider a union of sets $\bar{\mathcal{B}} = \bigcup_{n=0;i}^{n=n_{\max}} \bar{\mathcal{B}}_{n,i}$. We say that the box $\bar{\mathcal{B}}_{s,i}$ is directly linked to the box $\bar{\mathcal{B}}_{m,j}$ if

$$d(\bar{\mathcal{B}}_{s,i}, \bar{\mathcal{B}}_{m,j}) \leq 70Q^{\min(s,m)} \tag{84}$$

(if two boxes overlap, we define the distance between them to always be 0). We refer to the condition above as the existence of an s link.

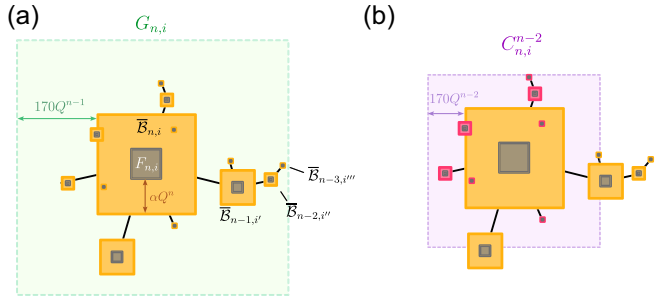


FIG. 17. Schematic illustrating linking of "fattened" clusters (shown in orange) into (a) a linked group $G_{n,i}$ and (b) $(n-2)$-candidate group $C^{n-2}_{n,i}$ (not to scale). The boxes that are linked are connected by a black line. The group $G_{n,i}$ contains all the boxes linked to $\bar{\mathcal{B}}_{n,i}$. The boxes of level $s \le n-2$ that are linked to $\bar{\mathcal{B}}_{n,i}$ via chains that do not contain $m > n-2$ boxes are shown in red in (b); together, these form an $(n-2)$-candidate group.

We say that the box $\bar{\mathcal{B}}_{s,i}$ is (indirectly) linked to the box $\bar{\mathcal{B}}_{m,j}$ if there exists a chain of links that starts at the box $\bar{\mathcal{B}}_{s,i}$ and ends at the box $\bar{\mathcal{B}}_{m,j}$. If such a chain of links does not exist, we say that $\bar{\mathcal{B}}_{n,i}$ and $\bar{\mathcal{B}}_{m,j}$ are unlinked.

The parameter 70 was chosen such that the following hold.

Definition 3 (linked groups). We define a linked group $G_{n,i}$ as a union of boxes that consists of at least one box and is obtained as recursively adding every box that is linked to any box that is already in the group, where n is the highest level of a box in the group (we show below that there is a unique highest-level box per group). In other words, $G_{n,i}$ is a connected component of the set $\bar{\mathcal{B}}$ under linking.

For any given $\bar{\mathcal{B}}_{n,i}$, define the according s-candidate group $C^s_{n,i}$ with $s \le n$ as the union of all $\bar{\mathcal{B}}_{m,j}$ with $m \le s$ that are directly or indirectly linked to $\bar{\mathcal{B}}_{n,i}$ via chains of links that do not contain any boxes of level larger than s.

Examples of a cluster group and a candidate group are shown in Fig. 17. By definition, the union of all fattened boxes can be decomposed into cluster groups: $\bar{\mathcal{B}} = \bigcup_{n=0;i}^{n=n_{\max}} G_{n,i}$. In the following lemma, we establish some properties of cluster groups that are instrumental to then showing that their union, in fact, provides a sparse cluster decomposition of $\bar{\mathcal{B}}$.

Lemma 4 (linking properties). Consider a set E obeying the cluster decomposition $E = \bigcup_{n \le n_{\max}} F_n$ from Lemma 1 for $Q > 252$. For each cluster $F_{n,i}$, define the associated "fattened" cluster $\bar{\mathcal{B}}_{n,i}$ as the smallest box fitting αQ^n neighborhood of $F_{n,i}$. Now consider the set of points $\bar{\mathcal{B}} = \bigcup_{n=0;i}^{n=n_{\max}} \bar{\mathcal{B}}_{n,i}$. Under Definitions 2 and 3 introduced above, the following properties hold:

(i) Consider three boxes $\bar{\mathcal{B}}_{n,i}$, $\bar{\mathcal{B}}_{m,j}$, and $\bar{\mathcal{B}}_{s,\ell}$ with $m \ge n$ and $s < m, n$. If the pairs $\bar{\mathcal{B}}_{n,i}$, $\bar{\mathcal{B}}_{s,\ell}$ and $\bar{\mathcal{B}}_{m,j}$, $\bar{\mathcal{B}}_{s,\ell}$ are directly linked, then $\bar{\mathcal{B}}_{n,i}$ and $\bar{\mathcal{B}}_{m,j}$ must be directly linked.

(ii) Any two same-level boxes $\bar{\mathcal{B}}_{n,i}$ and $\bar{\mathcal{B}}_{n,j}$ with $i \ne j$ cannot be linked directly. They also cannot be linked indirectly unless there exists a higher-level box $\bar{\mathcal{B}}_{m,\ell}$ to which they both are linked.

In addition, the groups of boxes have the following properties:

(iii) Each linked group $G_{n,i}$ contains a unique highest-level box $\bar{\mathcal{B}}_{n,i}$.

(iv) For any given box $\bar{\mathcal{B}}_{n,i}$ and $s \le n$, the candidate group $C^s_{n,i}$ fits in the $170Q^s$ vicinity of the box:

$$C^s_{n,i} \subseteq \mathcal{N}_{170,s}(\bar{\mathcal{B}}_{n,i}). \tag{85}$$

Proof. (i) Assume that $\bar{\mathcal{B}}_{s,\ell}$ is directly linked to both $\bar{\mathcal{B}}_{n,i}$ and $\bar{\mathcal{B}}_{m,j}$. This means

$$d(\bar{\mathcal{B}}_{s,\ell}, \bar{\mathcal{B}}_{n,i}) \le 70Q^s, \qquad d(\bar{\mathcal{B}}_{s,\ell}, \bar{\mathcal{B}}_{m,j}) \le 70Q^s. \tag{86}$$

By the triangle inequality, this means that $\bar{\mathcal{B}}_{n,i}$ and $\bar{\mathcal{B}}_{m,j}$ must be not too far from each other:

$$d(\bar{\mathcal{B}}_{n,i}, \bar{\mathcal{B}}_{m,j}) \le 2 \times 70Q^s < 70Q^n. \tag{87}$$

The second inequality holds for $Q > 2$ and shows that $\bar{\mathcal{B}}_{n,i}$ and $\bar{\mathcal{B}}_{m,j}$ are linked, proving statement (i).

(ii) First of all, we show that a direct link between $\bar{\mathcal{B}}_{n,i}$ and $\bar{\mathcal{B}}_{n,j}$ is impossible. We establish the following lower bound on the distance between the boxes:

$$\begin{aligned} d(\bar{\mathcal{B}}_{n,i}, \bar{\mathcal{B}}_{n,j}) &\ge d(F_{n,i}, F_{n,j}) - 2\alpha Q^n \\ &\ge \frac{1}{3} Q^{n+1} - 2\alpha Q^n > 70Q^n. \end{aligned} \tag{88}$$

The last inequality is true for $\alpha = 7$, $Q > 252$. Now consider the case when $\bar{\mathcal{B}}_{n,i}$ and $\bar{\mathcal{B}}_{n,j}$ are connected via a chain of direct links, which is irreducible in the sense that only the boxes that neighbor in the chain are linked directly, but no other pair in the chain is. On the contrary to the statement, let us assume that the chain contains only intermediary boxes of the level $s \le n$. There must exist a box $\bar{\mathcal{B}}_{s,\ell}$ with $s < n$ that directly links to two other boxes whose level is larger than s, which contradicts (i) by the irreducibility assumption. This shows (ii).

Property (iii) follows straightforwardly from (ii). By definition, any pair of boxes in $G_{n,i}$ are (indirectly) linked. Any highest-level box $\bar{\mathcal{B}}_{n,j}$ in $G_{n,i}$ cannot be linked by (ii) to any other box of the same level. Thus, it must be the only box of this level contained in $G_{n,i}$.

Finally, we show the last property (iv). Consider any $\bar{\mathcal{B}}_{n,i}$ and the associated s-level candidate group $C^s_{n,i}$. By properties (i) and (ii), the linking in the candidate group from $\bar{\mathcal{B}}_{n,i}$ to any lower-level box within the group must go monotonically down in levels. Therefore, in the worst-case

scenario, a single chain of links from the box $\bar{\mathcal{B}}_{n,i}$ goes all the way from s to 0, while each next box in $\{\bar{\mathcal{B}}_{m,i_m}\}_{m=s\ldots0}$ is directly linked to the previous one. The length these occupy together with the spacing due to the links can be bounded as

$$\mathcal{L} \leq 70Q^s + \mathrm{diam}(\bar{\mathcal{B}}_{s,i_s}) + 70Q^{s-1} + \mathrm{diam}(\bar{\mathcal{B}}_{s-1,i_{s-1}}) + \cdots + 70 + \mathrm{diam}(\bar{\mathcal{B}}_{0,i_0}). \tag{89}$$

Using that $\mathrm{diam}(\bar{\mathcal{B}}_{s,i_s}) = (2\alpha+1)Q^s = 15Q^s$, we arrive at

$$\mathcal{L} \leq (70+15)\frac{Q^{s-1}-1}{Q-1} < 170Q^s, \tag{90}$$

which is true for the chosen range of Q. Thus, since all link chains together with respective boxes must be within $\mathcal{L} < 170Q^s$ distance away from $\bar{\mathcal{B}}_{n,i}$, the entire candidate group must be in its appropriate vicinity $C^s_{n,i} \subseteq \mathcal{N}_{170,s}(\bar{\mathcal{B}}_{n,i})$, which gives statement (iv). ∎

Finally, we are ready to show that the set $\bar{\mathcal{B}}$ admits a sparse decomposition (into sets of points that are linked groups of clusters).

Lemma 5 (sparse decomposition of fattened clusters). Under the assumptions of Lemma 4, for $Q > 252$, the decomposition of $\bar{\mathcal{B}}$ into linked groups,

$$\bar{\mathcal{B}} = \bigcup_{n=0;i}^{n=n_{\max}} G_{n,i}, \tag{91}$$

is sparse. That is, there exists a constant $\beta = 20$ such that

(i) each group is bounded in size:

$$\mathrm{diam}(G_{n,i}) < \beta Q^n, \tag{92}$$

and

(ii) the separation between any given group and all the groups that are same level or higher is at least 3 times the size of this group; namely, for all $G_{m,j}$ with $m \neq n$, $m \geq n$, we have

$$d(G_{n,i}, G_{m,j}) > 3\beta Q^n. \tag{93}$$

Proof. Consider any cluster group $G_{n,i}$ whose highest-level box is $\bar{\mathcal{B}}_{n,i}$. By Lemma 4, it follows that $G_{n,i} \equiv C^{n-1}_{n,i}$. From Lemma 4(iv), we have $G_{n,i} \subseteq \mathcal{N}_{170,n-1}(\bar{\mathcal{B}}_{n,i})$, and, thus,

$$\begin{aligned}\mathrm{diam}(G_{n,i}) &\leq \mathrm{diam}[\mathcal{N}_{170,n-1}(\bar{\mathcal{B}}_{n,i})] \\ &\leq (2\alpha+1)Q^n + 2\times 170Q^{n-1} \\ &= 15Q^n + 340Q^{n-1} < \beta Q^n,\end{aligned} \tag{94}$$

which is true for $\beta = 20$ and $Q > 252$. This shows (a).

We now show (b). Specifically, we show that any box

$$\bar{\mathcal{B}}_{m,j} \in G_{\geq n} \backslash G_{n,i} \bigcup_{\substack{s\geq n;\ell \\ (s,\ell)\neq(n,i)}} G_{s,\ell}$$

is further from $G_{n,i}$ than $3\beta Q^n$. Any such box $\bar{\mathcal{B}}_{m,j}$ is either of the same or greater level $m \geq n$ or smaller level $m < n$ than the considered group $G_{n,i}$ and is not linked to any of the boxes in $G_{n,i}$ by assumption. We consider these cases separately and show that the distance between $G_{n,i}$ and any of such boxes is greater than $3\beta Q^n$, which is equivalent to (b).

In the case $m \geq n$, consider $d(G_{n,i}, \bar{\mathcal{B}}_{m\geq n,j})$. By Lemma 4 (iv) and since $\bar{\mathcal{B}}_{n,i}$ and $\bar{\mathcal{B}}_{m\geq n,j}$ must be unlinked by assumption, we have

$$\begin{aligned}d(G_{n,i}, \bar{\mathcal{B}}_{m\geq n,j}) &\geq d(\bar{\mathcal{B}}_{n,i}, \bar{\mathcal{B}}_{m\geq n,j}) - 170Q^{n-1} \\ &\geq 70Q^n - 170Q^{n-1} > 3\beta Q^n,\end{aligned} \tag{95}$$

which is true for $\beta = 20$ and $Q > 252$. This shows the statement (b) for $m \geq n$.

If $m < n$, there must exist some $\bar{\mathcal{B}}_{s,\ell} \in G_{\geq n}\backslash G_{n,i}$ with $s \geq n$ that must be linked to it; otherwise, $\bar{\mathcal{B}}_{m,j}$ would be in a lower-level group and would not be contained in $G_{\geq n}\backslash G_{n,i}$. We choose $\bar{\mathcal{B}}_{s,\ell}$ to be any lowest-level box satisfying $s \geq n$ that is linked to $\bar{\mathcal{B}}_{m,j}$. By assumption that we chose the box of the smallest such level, $\bar{\mathcal{B}}_{m,j}$ must be within its associated candidate group $C^{n-1}_{s,\ell}$. By Lemma 4 (iv), $\bar{\mathcal{B}}_{m,j} \in C^{n-1}_{s,\ell} \subseteq \mathcal{N}_{170,n-1}(\bar{\mathcal{B}}_{s,\ell})$, and, thus,

$$\begin{aligned}d(G_{n,i}, \bar{\mathcal{B}}_{m<n,j}) &\geq d[G_{n,i}, \mathcal{N}_{170,n-1}(\bar{\mathcal{B}}_{s\geq n,\ell})] \\ &\geq d[\bar{\mathcal{B}}_{n,i}, \mathcal{N}_{170,n-1}(\bar{\mathcal{B}}_{s\geq n,\ell})] - 170Q^{n-1} \\ &\geq d(\bar{\mathcal{B}}_{n,i}, \bar{\mathcal{B}}_{s\geq n,\ell}) - 2\times 170Q^{n-1} \\ &\geq 70Q^n - 2\times 170Q^{n-1} > 3\beta Q^n.\end{aligned} \tag{96}$$

The inequality before last holds because $\bar{\mathcal{B}}_{n,i}$ and $\bar{\mathcal{B}}_{s\geq n,\ell}$ must be unlinked; otherwise, they would belong to the same group. The last inequality holds because $\beta = 20$ and $Q > 252$. This shows (b) for $m < n$ and completes the proof of the lemma. ∎

The lemma above gives the following corollary, which shows that some of the criteria for the absence of logical errors must be satisfied when $Q > 252$.

Corollary 1. Condition (a) in Lemma 2 for the absence of logical failure is fulfilled for any error configuration E obeying a cluster decomposition.

Proof. By Theorem 2 and Lemma 5, $\mathcal{N}_1(E \cup W_f \cup C_f)$ obeys a cluster decomposition as well. Any sparse cluster decomposition where the size of the largest cluster is smaller than half of the system size is confined to isolated boxes and, thus, topologically trivial. ∎

5. Fault tolerance in the Abelian charge sector

For the Abelian charge sector, we simply record the syndromes until the end of the protocol and decode them using the usual RG decoder at the end of the circuit. We now show that the RG decoder succeeds if the noise obeys a cluster decomposition and the just-in-time decoder fattens the noise clusters such that the outcome is sparse.

Lemma 6. Under assumptions of this section, given any noise realization $\tilde{E}$ that obeys cluster decomposition with $Q > 252$, the RG decoder in Ref. [107] that is run at the end of a circuit with just-in-time decoding succeeds. Namely, condition (v)(b) in Lemma 2 holds [along with condition (v)(a), which was shown to hold in the previous subsection].

Proof. We first note that the charge syndromes are restricted to the region $\bar{\mathcal{B}}$ due to Theorem 2 and Lemma 2(ii). For any group $G_{n,i}$ in isolation, the RG decoder admits a correction that is in its 1-neighborhood. When $Q > 252$, from the sparse decomposition of $\bar{\mathcal{B}}$ shown in Lemma 5, each group $G_{n,i}$ must have syndromes fitting in a box of size less than $\beta Q^n + 2$ (recall that $\beta = 20$) and is separated by distance that is larger than $3 \times \beta Q^n - 2 > 2 \times \mathrm{diam}(G_{n,i})$ from any group of the same and larger size. Going inductively from smaller to larger levels as in the proof in Ref. [107], we find that the decoder must issue an appropriate correction [namely, the one that satisfies $\partial c(n,i) \oplus W_c(n,i) \oplus \sigma^c_{G_{n,i}} = 0$] that is within a 1-neighborhood of each cluster. Thus, the global RG decoder connects the syndrome associated with every group $G_{n,i}$ inside a 1-neighborhood of $G_{n,i}$. Since the boxes together with corrections from the RG decoder are isolated, the homology class of the corrections fits the error configuration, and thus condition (v)(b) of Lemma 2 holds. ∎

Finally, this brings us to the proof of the threshold theorem.

Proof of Theorem 1. Because of Lemmas 1 and 6, the corrections for both flux and charge sectors fit in a set of disjoint balls. Conditions (a) and (b) of Lemma 2 hold for any noise realization satisfying cluster decomposition with $Q = 253 > 252$. Thus, Lemma 2 states that there must be no logical failure in the circuit in this case.

The only circumstance when the logical failure can happen is when the p-bounded noise does not admit sparse decomposition with the chosen Q. By Ref. [107], for $p < p_c$ with $p_c = 1/(3Q)^6$, the probability of this is exponentially suppressed:

$$\mathbb{P}(\mathrm{fail}) = p_L \leq C \exp[-O(L^\eta)] \tag{97}$$

with some constants $C, \eta > 0$. Summed over all noise realizations, this yields Eq. (81) and proves the theorem. ∎

C. Fault tolerance of the full logical protocol

So far, we have proven only that we can fault-tolerantly store quantum information in the ground-state space of the TQD for an exponentially long time in the absence of domain walls and boundaries as well as spacetime regions with Abelian phases that are generally present in all protocols. We now discuss how our arguments can be extended to show fault tolerance of the full logical protocols proposed in this paper, which must include dealing with spatial boundaries, domain walls, and the regions with Abelian phases at the beginning and the end of the protocol. In the rest of this section, we present arguments that support our statements and leave rigorous proofs to future work.

1. Abelian phase in the protocols

We first consider the simpler case of protocols where the domain wall between the TQD phase and the Abelian toric codes phase is time-perpendicular (i.e., a switching occurs in one round in the whole space; note that, for every protocol in Sec. IV, we presented a variation which includes only such domain walls), such as in $\overline{CZ}$ logical measurement protocols. We consider the flux sector only here and discuss the charge sector at the end of this subsection.

In these protocols, the TQD state is prepared from the toric code state, which is also realized by a noisy circuit. We first present an informal argument showing that the earlier considerations for the absence of logical failure extend to this case, if the just-in-time decoding in the TQD phase is preceded and followed by global RG decoding for the Abelian phase. Right before switching to the TQD phase, we perform the usual global RG decoding for the Abelian phase and apply the correction [for the RG decoder to be successful, it might be necessary for the Abelian phase to have existed for $O(d)$ time, where d is the spatial distance of the code]. Next, the TQD phase is realized by a circuit together with the just-in-time decoder. The residual noise (which will be there in the presence of measurement errors) can be seen by the just-in-time decoder as having started at the beginning of the TQD phase. After that, the protocol again switches to the Abelian phase, where we can simply continue recording measurement outcomes without applying corrections. At the end, we use the information about the unpaired syndromes left by the just-in-time decoder at the end of the TQD phase together with the syndromes of the new noise in the Abelian phase following the TQD one to perform another round of the usual global RG decoding. With these modifications, the flux and charge worldlines associated with noise and corrections throughout the entire circuit continue being sparsely clustered.

2. Domain walls and boundaries

In other examples, such as unitary $\overline{CCZ}$ and $\bar{T}$ protocols, the regions with the Abelian phase are connected to the TQD region via domain walls that are tilted in spacetime. In this case, starting from the point where such domain walls appear, we use a (modified) just-in-time decoding in both the Abelian and non-Abelian regions. Namely, the

just-in-time protocol now must take the presence of the domain wall into account by appropriately matching the fluxes of specific colors to it. For different flux colors, the decoding must work differently. For an $\langle rp, g\rangle$ domain wall, the decoding for the r fluxes in the TQD and the p fluxes in one of the toric code copies must be performed together (i.e., viewed as the same type of flux by the decoder) "ignoring" the domain wall. Thus, for these sectors, the just-in-time decoding works as if the domain wall was absent. On the other hand, the green g fluxes can terminate on the $\langle rp, g\rangle$ domain wall. Therefore, for the green flux sector, the domain wall can be treated as a boundary that is tilted in spacetime. We now explain how the just-in-time decoding must be modified in one flux sector to appropriately account for a boundary where one type of fluxes can terminate.

Consider a TQD circuit with a flux-terminating boundary (domain wall). We call the set of points in spacetime corresponding to the locations on the boundary $\mathcal{Y}$. The spatial coordinates of the boundary at time t are denoted $\mathcal{Y}_t$. For a correction string C_t terminating at the boundary, $\partial C_t|_{\mathcal{Y}}$ is always set to be empty. We show a modified version of the just-in-time RG decoder which now matches points to the boundary (or a domain wall) when it is appropriate.

The modification consists of adding steps 3.2 and 3.3. According to 3.2, the noise clusters that are close to the boundary can be ignored if the phase in their spatial support becomes trivial before the correction needs to be applied. Otherwise, according to step 3.3, the points are preferentially matched to the boundary as long as the cluster is close enough to the boundary in spacetime. However, they are specifically matched to the *current* location of the boundary since the latter may be moving.

The argument for the existence of the threshold in the presence of a boundary is analogous to the case without boundaries as long as the boundary moves at a slow enough speed. Assume that its speed is χ per time step. Consider any cluster $F_{n,i}$ at level n. If it is located in the proximity of the boundary in spacetime, in the worst-case scenario the boundary moves away from the cluster by $\chi(Q^n + 2Q^n + 5)$ in the (upper-bounded) time it takes the decoder to make a decision to match. Therefore, now the padding region can increase by this amount, i.e., from $2Q^n + 5$ to $f(Q) \equiv (1+\chi)(2Q^n+5)+\chi Q^n$. This effectively changes α in the fault tolerance proof and, thus, the rest of the numerical parameters (which depend on α, even though we kept this dependence implicit in our proof) but nothing else.

Finally, other scenarios involving boundaries include situations when a certain color of fluxes cannot terminate at the boundary (in which case the just-in-time protocol is unchanged and simply does not match to the boundary) or two colors of fluxes can terminate only simultaneously (an example being $\langle rg\rangle$ boundary). In this case, syndromes corresponding to fluxes of two colors have to be considered simultaneously by the decoder. For syndromes of two different colors u and v, we can introduce a boundary-aware distance metric which is computed from the shortest path that connects these two syndromes through the boundaries. If the two such syndromes have to be matched, this is done by a correction string that appropriately changes color at this boundary.

Algorithm 2. Just-in-time RG decoder in the presence of a boundary or a domain wall $\mathcal{Y}$.

Initialize empty Σ_0. At each time step $t \geq 0$
1: Determine $\mathcal{M}_t$ from m_t and C_{t-1}.
2: Update $\Sigma_t \to \Sigma_t \oplus \mathcal{M}_t$;
3: Initialize C_t to be empty and the set $s = \Sigma_t$.
4: For each $0 \leq k \leq k_m$:
- 3.1: remove all points $u = (u_1, u_2, u_t)$ in s, such that $t - u_t < 2^k$;
- 3.2: remove each $u \in s$ from s and from Σ_t, such that $d(u, \mathcal{Y}) \leq 2^k$, if the current coordinate u is outside the spatial domain of the TQD at time t;
- 3.3: for any remaining $u \in s$ such that $d(u, \mathcal{Y}) \leq 2^k$, find any shortest path for the Pauli correction $C_t^{(k),i}$ matching the spatial coordinate of u with the boundary at the current time $\mathcal{Y}_t$ and remove u from s and from Σ_t;
- 3.4: for each $u \in s$, if $u' \in s$ exists such that $d(u, u') \leq 2^k$, find any shortest path for the Pauli correction $C_t^{(k),i}$ matching their spatial coordinates. Update $C_t \to C_t \cdot C_t^{(k),i}$ and remove u, u' from s and from Σ_t;

5: Pass the correction string C_t to the circuit.

3. Putting it all together

The fault tolerance of any full protocol comes from combining all the aspects discussed above as well as a few additional points that we address here.

First, we discuss what happens in the charge sector. There, we, as usual, record the measurement outcomes until the end of the protocol. Afterward, we run the usual RG decoder that must be modified to take into account the possibility of matching to the boundaries and domain walls. Similarly to the case with the trivial protocol, for large enough Q, the regions to be corrected form a cluster decomposition such that the RG decoder must succeed.

Another point is that the logical error [see Eq. (81)] in actual protocols is suppressed as $(p/p_c)^{O(d^n)}$, where d is now the length of the *shortest nontrivial homology class of charge or flux worldlines in spacetime*. This is because we have to compare the size of the largest cluster in the cluster decomposition not only with the spatial dimensions of the manifold, but also with the smallest distance between a pair of nontrivial boundaries in spacetime. Specifically, this is the distance between two points that, if they are connected by a cluster of error, affect the logical action of the protocol.

Generally, for the protocols in Sec. IV, all characteristic dimensions in spacetime geometry have to be $O(L)$, which then gives $d = O(L)$.

Additionally, in some of the protocols (namely, the logical $\overline{CZ}$ and $\overline{XS}$ measurements), the measurement outcomes themselves are used to detect nontrivial homology classes associated with one type of the charge worldlines, such as C_g. A protocol experiences a logical failure if we cannot determine the homology class reliably. However, this is the same as the failure of the error correction in the respective charge sector. Thus, the fault tolerance of global measurements (or, in other words, of determining the charge under the global symmetry that is being gauged in the protocol) is the same as fault tolerance in the respective charge sector.

And finally, since the protocols always end and begin in an Abelian phase, the fault-tolerant initialization and readout is the same as for a stand-alone Abelian phase, modulo the need to perform decoding and apply corrections immediately before the implementation of the TQD phase in the protocol begins.

D. MBQC-like protocols

In the previous subsections, we have restricted ourselves to circuits where the measured flux worldlines can propagate only in the time direction. We now briefly discuss how to generalize this to other MBQC-like and Floquet-like circuits where flux worldlines can propagate in any direction. Naively, this constitutes an issue since, even in the absence of noise, we measure a random and dense configuration of flux worldlines. Since charge worldlines can terminate on the flux worldlines, the fault tolerance proof of the previous subsection does not go through.

Before we show how to solve this problem, we describe an alternative way to perform the corrections for circuits that we discussed up till now, such as in Sec. III or V A 2, where fluxes propagate *only* in the time direction in the absence of noise. Instead of applying a (Pauli-X) string operator C_t that is determined at step 5 of the just-in-time decoding Algorithm 1, we instead commute the Pauli-X correction string until the end of the protocol through all following operations in the circuit. Conjugating the CCZ gates with X yields additional CZ gates in the circuit, and conjugating the Z-type measurements at the end points of the string C_t will flip all future measurement outcomes at these end points to -1. In the path integral, this is the same as inserting a path-integral membrane (by design) as depicted in Fig. 18, which we refer to as inserting the *gauge-fixing membrane*.

While this different way of implementing corrections may seem more complicated, it is precisely the way MBQC-like and Floquet-like protocols must work, such as these derived from considering $(3+0)$D circuits as discussed in Sec. VI and in Refs. [22,23]. In such protocols, if there is no noise, the measured flux configurations must

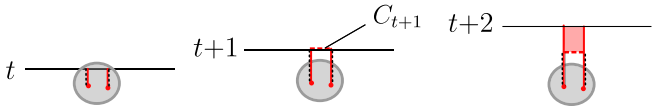


FIG. 18. Alternative way of using the output C_{t+1} of the just-in-time decoder (applied to a configuration of two syndrome points caused by an error region in gray) to perform corrections. Instead of applying a Pauli-X string along C_{t+1}, we insert the gauge-fixing membrane, as shaded in red, which is equivalent to changing the circuit such that it is conjugated by the Pauli-X string associated with correction C_{t+1} starting at time $t+1$ and onward. After this, we keep measuring -1 outcomes at the end points of C_{t+1}; that is, the flux worldlines continue indefinitely in time direction—however, their properties are "mitigated" by the gauge-fixing membrane.

be all closed loops. In the absence of the noise, all we need to do is to choose any gauge-fixing membrane whose boundary coincides with the measured flux configuration. In the circuit, the gauge-fixing membrane corresponds to conjugating all CCZ gates with Pauli-X operators.

Now, if there is noise in the circuit, then the measured flux worldlines W_f do not have to be closed, and so, naively, we do not know where to insert the gauge-fixing membranes. In order to determine where the gauge-fixing membrane needs to be inserted, we need to find a loop-closing "fix" string, which we obtain in a just-in-time way as a sum of two parts, $\tilde{C}_{f,t} + C_t$. The fix string has the same end points as W_f, $\partial(W_f + \tilde{C}_{f,t} + C_t) = 0$, and we choose the gauge-fixing membrane such that its boundary is given by $W_f + \tilde{C}_{f,t} + C_t$.

We now describe how $\tilde{C}_{f,t}$ and C_t are obtained using the just-in-time decoder. C_t is given by the correction strings which are the output of the just-in-time decoder in Algorithm 1 (noting that we do not apply a Pauli-X string at C_t here). $\tilde{C}_{f,t}$ consists of timelike flux lines that begin at every flux syndrome point unmatched by the decoder and propagate straight in the time direction. When the decoder pairs two syndrome points and outputs an appropriate string C_t, this closes off the $\tilde{C}_f$ lines from the past time steps originating from these syndrome points. This is illustrated in Fig. 19.

Finally, we note that, in the case where the measured flux worldlines can only propagate in t direction, $\tilde{C}_f$ always coincides with the measured flux worldlines W_f originating from the syndrome points, so we have to start filling in the gauge-fixing membrane only after adding the C_t part of the correction, as shown in Fig. 18.

E. Comments on decoding logical protocols in practice

So far, we considered one particular way of performing decoding that was chosen specifically to make the proof simple. This leaves vast space for improvement in terms of

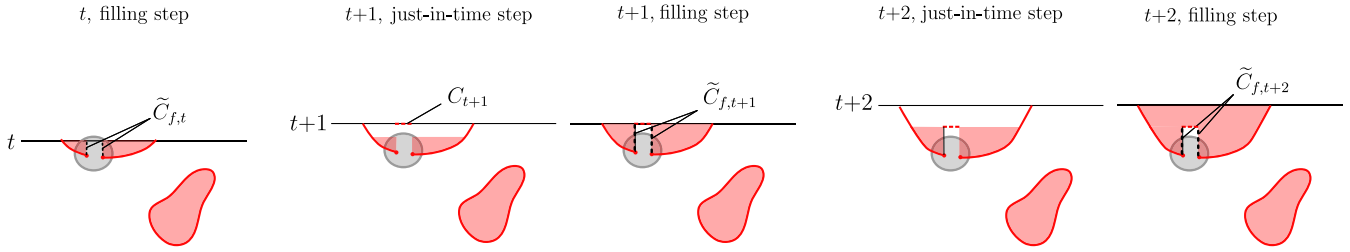


FIG. 19. A schematic showing an example of the gauge-fixing membrane filling (shown as red shading) in the presence of noise (shown as a gray region). At each time step t, the decoder finds a string $C_t + \tilde{C}_{f,t}$ that closes off the measured (up to time t) flux configuration W_f, where C_t (shown as red dashed lines) is the output of Algorithm 1 and $\tilde{C}_{f,t}$ (shown as black dashed lines) consists of lines propagating in time direction from any new syndrome point, until it is paired with another syndrome by the decoder and closed off by a C_t string. Note that, in contrast to discussion earlier in this section, we do not actually apply Pauli-X strings at C_t, and we do not measure -1 outcomes at $\tilde{C}_{f,t}$.

performance. Here, we propose a more efficient decoding scheme and point out particular areas where improvement is the easiest to achieve.

Below, we propose a minimum-weight matching (MWPM) algorithm for the flux sector that is broadly inspired by Bombin's version of the just-in-time algorithm [22]. For simplicity, we show its version in the absence of boundaries or domain walls. The modification of the algorithm in the presence of domain walls and boundaries is in line with the discussion in previous subsections. In addition, in the presence of a spatial domain wall, we can use global decoding on the Abelian phase side that appropriately matches defects to the domain wall, followed by the MWPM domain wall-aware decoding on the TQD side from the domain wall. We do not prove the fault tolerance of this algorithm; in comparison to the proof presented earlier, proving fault tolerance of the MWPM-based decoder would require taking into account the possibility that the decoder matches separate noise clusters (the neighborhood the noise clusters together with their corrections then becomes a source of additional Abelian noise) and estimating the distribution of the outcome in spacetime.

Similarly, the charge sector can be decoded by a boundary- and domain-wall aware MWPM at the end of the protocol. In addition, the performance can be improved by explicitly taking into account in the decoding that the charge worldlines can terminate on the flux worldlines, namely, treating the measured flux worldlines W_f as the heralded locations where we know the charge strings can terminate.

Apart from the choice of the decoding strategy, the actual performance of any decoder is sensitive to the particular microscopic circuit. It would be interesting to compare different circuits in Sec. V to each other. Additionally, the search for decoding heuristics that will lead to especially high thresholds is an interesting direction for future work.

We remark that, for numerical benchmarking of the non-Clifford protocols, there is no need to perform a full simulation of the circuit involving non-Clifford physical operations. Given the noise with sufficiently local supports of possible channels, we can compute an effective distribution over flux and charge configurations created by the noise directly relating to measurement outcomes, which are then used by the decoder. Given a (sampled) fault configuration, this distribution can be efficiently computed and then sampled from in the simulation. This computation has to be performed only once for every fault type, yielding a scalable simulation scheme. In addition, for certain noise models, e.g., i.i.d. single-site X and Z noise, together with probabilistic noise on the measurement outcomes, it is possible to obtain an analytic expression for the effective probability distribution of charges and fluxes.

Algorithm 3. Just-in-time matching decoder.

Initialize empty Σ_0, Σ_0'. At each time step $t \geq 0$
1: Determine $\mathcal{M}_t$ from m_t and C_{t-1}.
2: Update $\Sigma_t \to \Sigma_t \oplus \mathcal{M}_t$;
3: Initialize C_t and S_t to be empty;
4: Run MWPM on Σ_t with smooth boundary at time t. Record the spacetime matching outcome in S_t;
5: For every pair $u, u' \in \Sigma_t$ matched in spacetime by S_t, run another MWPM round for their spatial coordinates and add the outcome to C_t; remove u, u' from Σ_t;
6: Pass the correction C_t to the circuit.

VIII. DISCUSSION

In this work, we have shown how to perform non-Clifford gates by introducing domain walls between toric or surface codes and the non-Abelian type-III twisted quantum double phase. This offers an alternative implementation of universal fault-tolerant quantum computation in two dimensions that differs from previous proposals. Given the compatibility of our protocols with the surface code architectures, our logic gates can directly supplement current experimental efforts to realize universal quantum computation with surface codes.

We have presented a number of different specific microscopic implementations of our logical protocols. It will be valuable to further adapt these implementations to run on realistic hardware [55,56]. The path-integral framework we have proposed gives us some flexibility to redesign circuits to satisfy hardware constraints. In addition, further work and numerical testing of the just-in-time decoder is essential, as well as careful evaluation of the resource cost of our proposals at a target logical error rate.

An important question that remains open is: What other fault-tolerant logic gates can be achieved by introducing domain walls between different topological phases? In this direction, the fault-tolerant non-Clifford gates we have introduced can be, in principle, generalized to all solvable non-Abelian anyon theories. An important related question is whether a version of just-in-time decoding can be used to achieve fault tolerance in these models in the presence of circuit-level noise. This presents an opportunity to design fault-tolerant analogs of recent finite-depth adaptive local unitary state preparation results for solvable anyons [60,69,111–113]. Another long-standing open question is the circuit-level fault tolerance of general non-Abelian anyon theories; it would be interesting to see if some of the ideas in the just-in-time decoding approach can be used to make progress there.

Our results have been derived through a number of different approaches. These include the topological path-integral framework, code deformation, and gauging logical operators via measurements. We have also drawn the connection between 2D non-Abelian topological codes and 3D topological codes with transversal non-Clifford gates, thereby illuminating the physical mechanism behind prior proposals to realize non-Clifford logic gates in two dimensions [22,23]. It would be interesting if the tools we have developed can be extended to more general quantum error-correcting codes, such as different classes of quantum low-density parity-check codes, to find new ways of performing non-Clifford gates on code blocks encoding a large number of logical qubits. We hope that the present work will serve as a guiding example toward a generalized framework for a universal computation with low-overhead quantum error-correcting codes.

Note added—Related work [114] appeared online recently.

ACKNOWLEDGMENTS

We are grateful to S. Balasubramanian, M. Barkeshli, P. Bonderson, J. Bridgeman, H. Dreyer, J. Garre Rubio, A. Lyons, N. Schuch, N. Tantivasadakarn, F. Verstraete, J. Wang, and G. Zhu for useful discussions. We also thank M. Kesselring for fruitful discussions and for suggesting the 3D brickwork geometry used in Sec. VI. B. J. B. thanks the Center for Quantum Devices at the University of Copenhagen for their hospitality. M. D. was supported by the Walter Burke Institute for Theoretical Physics at Caltech. M. D. acknowledges funding provided by the Institute for Quantum Information and Matter, an NSF Physics Frontiers Center (NSF Grant No. PHY-2317110). Part of this work was done while D. J. W. was visiting the Simons Institute for the Theory of Computing. D. J. W. was supported in part by the Australian Research Council Discovery Early Career Research Award (DE220100625). A. B. was supported by the U.S. Army Research Laboratory and the U.S. Army Research Office under Contract No. W911NF2310255. A. B. and B. J. B. both received support from the U.S. Department of Energy, Office of Science, National Quantum Information Science Research Centers, and the Co-design Center for Quantum Advantage (C2QA) under Contract No. DE-SC0012704. J. C. M. is supported by the DFG (CRC 183). M. W. is supported by the Engineering and Physical Sciences Research Council (Grants No. EP/W032635/1 and No. EP/S005021/1).

DATA AVAILABILITY

There are no publicly available research data or software supporting this manuscript. Requests for further information or data should be sent to the authors.

APPENDIX A: LOOP-SUM PICTURE FOR THE TQD MODEL

Here, we review the *loop-sum picture* for the TQD ground states, which is useful for developing a qualitative understanding of our protocols. The loop-sum picture presented here can be understood as a continuum version of the lattice model, which nevertheless admits a direct relation to the microscopic realization, where configurations of qubits of any given color that are in the $|1\rangle$ state form closed loops on the lattice (here, we refer to states that appear in an expansion of the ground state in the computational basis). We start from the well-known description of the code states of the standard toric code phase in the loop-sum picture [2,4].

The toric code can be defined on an arbitrary two-dimensional cellulation with one qubit on each edge. A configuration of qubits in the computational basis corresponds to marking all qubits on the Poincaré dual cellulation that are in the $|1\rangle$ state (we work with the Poincaré dual cellulation unless specified otherwise). The plaquette stabilizers of the toric code ensure that the code states have an even number of $|1\rangle$-state qubits around every plaquette, and, thus, only closed-loop configurations appear in the expansion of the code state. The vertex stabilizers ensure that all possible closed-loop configurations appear with equal amplitudes in the expansion of the code state. More accurately, all possible closed-loop configurations *within the same cohomology class* appear with equal amplitudes in the expansion of the code state. Thus, the code states of the toric code in the logical-Z basis are equal-weight superpositions of closed-loop configurations, as shown in Fig. 20, and otherwise are linear combinations thereof.

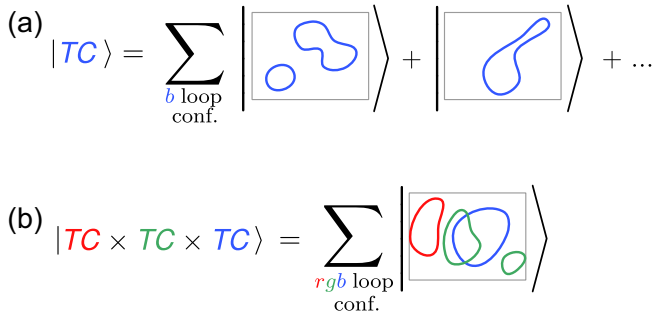


FIG. 20. Schematic loop-sum picture for a ground state of (a) a single toric code and (b) three copies of the toric code labeled red, green, and blue (denoted "TC" in the appropriate color). The three copies of toric code do not have to be supported on the same microscopic lattice but can be defined on three different superimposed lattices.

The non-Abelian model introduced in Sec. III is similar to three copies of the toric code defined on three superimposed triangular lattices shown in Fig. 20 with the difference that the states do not form an equal-weight superposition (here, by weight we mean the coefficient in front of the state). Instead, due to the "twist," there is a relative phase between any two configurations [7,9], as shown in Fig. 21. If two configurations differ by moving a string of one color over an intersection of strings of the other two colors, their amplitudes differ by a factor of -1, as shown in Fig. 21(b).

We show three superimposed triangular lattices in Fig. 22(a) as well as the Poincaré dual of the blue sublattice. In Fig. 22(b), we show a closed-loop configuration of three colors. In Fig. 22(b), we illustrate how the lattice model in Sec. III corresponds to the closed-loop picture. The stabilizers can be derived from the condition that they stabilize the code state described by the superposition

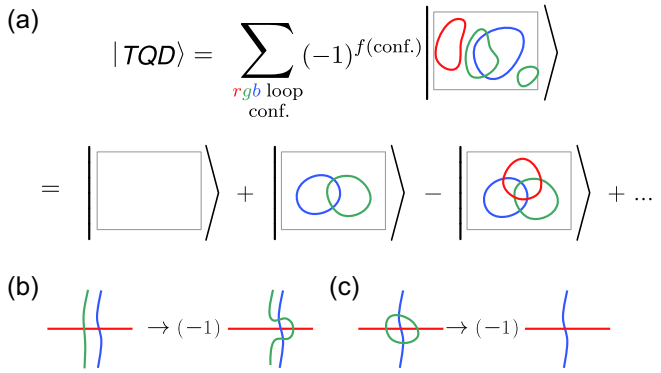


FIG. 21. (a) Schematic loop-sum picture for one of the ground states of the type-III twisted quantum double model (denoted "TQD"). It is a superposition of all three-color loop configurations with appropriate complex phases determined by the type-III 3-cocycle function $\omega_{\text{III}}(\text{conf}) = (-1)^{f(\text{conf})}$. The rule that determines the relative phase of the configurations entering the superposition is shown in (b).

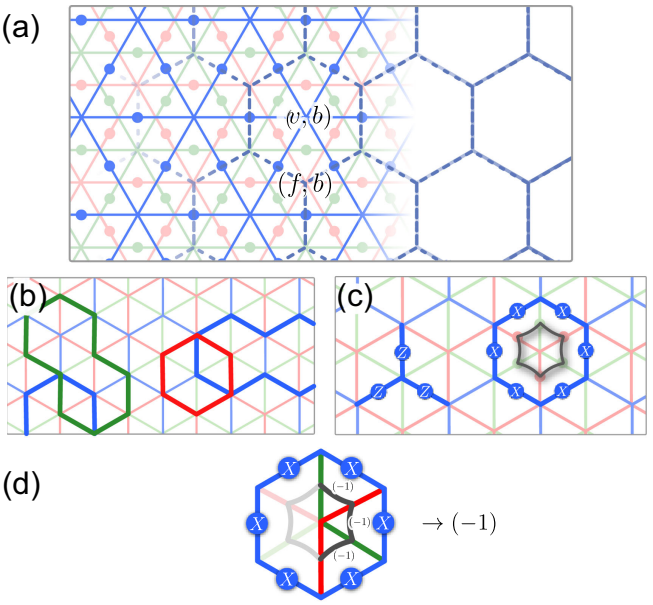


FIG. 22. (a) The primal lattice (left) used to define the type-III TQD model and the Poincaré dual of the blue sublattice (right). (b) A three-colored loop configuration shown on the dual lattice. (c) Stabilizers of the TQD shown on the dual lattice. (d) A blue vertex stabilizer on the dual lattice showing the (-1) factor due to a red-green loop crossing within its support.

above. Similar to the toric code, the plaquette stabilizers ensure that the configurations appearing in the code state are closed-loop configurations. However, to enforce that the relative amplitudes are changed by the necessary factors, the vertex stabilizers must be modified. We now verify that the choice shown in Fig. 22(c) (which are the stabilizers from Sec. III shown on the dual lattice) produces the necessary phase factors. Consider the vertex stabilizer in Eq. (2), where the color c is chosen to be blue without loss of generality. When this operator is applied, the Pauli-X component of the stabilizer adds a small blue loop to the closed-loop configuration. The CZ component of the operator evaluated on a state in the computational basis yields $(-1)^n$, where n is the number of green-red intersections inside the associated blue loop. An example of this is shown in Fig. 22(d), where adding a small blue loop around a single red-green intersection yields a factor of (-1). This indeed verifies the correct loop statistics.

APPENDIX B: ELEMENTARY EXCITATIONS OF THE TQD MODEL

In this appendix, we provide an informal overview of the excitations of the type-III twisted quantum double which is the non-Abelian model featured in this work. In the following appendix, we show how to obtain these excitations from gauging a $\mathbb{Z}_2$ symmetry of a pair of toric codes and discuss more of their properties. For a more in-depth discussion, see Refs. [10,56,115].

The excitations of the type-III cocycle twisted TQD (which is isomorphic to the D_4 quantum double [10]) can be labeled similarly to three uncoupled copies of the toric code (which we associate with red, green, and blue colors). The generating set is also given by three charges and three fluxes, except that the fluxes are now non-Abelian. There are a total of 22 anyons in this model.

The electric charges $e_{r,g,b}$ are independent Abelian $\mathbb{Z}_2$ bosons. Together, they generate eight Abelian superselection sectors, namely, $\{1, e_r, e_g, e_b, e_r e_g, e_r e_b, e_g e_b, e_r e_g e_b\}$. The flux excitations are non-Abelian bosons of quantum dimension $d = 2$, defining six sectors $\{m_r, m_g, m_b, m_{rg}, m_{rb}, m_{gb}\}$. m_r braids nontrivially with e_r (with a phase -1) but also with other fluxes.

There are also six fermions $\{f_r, f_g, f_b, f_{rg}, f_{rb}, f_{gb}\}$ that are fusion products as $m_r \times e_r = f_r$ and $m_{rg} \times e_r = m_{rg} \times e_g = f_{rg}$. Finally, there is a semion s (which is contained in the fusion product of three fluxes of different colors) and an antisemion $\bar{s}$.

Below are a pair of representative fusion rules for the non-Abelian bosons:

$$m_r \times e_{g,b} = m_r,$$
$$m_r \times m_r = 1 + e_g + e_b + e_g e_b, \tag{B1}$$

and similarly for other colors. The vacuum fusion channel for a pair of, say, m_r fluxes changes to e_g upon braiding with an m_b flux.

In a circuit realization of topological phases by measurements, if all the measurement outcomes are $+1$, the state is excitation-free. If some of the measurements are violated, an excited state of the model is realized. However, generically, these excitations are not in one-to-one correspondence with specific anyons—instead, an excitation can be a superposition of several kinds of anyons in the model. In the models from Sec. V, the Clifford (vertex) stabilizers can be associated with e anyons (charges) which are Abelian. However, the states that violate Pauli (usually, Z-type) plaquette stabilizers correspond to superpositions of non-Abelian anyons. Therefore, we call the violated plaquette measurements "flux defects" (as opposed to "excitations" or "anyons") which we use in the path-integral framework as well as analysis of fault tolerance.

In the closed-loop path-integral picture, charge defects are in one-to-one correspondence with Abelian anyons, while flux defects correspond to the loop-shaped termination lines of open membranes. However, with some changes we can draw a connection between the flux defects and the non-Abelian anyon excitations. A depiction of a non-Abelian anyon excitation in the path-integral picture is shown in Fig. 23. Figure 23(a) shows a blue m_b anyon worldline in spacetime that goes up to time t_1 and the associated (deformable) blue membrane. There is a green

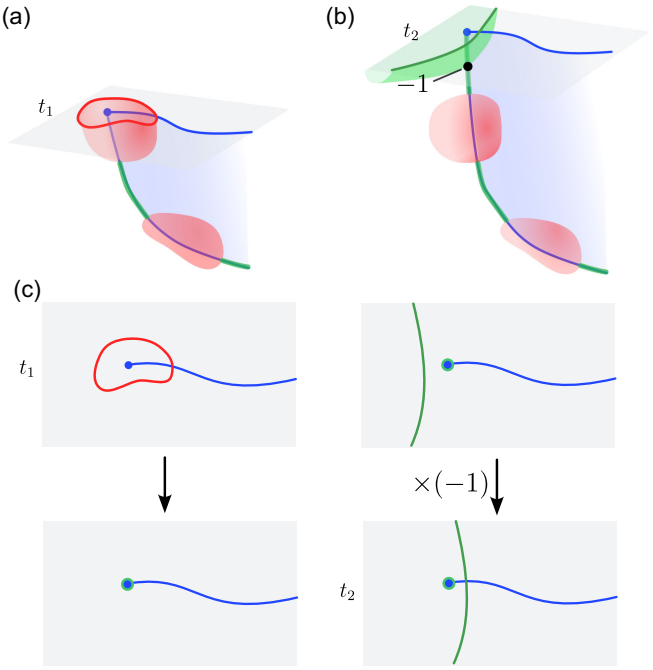


FIG. 23. (a) A section taken at time t_1 showing a blue m_b anyon worldline in spacetime with an open membrane terminating on it. At intersection points with closed red membranes, additional green strings end that produce -1 phases when intersecting with green membranes as shown in (b). The spacetime region in (b) displays the same configuration until a later time $t_2 > t_1$. (c) shows four time slices between t_1 and t_2 of the spacetime configuration in (b).

string along the worldline that behaves as a *fluctuating charge* of the third (green) color. This string has the same consequence as the green charge worldline: It introduces a -1 weight for each intersection with a green membrane. This is shown in Fig. 23(b), which depicts a time slice at a later time $t_2 > t_1$. However, the end points and the number of such strings are not fixed in the sum over all closed configurations of red membranes in the path integral, which results in different string distributions. The same property holds for m_b excitation worldline and green closed membranes upon exchanging red and green colors. These properties explain the fusion and braiding statistics of the non-Abelian fluxes in the twisted quantum double.

The presence of an open blue membrane can make the path integral non-gauge-invariant. However, for the m_b excitation worldline, this is fixed thanks to the decoration described above. The -1 phase due to a possible triple intersection with an open blue membrane is always canceled by the extra -1 factor coming from intersections between closed membranes and the strings along the blue anyon worldline, resulting in no overall ambiguity.

We refer to the appendix in Ref. [56] for an interpretation of non-Abelian excitations via gauging one-dimensional symmetry-protected topological states.

APPENDIX C: GAUGING CLIFFORD SYMMETRIES

In this section, we first review the algebraic description of gauging an Abelian symmetry of an Abelian anyon theory. In general, this can lead to a gauged theory that includes non-Abelian anyons. We show explicitly how gauging a $\mathbb{Z}_2$ symmetry on two copies of the toric code can produce D_4 anyons, which is equivalent to the anyon model of the type-III non-Abelian TQD. We conclude the example by showing how to identify the twisted quantum double model introduced in Sec. III as a gauged model on the level of the stabilizers that define the code. This highlights the physics underlying the existing protocols for non-Clifford logical operations on 2D topological codes and enables the design of new procedures.

1. Gauging Abelian symmetries in Abelian anyon models

The anyon theory realized by a Pauli stabilizer model is always Abelian [116–118]. The anyon labels themselves form an Abelian group A where the fusion of the anyons is given by the group multiplication. The braiding is defined by the *topological spin* $\theta: A \to U(1)$ that assigns a complex phase to each anyon type and captures how the wave function of a state with an anyon transforms under a 2π rotation of that anyon. The trivial anyon, labeled by the identity element $1_A \in A$, has topological spin $\theta_{1_A} = 1$. For any a, if $\theta_a = 1$, we say a is a boson, and if $\theta_a = -1$, we say a is a fermion.

We consider an Abelian symmetry (automorphism) of an Abelian anyon model that acts like some group G. For each group element $g \in G$, there is a map $\rho_g: A \to A$ such that $\rho_g(1_A) = 1_A$ and $\rho_g[\rho_h(a)] = \rho_{gh}(a)\ \forall\ g, h \in G, a \in A$. Moreover, an anyon automorphism has to preserve the braiding data, i.e., $\theta_{\rho_g(a)} = \theta_a\ \forall\ g \in G, a \in A$ and be compatible with the fusion, $\rho_g(ab) = \rho_g(a)\rho_g(b)\ \forall\ g \in G; a, b \in A$.

Such a symmetry can be gauged. Before gauging G, we need to incorporate the *twist defects* associated to the symmetry G into the theory. In the literature, this procedure is referred to as constructing the "G extension of A" [6]. To each group element $g \in G$, we can associate a twist. In the stabilizer model, twist defects can be introduced at end points of an (invertible) domain wall associated with the corresponding group element; see Fig. 24. The twist defect can *condense* some of the anyons. Namely, for every fermion c of the form $c = a \times b$ with $\rho_g(b) = a^{-1}$, we can create a single c anyon in the vicinity of the twist defect with a local operator; see Fig. 24. For example, the twist defect in the toric code that exchanges e and m anyons leaves their composite, the fermion f, invariant. In addition, we can apply a local operator close to the twist that creates a single fermion, which is not possible without the twist.

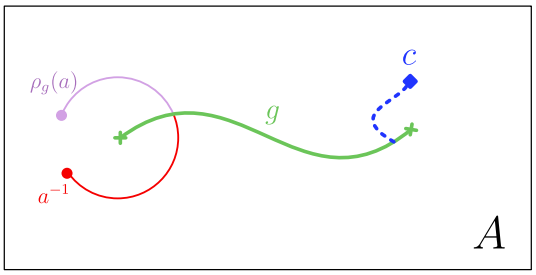


FIG. 24. To any symmetry operation $g \in Aut(A)$ in an Abelian anyon theory A, there is an associated domain wall (green). Terminating this domain wall creates twist defects at the end points, indicated by crosses. Close to the twist defects, anyons that where previously inequivalent can become identified. On the left, we show how the anyon $\rho_g(a)$ can be created together with an a^{-1} anyon close to the defect by applying an operator that crosses the domain wall. Without the defect, this would not be possible if $\rho_g(a) \neq a$. On the right, we show how an individual anyon can be created close to the defect with a local operator if it is the fusion product of $a \times b = c$, where $\rho_g(b) = a^{-1}$. This implies that in the vicinity of the twist the total charge of c (possibly fermions) is not conserved; i.e., they can condense.

We label the subgroup of anyons that can condense at a g twist by C_g. In the case where all anyons are Abelian the *quantum dimension* of the twist defect g is equal to [6]

$$d_g = \sqrt{|C_g|}. \tag{C1}$$

Since C_g is a subgroup of A, we can take the quotient A/C_g. This groups the anyons of the original ungauged model A into cosets of C_g. To simplify the notation, we label a coset $aC_g \in A/C_g$ by a_g.

Close to the twist defect, anyon a can be transformed into $\rho_g(a)$ by moving it once around the twist. Clearly, this can be done by applying a local operator, and, hence, this makes a and $\rho_g(a)$ topologically indistinguishable in the vicinity of the twist. This decomposes A/C_g into G orbits $[a_g] = \{\rho_x(a)C_g | x \in G\}$. These orbits are the (isomorphism classes of) simple objects in the structure that described the symmetric model before gauging. Similar to anyons in the parent model, before gauging the symmetry, the simple objects can be equipped with a (G-graded) *fusion* and *braiding*. This depends on the details of how the symmetry acts on the anyons of the model. In the general case, these additional data can lead to an *anomaly* [6] that prevents gauging the symmetry. However, in the following, we assume that the symmetry is anomaly-free, as it is for the symmetries that we gauge in this paper. More specifically, the anomaly-free condition is guaranteed when the symmetry action can be represented by an invertible domain wall within a lattice model realizing the anyon theory, for example, by applying a transversal gate to parts of the model. Thus, at a microscopic level, this is true for any

logical gate that has a (nonanomalous) finite-depth local unitary implementation on the associated code.

Given the data described above, we can construct the labels for the anyons of the gauged model in three steps:

(1) Decompose $A_g := A/C_g$ into G orbits $\{[a_g]|a_g \in A_g\}$.
(2) For each G orbit $[a_g]$ identify the *stabilizer group* $S_{a_g} = \{h \in G|\rho_h(a_g) = a_g\} \leq G$ of one of the cosets in the orbit $a_g \in [a_g]$.
(3) Find the irreducible representations of S_{a_g} for each orbit $[a_g]$.

Given these structures, the anyons in the gauged model are labeled by a triple

$$(g, [a_g], \Gamma_a), \tag{C2}$$

where $g \in G$, $[a_g] \subseteq A_g$ a G orbit, and Γ_a an irreducible representation of S_{a_g}. Note that, since S_{a_g} is Abelian, Γ_a can be viewed as group elements of S_{a_g}.

To complete the data of the gauged anyon model, we have to also determine the fusion and braiding data. Both are directly inherited from the data of the symmetry group and associated twist defects. The quantum dimensions are given by the formula

$$d_{(g,[a_g],\Gamma_a)} = d_g|[a_g]|, \tag{C3}$$

where $d_g = \sqrt{|C_g|}$ is the quantum dimension of a g twist in the ungauged theory. Note that this simple formula holds only for Abelian G. For non-Abelian G, additional factors appear, related to the conjugacy classes that define the anyon as well as to higher-dimensional irreps of the stabilizer groups S_{a_g}.

Next, we discuss the topological spins of the anyons in the gauged theory. They are built from the topological spin of the g twist paired with an anyon in the $[a_g]$ orbit in the ungauged theory, θ_{g,a_g}, and the irrep Γ_a. Note that this quantity depends on the *symmetry fractionalization* [6,119] pattern in the ungauged, symmetric, anyon theory. We do not aim to give a complete introduction here but merely show how to, given the complete data associated to the symmetry action ρ within the ungauged theory, obtain the algebraic data describing the gauged anyon theory.

In particular, we can consider the group character function $\chi_{\Gamma_a}: S_{a_g} \to U(1), g \mapsto \mathrm{Tr}[\Gamma_a(g)]$. The topological spin of the related anyon in the gauged theory is

$$\theta_{(g,[a_g],\Gamma_a)} = \theta_{g,a_g}\chi_{\Gamma_a}(g). \tag{C4}$$

The remaining data of the gauged model, such as the S-matrix and the fusion data, are inherited from the data of the G extension of the ungauged model and depend on the details of how G acts on A. In the anomaly-free case, this can always be lifted unambiguously to the anyons in the gauged model and the procedure outlined above indeed constructs a proper anyon model.

Upon gauging, the anyons that have quantum dimension larger than 1 are non-Abelian [which can be determined from Eq. (C3)]. For a more in-depth discussion of non-Abelian groups and anyons, we refer the interested reader to Refs. [6,119,120].

2. Non-Abelian TQD from gauging $\mathbb{Z}_2$ symmetry in toric codes

Two toric codes are jointly described by the fusion group $A_{TC^{\times 2}} = \langle \{e_i, m_i | i = 1, 2\} \rangle \simeq \mathbb{Z}_2^{\times 4}$ with topological twist $\theta_{(\mathbf{e}^\mathbf{a}, \mathbf{m}^\mathbf{b})} = (-1)^{a_1 b_1 + a_2 b_2}$, where $(\mathbf{e}^\mathbf{a}, \mathbf{m}^\mathbf{b}) = e_1^{a_1} e_2^{a_2} m_1^{b_1} m_2^{b_2}$ with $a_i, b_i \in \{0, 1\}$ labels an anyon in $A_{TC^{\times 2}}$.

We consider a permutation symmetry of $A_{TC^{\times 2}}$ defined by its action on the generators

$$\begin{aligned} e_i &\mapsto e_i, \quad i = 1, 2, \\ m_1 &\mapsto m_1 e_2, \qquad m_2 \mapsto m_2 e_1. \end{aligned} \tag{C5}$$

It is straightforward to check that it preserves all the topological spins of the model and squares to the trivial permutation. As such, it generates a $\mathbb{Z}_2$ symmetry of the anyon model of the two toric codes.

To obtain the gauged theory, we first construct the G extension of $A_{TC^{\times 2}}$. It has two sectors, $A_0 = A_{TC^{\times 2}}$ and $A_1 = A_{TC^{\times 2}}/C_1$, where C_1 is the subgroup of bosons that can condense at the nontrivial twist defect associated to the $\mathbb{Z}_2$ symmetry. We find that $C_1 = \langle e_1, e_2 \rangle$; see Fig. 25. This leads to $A_1 = \{C_1, m_1 C_1, m_2 C_1, m_1 m_2 C_1\}$. Decomposing both A_0 and A_1 into G orbits leads to

$$\begin{aligned} A_0 = &\{1\} \sqcup \{e_1\} \sqcup \{e_2\} \sqcup \{e_1 e_2\} \sqcup \{m_1, m_1 e_2\} \\ &\sqcup \{m_2, e_1 m_2\} \sqcup \{m_1 m_2, f_1 f_2\} \sqcup \{m_1 f_2, f_1 m_2\} \\ &\sqcup \{f_1, f_1 e_2\} \sqcup \{f_2, e_1 f_2\} \quad \text{and} \end{aligned} \tag{C6a}$$

$$A_1 = \{C_1\} \sqcup \{m_1 C_1\} \sqcup \{m_2 C_1\} \sqcup \{m_1 m_2 C_1\}. \tag{C6b}$$

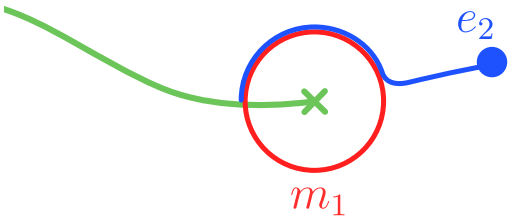


FIG. 25. Abstract depiction of the twist defect associated to the $\mathbb{Z}_2$ symmetry in two layers of toric code (shown in green). We illustrate an anyon from the subgroup of Abelian anyons $C_1 = \langle e_1, e_2 \rangle$ that can condense at the twist defect. There exists a local string operator that creates a single e_2 anyon, indicated with the blue dot. The operator creating the anyon can be interpreted as creating a pair of m_1 anyons (shown in red) and moving one of them over the domain wall attached to the twist defect which acts on as $m_1 \mapsto m_1 e_2$. Fusing it again with the first m_1 anyon (closing of the red circle) leaves a single e_2 anyon behind.

A_0 decomposes into ten orbits and A_1 into four orbits. The only possible stabilizer groups are the trivial group $\mathbb{Z}_1 \leq \mathbb{Z}_2$ or the full symmetry group $\mathbb{Z}_2$. The stabilizer groups label the additional "symmetry charges" (irreducible representations) carried by the anyons. The eight sets containing a single element admit the full stabilizer group, and the six order-2 sets admit the trivial stabilizer group. In total, this yields $2 \cdot 8 + 6 = 22$ anyons in the gauged theory. We label an anyon associated to sector $A_{0,1}$, an orbit $\{a\}$, and irrep $j = 0,\ 1$ as $(i, [a], j)$. The quantum dimensions are obtained by taking the product of the quantum dimension of the associated twist and the cardinality of the orbits. The Abelian anyons, with quantum dimension 1, are

$$\begin{aligned}&(0,[1],0),(0,[1],1),\\&(0,[e_1],0),(0,[e_1],1),\\&(0,[e_2],0),(0,[e_2],1),\\&(0,[e_1e_2],0),(0,[e_1e_2],1).\end{aligned} \tag{C7}$$

They are closed under fusion and form the group $\mathbb{Z}_2^{\times 3}$. In color representation, they can be identified with (for example) $\{1, e_g, e_r, e_re_g, e_b, e_be_g, e_re_b, e_re_be_g\}$, though this is not a unique association; see Appendix B. All other anyons have quantum dimension 2 and, hence, are non-Abelian. This is either because their orbits are of size 2 or because their associated twist defect has quantum dimension 2.

The topological spins of the 22 anyons can also be inferred directly from the data of the anyons and twists that enter an anyon of the gauged model. We find that all Abelian anyons are bosons. Moreover, there are three non-Abelian bosons and three non-Abelian fermions associated to anyons in the ungauged toric codes in the trivial sector A_0. Three of the non-Abelian anyons associated to the sector A_1 are bosons:

$$\begin{aligned}&(1,C_1,0),\\&(1,m_1C_1,0),\\&(1,m_2C_1,0),\end{aligned} \tag{C8}$$

which, in the color picture, are translated into $\{m_g, m_rm_g, m_bm_g\}$; see Appendix B. Three are fermions:

$$\begin{aligned}&(1,C_1,1),\\&(1,m_1C_1,1),\\&(1,m_2C_1,1),\end{aligned} \tag{C9}$$

translated into $\{f_g, m_rf_g, m_bf_g\}$—see Appendix B—and two are (anti)semions, carrying a topological spin of $\pm i$, namely,

$$\begin{aligned}&(1,m_1m_2C_1,0),\\&(1,m_1m_2C_1,1).\end{aligned} \tag{C10}$$

In the color picture, these are simply denoted as s and $\bar{s}$, respectively.

Additionally, the anyons inherit their fusion and braiding data from the G extension, i.e., the toric code anyon models together with the twist defects of the $\mathbb{Z}_2$ symmetry. The fusion and braiding data of the anyons of the gauged theory coincide with the anyons of a quantum double of $\mathbb{Z}_2^{\times 3}$ twisted by a type-III cocycle, respectively, the quantum double of the non-Abelian group D_4. We refer to Refs. [10,56,115] for a more in-depth discussion of the braiding and fusion data.

3. Implementation of gauging on-site symmetries

We now provide a brief overview of the gauging measurement procedure for a $\mathbb{Z}_2$ on-site symmetry, following Ref. [48].

We consider a symmetry action of the form

$$U = \prod_{v \in \mathcal{G}} U_v, \tag{C11}$$

which acts on the vertices of an oriented graph $\mathcal{G}$, equipped with a chosen $\mathbb{Z}_2$-cycle basis with elements labeled by plaquettes p. The graph should be chosen to admit a sparse cycle basis. The fault-tolerant gauging measurement procedure is performed as follows:

(1) Introduce an ancilla qubit initialized in the $|0\rangle$ state on each edge of the graph $\mathcal{G}$.
(2) Measure local star operators A_v, defined below, on every vertex $v \in \mathcal{G}$ (this condenses the domain walls associated with the symmetry and results in a deformed intermediate code).
(3) Repeat d rounds of syndrome extraction for the deformed code.
(4) Measure out the edge qubits in the Z basis (this ungauges the symmetry and brings us back to the original, undeformed code).
(5) Apply an Abelian correction operator C, defined below.

We now describe the deformed code. By construction, this code is LDPC even if the symmetry operator that is gauged has total extensive weight. The deformed code has stabilizer checks A_v, B_p, and $\{\tilde{s}_j\}$, which we refer to as star, plaquette, and deformed checks, respectively. The star checks are

$$A_v = U_v \prod_{e \in \delta v} X_e, \tag{C12}$$

where δ denotes a $\mathbb{Z}_2$ coboundary map on $\mathcal{G}$. The measured value of the symmetry operator U is given by the product of

the measurement outcomes of all vertex checks. The new plaquette checks are

$$B_p = \prod_{e \in \partial p} Z_e, \tag{C13}$$

where ∂ denotes a $\mathbb{Z}_2$ boundary map for $\mathcal{G}$. To describe the deformed checks $\{\tilde{s}_j\}$, we expand the original checks $\{s_j\}$ of the (nondeformed) code into a linear combination of operators with definite on-site symmetry charge μ, namely,

$$s_j = \sum_{\mu} c_j^{\mu} O_{\mu}, \tag{C14}$$

where c_j^{μ} are complex coefficients defining the expansion. The index μ runs over the possible charges with respect to each U_v independently. In fact, since $\{U_v\}_v$ are defined on vertices and generate a $\mathbb{Z}_2$ symmetry, and since O_{μ} is a local operators that commutes with the global symmetry U, μ can be considered a $\mathbb{Z}_2$-valued 0-boundary. Under this identification, we write $\mu = d\phi$ for a $\mathbb{Z}_2$-valued 1-chain ϕ, where d is the boundary operator of the chain complex defined by $\mathcal{G}$. The choice of ϕ is arbitrary up to adding edges in the support of B_p operators. We choose a ϕ with minimal support. The deformed check is obtained by replacing each term in the above expansion as follows:

$$O_{\mu} \mapsto \tilde{O}_{\mu} = O_{\mu} \prod_{e \in K} Z_e, \tag{C15}$$

where K is the support of ϕ. Hence,

$$\tilde{s}_j = \sum_{\mu} c_j^{\mu} \tilde{O}_{\mu}. \tag{C16}$$

Upon performing Z measurements on the edges of the graph $\mathcal{G}$ in step 4, the observed outcomes of these measurements form a $\mathbb{Z}_2$-valued 1-coboundary h_e, due to the B_p stabilizer checks of the deformed code. This leads to a correction operator

$$C = \prod_{v \in W} X_v, \tag{C17}$$

where W denotes the support of a $\mathbb{Z}_2$-valued 0-cochain w with coboundary satisfying $(\delta w)_e = h_e$.

Following the above procedure, gauging a Clifford symmetry of a Pauli stabilizer code can be used to perform logical non-Clifford gates by switching to an intermediate code, that is stabilized by a non-Abelian Clifford stabilizer group, and back. This approach was used to derive the procedure in Sec. III, which can be understood as gauging an anyon-permuting $\mathbb{Z}_2$ symmetry on two copies of the toric code. We explain this for the specific model in the next subsection. The gauging procedure also applies to the logical $\overline{XS}$ measurement (i.e., $\bar{T}$-state preparation) protocol; see Appendix D.

4. Gauging a $\mathbb{Z}_2$ symmetry in a microscopic model

In the following, we show how the Clifford stabilizer model introduced in Sec. III can be thought of as a model obtained from gauging the CZ symmetry (which acts as $\mathbb{Z}_2$ in the way described in Appendix C 2) in two copies of the toric code on triangular lattices. In doing so, we illustrate how an abstract gauging operation is implemented microscopically by enforcing (projecting or measuring with either postselection or an appropriate correction) local stabilizers whose product is the symmetry operator. In this method, the global symmetry is enforced via a combination of local symmetries.

We start with two toric codes on two triangular lattices, shown in red and blue in Fig. 26 that are displaced with respect to each other. Using the notation from Sec. III, we start with a code that is stabilized by $A_{v,c}$ and $\mathcal{B}_{f,c}$ for $c = r, b$. This code has a global $\mathbb{Z}_2$ symmetry generated by products of CZ unitaries acting between qubits on the red and the blue layers. We depict this symmetry operator and its action on the logical operators in Fig. 26.

Gauging is performed by adding extra degrees of freedom initialized in the Z basis, e.g., in $|0\rangle$. We choose to place them on the edges of a third triangular lattice, which we label in green, in line with Sec. III. The CZ symmetry is gauged by projecting onto the $+1$ eigenspace $\mathcal{A}_{v,g}$ operators as shown in Fig. 27, which are essentially the domain walls living on the green surrounding domains where the CZ symmetry of the red and blue toric codes has been applied locally. The fact that these operators are stabilizers of the gauged model can be understood as a *proliferation* of domain walls of the CZ symmetry—i.e., promoting this symmetry to a dynamical gauge field.

The product of all the local operators $\mathcal{A}_{v,g}$ reproduces the global symmetry operator CZ corresponding to the gauged symmetry. In Fig. 26, we see that the symmetry operator naturally decomposes into products over groups of six

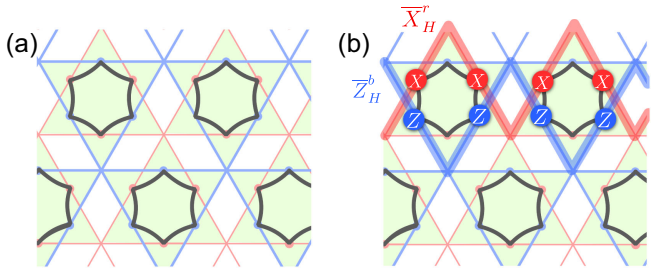


FIG. 26. (a) The $\mathbb{Z}_2$ symmetry of red and blue copies of the toric code. Each dark line corresponds to a CZ gate acting between the red and blue qubit at its end points. (b) shows the action of the corresponding realizations of the symmetry on $\bar{X}_H^r$, giving $\bar{X}_H^r \bar{Z}_H^b$.

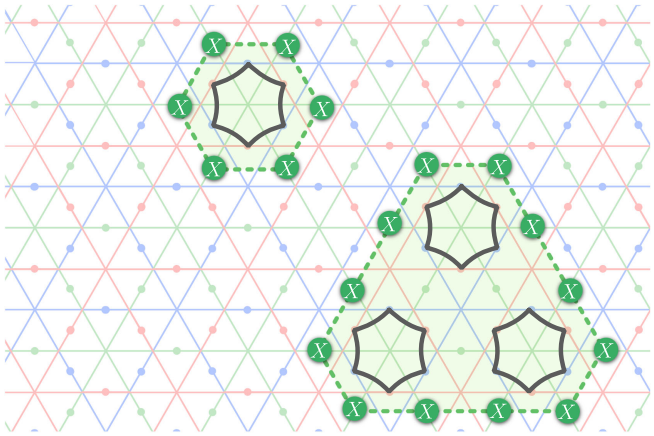


FIG. 27. An example of one of the operators that need to be added to the stabilizer group upon gauging two copies of the toric code into the twisted quantum double. These operators consist of the *CZ*-like symmetry shown in Fig. 26 applied only in certain domains, surrounded by a green domain wall (seen as a continuous green loop on the dual lattice).

qubits. In Fig. 27, we show the local stabilizers that have to be added to the gauged model. Note that they exactly coincide with the $\mathcal{A}_{v,g}$ Clifford stabilizers of the TQD model in Sec. III.

Viewed through this lens, we see that the protocol presented in Sec. III is precisely gauging the *CZ* symmetry where the gauging projection is implemented with quantum measurements and error correction or postselection. We also show how the symmetry can be "ungauged" using single-qubit Pauli measurements. This reverts the gauging process while preserving the eigenspace of the gauged symmetry operator. This way, the gauging and ungauging procedure projects onto a definite eigenstate of a logical Clifford operator.

As a side remark, this prescription performs the gauging of symmetry with non-on-site local action (albeit on-site after coarse-graining). For this kind of symmetry action, the usual prescription for gauging that involves the cluster state entangler [121] would be quite cumbersome. In addition, the usual prescription would prepare the twisted quantum double phase on an entirely new set of qubits, requiring a larger overhead. In contrast, in our prescription, the red and blue qubits are shared between the toric code and the twisted quantum double model.

APPENDIX D: ADDITIONAL MICROSCOPIC EXAMPLES FROM GAUGING

As we discussed in Sec. III E and Appendix C, logical protocols can be implemented via code deformations derived from gauging Clifford symmetries of the input codes. We now turn to several additional microscopic examples of logical protocols on codes with open boundary conditions that we derive using this method. We first show a planar preparation of a single *CZ* state, we then show a version of the *CCZ* gate, and finally we show a planar model with appropriate boundary conditions to prepare a *T* state.

1. *CZ*-state preparation with open boundary conditions

We first discuss a planar version of the *CZ*-magic state preparation scheme. Specifically, we describe how the boundaries are prepared to implement this gate. We also remark that a small implementation of such a *CZ*-state preparation scheme has been demonstrated experimentally in Ref. [55] (there, a slightly different microscopic circuit that is based on a folded color code was used).

We show the lattice geometry in Fig. 28. The figure shows three triangular lattices that each can support a surface code, with qubits on the lattice edges. Before gauging, the red and the blue copies of the surface code admit a finite-depth logical $\overline{CZ}$ gate, whose expression is

$$\overline{CZ} \cong \prod_{\mathbf{col}(v)=g} \mathcal{A}_{v,g}. \tag{D1}$$

We then prepare a gauged code as discussed in Sec. III, except now that the boundary stabilizers are present. For this, we similarly prepare all the green qubits in the $|0\rangle$ state

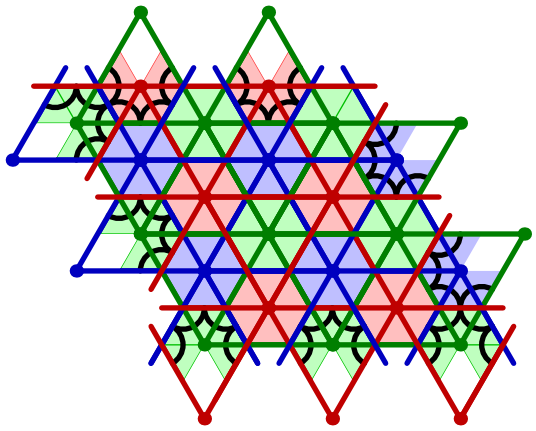


FIG. 28. A schematic depiction of the lattice geometry for implementing a logical $\overline{CZ}$ gauging measurement on two planar surface codes. We place a qubit on an edge of each color. The surface codes are initialized on red and blue triangular sublattices, which are equipped with appropriate boundary conditions. The stabilizers correspond to six-qubit Pauli-*X* vertex terms on red and blue edges and three-qubit Pauli-*Z* plaquette terms associated with each triangular plaquette of red or blue color. The green sublattice is used as an auxiliary block to implement the gauging. Upon gauging, we obtain a Clifford stabilizer model with three-qubit Pauli-*Z* plaquette stabilizers for each triangle and Clifford stabilizers associated with vertices of all three colors. In the bulk, this model is identical to the one shown in Fig. 3. At the boundary, the *CZ* parts of stabilizers are shown in black where they differ from the bulk ones.

and measure the green star operators $\mathcal{A}_{v,g}$, as described in Fig. 28. The product of these measurements returns the value of the parity operator $\hat{C}_g = \prod_v \mathcal{A}_{v,g}$. As such, the gauging procedure measures the logical $\overline{CZ}$ operator for the ungauged code. Measurements that project onto the even-charge-parity subspace therefore produce a CZ state, assuming we have initialized the red and blue surface code in the logical $|\overline{++}\rangle$ state. To initialize, we prepare all the physical qubits on the red and blue lattices in the $|+\rangle$ state and measure the plaquette operators to obtain the logical $|\overline{++}\rangle$ state and the red and blue surface codes. Importantly, before we perform the gauging operation, we have to repeat the plaquette measurements $O(d)$ times to correct the two lattices such that we can reliably recover the $B_p = +1$ subspace.

The ungauging procedure is carried out in the same way as described in Sec. III.

2. The *CCZ* gate

We now show a version of the protocol for the unitary CCZ gate on three copies of the surface code that we call "implementation by measurements." In Fig. 29, we show separate steps of this scheme: The green surface code is deformed over the red and blue surface codes and then transferred to the opposite side. In spacetime, this protocol looks like a version shown in Fig. 1, except that the spacetime regions are rotated and deformed such that the result matches. In the noise-free case, we can start with the configuration as shown in Fig. 29 and implement the scheme in just a few steps. In the first step, we prepare a TQD in the middle region with boundary conditions that glue it to an extended region of the green toric code on the left and right from it. Next, the green code is transported to the other side of the red and blue copy via the ungauging process, where the green qubits that do not maintain the surface code are measured in the Pauli-Z basis.

In Sec. II, we provided a physical interpretation of the CCZ gate where a green surface code is deformed slowly over a red and blue surface code via measurement. Here, we give an interpretation in terms of measurement-based operations. To show how this gate works, we observe that we can perform a controlled-controlled phase gate on three qubits via measurements together with a single additional qubit prepared in the zero state. We aim to perform a CCZ operation on the logical qubits indexed r, g, and b, with the additional qubit indexed a, prepared in state $|0\rangle_a$. We begin by measuring $M_1 = \bar{X}_g \overline{CZ}_{r,b} \bar{X}_a$ and apply a Pauli-Z correction, $\bar{Z}_a$, in the case of the -1 measurement outcome. Then, we measure the green qubit in the Pauli-Z basis to complete the CCZ operation while teleporting the green qubit onto the ancilla qubit. The logical circuit can be depicted via the circuit diagram which applies the logical CCZ gate up to a Clifford frame. We apply the Clifford frame $\overline{CZ}_{r,b} \bar{X}_a$ (not shown in the circuit) in the event when the outcome of the $\bar{Z}_g$ measurement was -1.

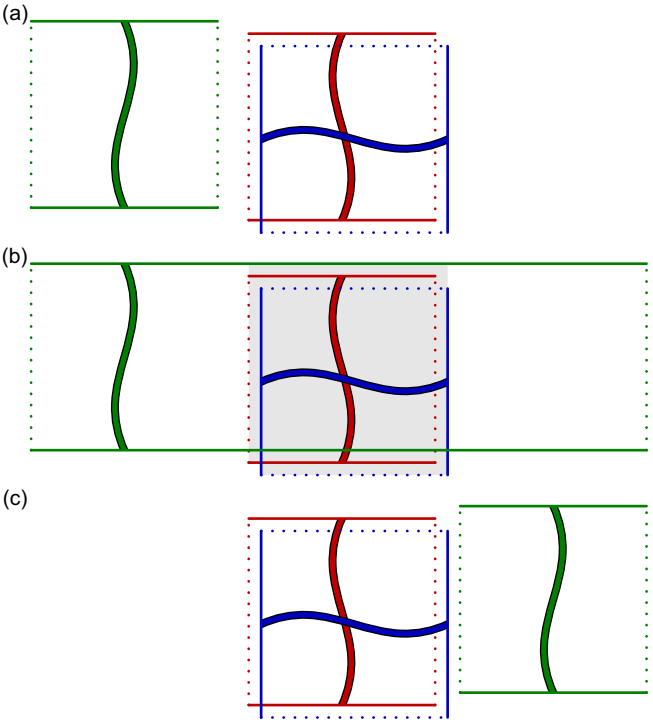
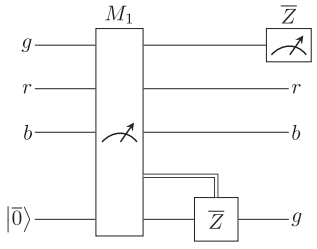


FIG. 29. A measurement-based implementation of the logical controlled-controlled phase gate. This accompanies the microscopic details shown in Fig. 30. Three qubits, colored red, blue, and green, are initialized on surface codes in (a). In step (b), the red and green copies in the middle are gauged, which is done simultaneously to joining them to the green surface code that has been prepared on the left and a second auxiliary green surface code to the right. This can be understood as deformation to a new code that consists of an extended region supporting the green surface code and TQD model in the middle. At the final step, we collapse the left side in the Pauli-Z basis and ungauge the middle region. This is done by measuring the green qubits in the Pauli-Z basis in the left and middle regions. This completes the CCZ gate with the green logical qubit teleported to the right-hand side, as shown in (c).



In Fig. 30, we show the microscopic details of how the CCZ gate is performed. At the bottom of the figure, we show how the system is initialized. The green code is shown to the left, and the red and blue codes are shown overlapping in the middle of the figure. An additional surface code region (which we labeled "a" above) shown

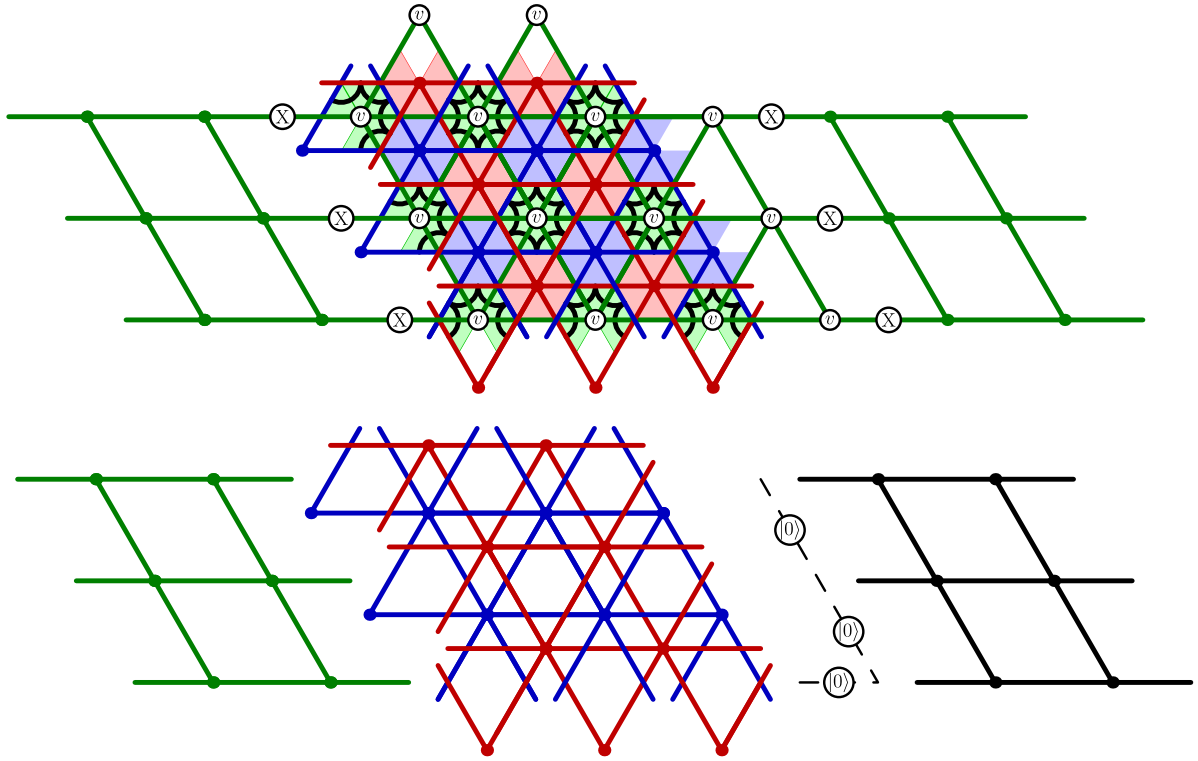


FIG. 30. The measurement-based CCZ gate by gauging. The system is prepared with one logical qubit on the green surface code to the left of the figure, and two logical qubits are prepared in the red and blue surface codes in the middle of the figure. The auxiliary (shown in black) surface code on the right is initialized in the logical zero state. The gauging operation merges the central twisted quantum double model with the two adjacent (green and black) surface codes. The product of the marked green vertex operators measure the operator $X_g CZ_{r,b} X_a$. As such, we measure the logical operator needed for the measurement-based CCZ gate, described in the main text, with its inclusion in the stabilizer group of the merged code.

in black is prepared to the right. The qubits of the additional black patch are initialized in the zero state. Next, we perform a gauging operation on the two central code patches, wherein we additionally couple the gauge degrees of freedom to the two adjacent surface codes, i.e., the green logical qubit and the additional logical qubit on the right. In doing so, we end up measuring the logical operator $\bar{X}_g \overline{CZ}_{r,b} \bar{X}_a$ of the initial state. Indeed, this operator is included in the stabilizer group of the new code. Specifically, it is given by the product of all the green vertex (Clifford) stabilizers marked in the top part of Fig. 30, which are represented by the star operator decoration around each of the green vertices. As a final step (not shown in Fig. 30, as it is simply a reflected version of the top panel), we measure the left code, and the green qubits of the central part in the Pauli-Z basis, to decouple the three surface code copies and to determine the correction we need to apply in the ungauging process. This measurement also teleports the green logical qubit onto the additional logical qubit to the right of the system and emulates the motion used in the proposal from Fig. 1. As a result, we obtain the CCZ unitary action on the r, g, and b qubits up to a Clifford frame $(I \otimes CZ \otimes X)^{m_Z}$, where m_Z is the result of logical Z measurement on the part of the green surface code that was performed during the ungauging procedure.

3. *T*-state preparation

Finally, we show how we can derive a preparation scheme for a T state, defined as $|T\rangle = T|+\rangle$, where $T = \omega^{|1\rangle\langle 1|}$ and $\omega = \exp(i\pi/4)$ from the gauging approach. The spacetime picture of such scheme is shown in Eq. (63); we then derive the microscopics from the gauging approach, starting form a color code model on a triangle defined on the lattice shown in Sec. VI C, and obtain the non-Abelian TQD by gauging. We first describe the resulting non-Abelian Clifford stabilizer model.

In Fig. 31, we depict the non-Abelian code on a triangle, including the boundary Clifford stabilizers. The lattice and stabilizers in the bulk are analogous to Sec. VI C, except that the plaquettes are now spatially deformed to form hexagons. The figure also shows how the Clifford stabilizers are defined at each of the boundaries. The boundaries chosen in this geometry are derived from the *color* boundaries of the color code, such that a boundary of color c supports no bricks of

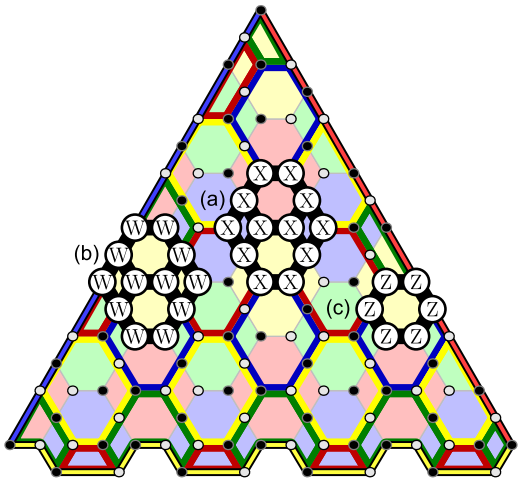


FIG. 31. Brickwork color code on a triangle. Qubits are shown in black and white; the stabilizers are associated with bricks and plaquettes. Decorated Clifford stabilizers are associated to the bricks—see, e.g., (a) and (b)—and Pauli-Z-type stabilizers are associated to the "faces" which are the intersections of top and bottom bricks (c). On the boundary the stabilizers are simply the truncations of the bulk stabilizers. The left and right boundaries are blue and red, respectively, and the bottom boundary is yellow.

color c. The left and right boundaries are colored blue and red, respectively, and the bottom boundary is yellow.

Following the argument in Sec. VI C, we can show that the gauged model presented in Fig. 31 is obtained by gauging a symmetry in a model that is equivalent to a 2D color code on a triangle. This color code model is obtained by condensing the green electric charges by measuring all of the green ZZ-edge terms of the lattice shown in Fig. 31. Specifically, given this color code model, the twisted quantum double is obtained by gauging a transversal $\overline{XS}$ gate of such a Pauli model. On the triangle, the color code encodes a single logical qubit on which the symmetry acts with a logical XS gate. Gauging this symmetry performs a logical measurement of the associated operator and, hence, can be used to prepare a T state; see Sec. IV D 4 and Appendix C 3. See also the appendix in Ref. [122] for a similar construction based on gauging the transversal H gate of the 2D color code.

Finally, ungauging this model is performed by measuring the green Z strings similarly to the explanation in Sec. VI. This teleports the logical state back into the ungauged color code.

APPENDIX E: TORIC CODE PHASE FROM THE PATH INTEGRAL

Here, we show how to relate the circuit for the square-lattice toric code to the path integral. The argument is not specific for a given lattice and can be extended to any spacetime cellulation. We provide only the basic structure and refer the reader to Ref. [77] for a more comprehensive description. The path integral for the toric code can be written as

$$Z = \sum_{\text{confs } \vec{c}} \prod_{\text{faces } f} \omega_f(\vec{c}) = \sum_{\vec{c}} \prod_f \delta\left(\sum_{\partial f} \vec{c} = 0 \bmod 2\right). \tag{E1}$$

That is, the weight of the path integral

$$w_f(\vec{c}) = \delta\left(\sum_{\partial f} \vec{c} = 0 \bmod 2\right)$$

enforces the constraint that the variables on the edges ∂f around the boundary of each plaquette f must have even total parity.

We now explicitly show that applying projectors $\frac{1}{2}(1+XXXX)$ for every vertex v and $\frac{1}{2}(1+ZZZZ)$ for every horizontal face f at each time step realizes the same action between input and output states as the expression for the path integral, upon appropriate identification between $\mathbb{Z}_2$ variables and qubits.

We consider a cubic lattice in spacetime and define the time direction to be along one of the axes of the cubic lattice. We call the edges that are parallel to the time direction timelike and the other edges spacelike. Note that the application of $\frac{1}{2}(1+XXXX)$ at each vertex of the spatial lattice corresponds to a sum over all possible configurations of operators where either 1 or the $XXXX$ operator is applied at each vertex. We associate the spacelike edges with data qubits for the toric code at corresponding time t and timelike edges with whether the 1 or the $XXXX$ operator is acting at a given vertex (to which the timelike edge is incident starting at time t). Here, the timelike edges remain "virtual" $\mathbb{Z}_2$ variables, while below, they are treated as ancilla qubits for vertex measurements.

To compute the global operator resulting from this circuit, we sum over all qubit configuration histories compatible with the actions of the projectors. Let $g_{t,e}$ denote the value of the qubit at the spacelike square lattice edge e before the time step t. Let the $h_{t,v}$ be the variable associated with the timelike edge incident to vertex v that tells us whether the 1 or $XXXX$ operator is applied to the edges adjacent to the square-lattice vertex v during time step t. We have

$$g_{t+1,e} = g_{t,e} + h_{v_0,t} + h_{v_1,t},$$

where v_0 and v_1 are the vertices adjacent to the edge e, since the qubit configuration at time $t+1$ is obtained from applying X to that at time t, according to the variables h at v_0 and v_1. The projector $\frac{1}{2}(1+ZZZZ)$ at a spacelike

plaquette f projects onto the space of configurations satisfying

$$g_{t,e_0} + g_{t,e_1} + g_{t,e_2} + g_{t,e_3} = 0, \tag{E2}$$

where e_0, e_1, e_2, and e_3 denote the edges of f. Hence, the overall operator implemented by the circuit can be written as a sum:

$$\sum_{\vec{g},\vec{h}} \left(\prod_{\substack{\text{times } t,\\ \text{edges } e}} (\delta_{g_{t+1,e}=g_{t,e}+h_{v_0,t}+h_{v_1,t}}) \prod_{\substack{\text{times } t,\\ \text{faces } f}} (\delta_{g_{t,e_0}+g_{t,e_1}+g_{t,e_2}+g_{t,e_3}=0}) \right). \tag{E3}$$

The second product in the brackets already looks identical to Eq. (E1) on the spacelike faces. The first product is, in fact, identical to Eq. (E1) on timelike faces. Thus, the expression that we obtain for the action of the circuit of projectors is identical to the toric code path integral from Eq. (E1) with our chosen identification of variables to qubits. Hence, the evaluation of the $+1$-postselected toric code circuit is the same as the evaluation of the toric code path integral on a cubic lattice.

When we replace the projectors with measurements, the -1 measurement outcome of an $XXXX$ operator at vertex v at time t corresponds to a factor of $(-1)^{h_{v,t}}$ in the path integral at the timelike edge associated with the variable $h_{v,t}$. The -1 measurement outcome of the $ZZZZ$-plaquette operator replaces the parity-even constraint with the parity-odd constraint on that plaquette in the path integral. These are associated with flux and charge worldlines, respectively, as we discuss in the main text.

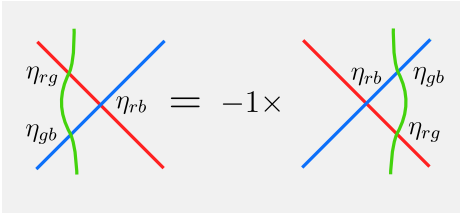
APPENDIX F: CORNERS AND BOUNDARIES OF THE TQD PHASE

In this appendix, we systematically derive the path-integral weights that occur at the boundaries and corners of the TQD path integral. A rigorous approach to this proceeds by defining microscopic path integrals on spacetime triangulations with boundaries, where membranes are represented as cellular cocycles, and the weights are associated to the highest-dimensional simplices, and then demanding either invariance under local changes in the triangulation (such as Pachner moves) or invariance under gauge transformations that locally change the cellular cocycles. This yields consistency conditions for the weights that are related to group cohomology that can be solved to determine the weights. For example, the weight associated to a bulk tetrahedron is a group 3-cocycle, and the weight associated to a boundary triangle is a group 2-cochain that trivializes the bulk 3-cocycle on a subgroup.

Here, we use the simpler continuum closed-membrane picture, where the weights are associated to intersections between lines or membranes. In this picture, deriving consistency conditions requires us to assume a convention for when membranes or loops are intersecting. Thus, this method is slightly less rigorous, but it brings us to the correct answer much faster.

We start by discussing the bulk weights associated with the configurations of red, green, and blue membranes. The only special points in the bulk are the triple intersections of the three different membrane types which we associate with a phase η. This weight must be a phase to define an path integral associated with a unitary action (i.e., quantum mechanics is unitary). Changing the orientation of a membrane configuration (i.e., reflecting it) should correspond to complex conjugating all the weights. Since the triple intersection is reflection invariant, we have $\eta = \bar{\eta}$, which has two solutions $\eta = 1$ and $\eta = -1$. $\eta = 1$ corresponds to three uncoupled copies of the toric code, whereas $\eta = -1$ is the TQD path integral that we consider here.

Next, we consider the boundaries. We start by arguing that there exists no topological boundary where membranes of all three colors are allowed to terminate. Such a boundary would have red, green, and blue termination lines at the boundary. We would have to associate some weight η_{rg} to every intersection of a red and green termination line inside the boundary and, similarly, weights η_{rb} and η_{gb} to red-blue and green-blue intersections. Demanding that the weights are invariant under local deformations means that we have the following equation:

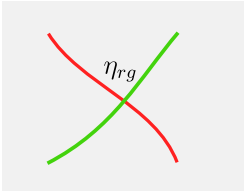


(F1)

$$\eta_{rg}\eta_{rb}\eta_{gb} = -\eta_{rg}\eta_{rb}\eta_{gb}.$$

Here and below, the light gray background represents the boundary in spacetime. The two sides show two different patterns of termination lines at the boundary. To match both sides, the attached bulk membranes (which are not shown) must include a triple intersection on one side, hence the -1 factor. The equation has no nonzero solution, and, thus, there is no topological boundary where all three membranes terminate. In the cellular cohomology derivation, the analogous argument is that terminating all membranes corresponds to having the full subgroup of the bulk $\mathbb{Z}_2^{\times 3}$ gauge group on the boundary, and the bulk weight is a cohomologically nontrivial group cocycle.

We next look at the $\langle r, g\rangle$ boundary where the red and green membranes can terminate. There is a weight η_{rg} associated to every intersection of a red and green termination line inside the boundary:



(F2)

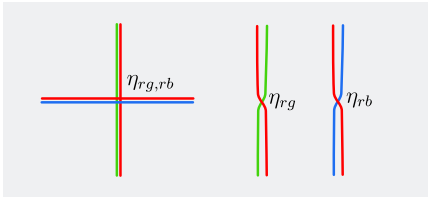
Such red-green intersections are invariant under reflection: In other words, there is no way to distinguish between a "left-handed" and a "right-handed" intersection. Thus, unitarity implies that $\bar{\eta}_{rg} = \eta_{rg}$. So we have two possible solutions, $\eta_{rg} = 1$ and $\eta_{rg} = -1$. These two solutions correspond to two different boundary conditions; however, they correspond to the same topological boundary phase. This is because one can be obtained from the other by adding a blue membrane along the boundary ("coating" the boundary with it), which is slightly displaced toward the bulk. Then, at every red-green boundary intersection, there is a bulk triple intersection between the attached red and green bulk membranes and the blue coating membrane. This way, the additional -1 factor in η_{rb} can be emulated by triple intersections with the help of blue coating membrane. Since the boundaries are equivalent, we pick the $\eta_{rg} = 1$ one.

Next, we consider the $\langle rg, rb\rangle$ boundary. There can now be both rg and rb termination lines on the boundary. However, it does not suffice to keep track of where red and green membranes terminate simultaneously; we also need to keep track of whether the red membrane attached to the termination line from the bulk is on the right or on the left from the green membrane attached to this line when looking along the rg (rb) termination line. To see this better, we can imagine that the red and green membranes do not terminate at exactly the same line but at two r and a g termination lines that are infinitesimally shifted away from each other. In order to depict different scenarios diagrams, we need to be able to swap the red and the slightly shifted green line, which causes the two attached membranes to cross at a line in the bulk:



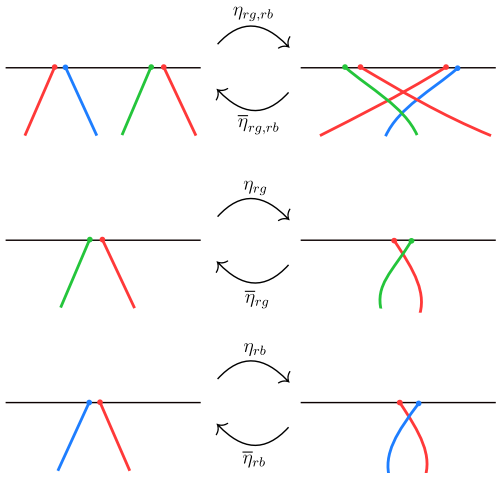
(F3)

For the $\langle rg, rb\rangle$ boundary, there are three different types of special points to which we can associate weights. These are the swapping points of rg, as well as rb termination lines, and intersections between rb and rg termination lines:



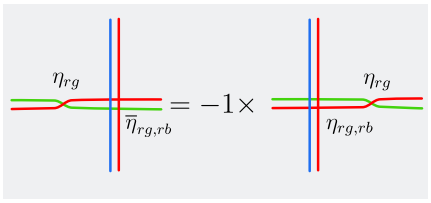
(F4)

In space, these weights correspond to the following processes of exchanging two pair termination points on the boundary (which we show by a solid black line in space):



(F5)

Note that, since rg and rb termination lines consist of infinitesimally separated lines, the above points are not reflection invariant and can be distinguished from their orientation-reversed versions. Because of unitarity, the orientation-reversed versions are associated with the complex conjugated weights. These weights have to fulfill the following consistency condition:



(F6)

which gives $\bar{\eta}_{rg,rb} = -\eta_{rg,rb}$. The global -1 sign occurs because the two sides differ by a triple intersection point between the attached green and blue membranes with the red membrane attached to the vertical red line. All other consistency conditions are trivial or equivalent to the one above. The solutions are given by $\eta_{rg,rb} = \pm i$, with η_{rg} and η_{rb} being arbitrary. Each of these solutions defines a different boundary condition, but they are all phase equivalent, so we choose the boundary with $\eta_{rg} = \eta_{rb} = 1$ and $\eta_{rg,rb} = i$. To see the equivalence, we first argue that η_{rg} can be changed arbitrarily by adding a phase of ϵ_{rg} to every

end point of every termination line in every diagram or $\overline{\epsilon_{rg}}$ if the ordering of red-green is clockwise. This changes the weight $\eta'_{rg} = \epsilon^2_{rg}\eta_{rg}$, but it does not affect any of the consistency conditions. In group cohomology language, the analogous statement is that the two different group 2-cochains are cohomologically equivalent and differ by the boundary of a 1-cochain ϵ. The same holds for η_{rb}. Finally, the $\eta_{rg,rb} = i$ and $\eta_{rg,rb} = -i$ boundaries are equivalent under adding a red membrane that "coats" the boundary, as this leads to an additional bulk triple intersection for every boundary $rg - rb$ intersection.

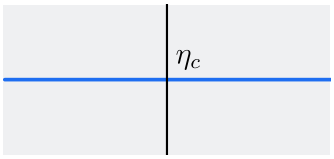
Finally, boundaries like $\langle\rangle$ or $\langle r\rangle$ where one or no membrane types are allowed to terminate do not have any nontrivial intersections, and, therefore, there are no nontrivial weights.

Next, we briefly consider the domain walls between the TQD and one or two toric codes. The domain walls that we use in our protocols simply couple the termination lines of the toric code membranes on one side to the TQD membrane termination lines on the other side of the domain wall. The weights are the same as for the according boundaries of the TQD.

Finally, we describe the corners that separate different boundary conditions of the TQD. Since there are 11 distinct boundaries, there are $11 \times 10/2 = 55$ different boundary pairs between which we can insert corners. For each pair of boundaries, there is a finite number of different superselection sectors of corners. There exists a standard method for classifying corners in topological fixed-point models by a dimensional reduction or "folding trick" to $(1+1)$D via compactifying the bulk into a thin slab with the two boundaries at the top and bottom [79,85–87]. We do not calculate the classification of corners here, but we do derive the path-integral weights for several of them from consistency conditions in the closed-membrane picture.

In our fault-tolerant protocols, we consider only corners that are "simple" and are fully specified by path-integral weights associated to the points where termination lines cross over the corner from one boundary to another.

As a first simple example, consider corners between the $\langle g, b\rangle$ boundary and $\langle r, b\rangle$ boundary. Only the blue termination lines can cross over the corner, and the crossing has an associated weight η_c:

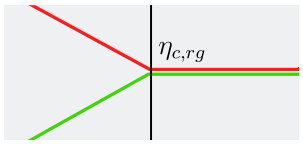


(F7)

Here and below, we depict a corner as a black line on a gray background (which denotes both neighboring boundaries). Since the crossing point is invariant under orientation reversal, we have $\eta_c = \bar{\eta}_c$. There are no further consistency conditions, so there are two solutions, $\eta_c = \pm 1$. These two solutions define two distinct corners which are, in fact, also in different superselection sectors. However, the $+1$ corner can be turned into a -1 corner by coating it with a blue charge anyon worldline; that is, we put the worldline on the corner and then slightly shift it into the bulk. The -1 factor we get from the blue termination line crossing the -1 corner is then emulated by the -1 factor we get when the coating anyon worldline crosses the attached blue membrane in the bulk. Which of the two corners we choose in the global topology of some fault-tolerant protocol changes the resulting logical operation. However, this change simply corresponds to a Pauli operator, usually a Pauli-Z operation acting on one of the logical qubits.

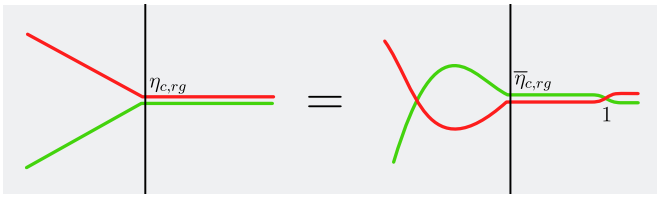
The corners between all $\langle g, b\rangle$, $\langle r, g\rangle$, and $\langle r, b\rangle$ in our protocols in Sec. IV D were all implicitly $+1$ corners. If, for example, we instead took the front right corner between the $\langle g, b\rangle$ and the $\langle r, g\rangle$ boundaries in Eq. (43) to be a -1 corner, as a result, this would swap the two measurement outcomes of the logical CZ measurement (and not apply a Pauli logical operator). Note that it is, in principle, possible to define a corner where the blue termination lines are not allowed to cross. However, this corner is not irreducible but rather is the direct sum of the $+$ and $-$ corners. If we used such a corner as a spatial corner in a protocol, it would carry a local logical qubit which is not topologically protected.

We next look at the corners between the $\langle rg, rb\rangle$ boundary and the $\langle r, g\rangle$ boundary. Note that, when crossing over the corner, termination lines are allowed to split. For example, a red-green termination line on one side may split into a separate red and green termination line on the other side. There is a weight $\eta_{c,rg}$ assigned to the points on the corner where an rg termination line on one side splits into separate r and g termination lines on the other side:



(F8)

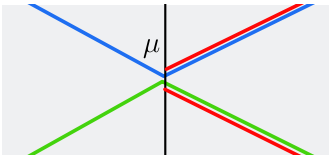
Because of the infinitesimal shift between the red and the green parts of the termination line, the point above is not orientation-reversal invariant. However, since crossing the red and green termination lines on the left does not produce any nontrivial weights, we get a consistency condition:



(F9)

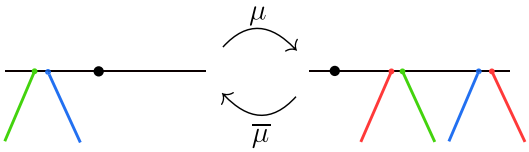
and the two possible solutions are again $\eta_{c,rg} = \pm 1$. The -1 boundary differs from the $+1$ boundary by coating with either a red or a green charge worldline. Again, we use the $+1$ corner in our protocols, and using the -1 corner instead would lead only to additional logical Pauli operators.

Finally, we consider the corners between the $\langle rg, rb\rangle$ boundary and the $\langle g, b\rangle$ boundary. There is a weight μ at every point on the corner where an rg and a rb termination line on one side can terminate at the same corner point as a g and a b termination line on the other side:



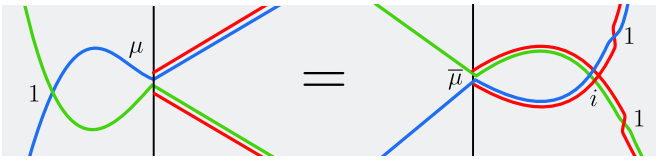
(F10)

In space, this weight corresponds to the process depicted below of moving termination points across the corner (in space, we schematically show the corner as a black dot on the boundary):



(F11)

The weight μ has to fulfill the following consistency condition:



$$\overline{\mu} = i\mu \, .$$

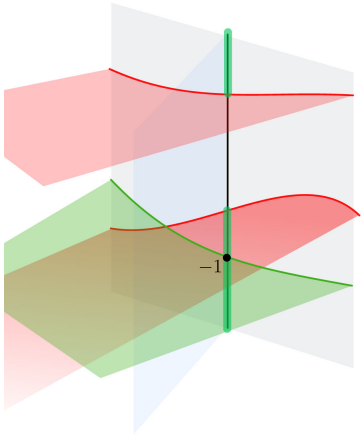
(F12)

There are two solutions to this equation, namely, $\mu = \pm\omega = \pm e^{2\pi i \frac{1}{8}}$. The $-\omega$ corner differs from the $+\omega$ corner by coating with either a green or a blue charge anyon worldline. So again, it does not matter which corner we choose, and we pick the $+\omega$ corner.

In addition to corners between boundaries, our protocols also involve corners that are the termination of a domain wall between the type-III TQD and one or two copies of the toric code. For these corners, we can ignore the attached toric code and the path-integral weights are the same as for the according corners between two TQD boundaries.

Finally, our protocols also involve zero-dimensional defects in spacetime, which are events where multiple corners meet. In general, such zero-dimensional defects are classified by vectors in a vector space. This vector space is the ground-state space of the model (i.e., the logical space of the code) on a two-dimensional topology, which is obtained by taking the intersection of the global topology with a small sphere that encloses the zero-dimensional point. For the protocol to be fault tolerant, this ground-state space must be one-dimensional. In fact, this is true not only for the zero-dimensional defects, but for any point on a corner, on a boundary, and in the bulk. So the choices of zero-dimensional defects differ only by a number. In a fault-tolerant protocol, this number is fixed by the fact that the circuits must be trace preserving, or, in other words, these numbers are only global prefactors which are irrelevant in quantum mechanics.

As an aside, note that there are also more interesting corners that do not just assign weights to crossing termination lines and which we do not use in our protocols. For example, consider the following "non-Abelian" corner between the $\langle r, g\rangle$ boundary and itself: (i) The corner is a fixed termination line for the blue membranes; (ii) we sum over all green string configurations inside the corner whose boundary points are given by the points where the red termination lines cross the corner (that is, for a connected corner segment, there are two possible opposite string configurations); (iii) for every green termination line crossing the green string on the corner, we get a factor of -1.



(F13)

APPENDIX G: NON-ABELIAN TQD MODEL FROM COPIES OF THE 3D TORIC CODE

Here, to complement the discussion in Sec. VI, we present a similar protocol for alternative microscopic realization of the TQD model from three-dimensional toric codes that support a transversal controlled-controlled-Z gate. We start with a cubic lattice with three qubits on each edge, indexed by $\{R, B, G\}$, with periodic boundary conditions. We denote the set of vertices with V and the set of faces with F. Moreover, we bicolor the cubic volumes such that no two cubes of the same color share a face. We label the set of cubes with the first color, white, with C_0 and the cubes with the second color, gray, with C_1. Their union, the set of all cubes, is labeled C.

The 3D code we start with is defined by a stabilizer group

$$S^{RBG} = \langle S^R, S^B, S^G \rangle, \tag{G1}$$

where S^R acts nontrivially only on qubits indexed by R, S^B on qubits indexed by B, and S^G on qubits indexed by G. Each of the three stabilizer groups define a three-dimensional toric code on the qubits A, B, and C individually.

The code on qubits R is defined via

$$S^R = \left\langle A^R_{c_0}, B^R_v | c_0 \in C_0, v \in V \right\rangle,$$
$$A^R_{c_0} = \prod_{e \in c_0} X^R_e, \qquad B^R_v = \prod_{e \in \delta v \cap C_1} Z_e, \tag{G2}$$

where, in slight abuse of notation, $e \in c$ denotes the set of edges surrounding the cube c. S^R has two types of generators. The X stabilizer generators are of weight 12 and act on the qubits surrounding a cube in C_0, and the Z stabilizer generators are of weight 3 and act on the qubits in the corner of cubes in C_1. Note that this code is the same as a three-dimensional surface code defined on a triangulation with degree 12 vertices. In Ref. [92], this code was introduced as the "rectified cubic code" and used in Ref. [23] to realize a logical CCZ gate on three 2D surface codes.

The stabilizer group S^B is defined in the same way as S^R but with the roles of the colors of the cubes reversed, i.e., $C_0 \leftrightarrow C_1$, meaning that in S^B the cubes in C_1 define the X stabilizers and the cubes in C_0 the Z stabilizers.

For the third subset of qubits, C, we pick a different microscopic realization of the toric code, defined by the stabilizer group

$$S^G = \langle A^G_v, B^G_f | v \in V, f \in F \rangle,$$
$$A^G_v = \prod_{e \in \delta v} X^G_e, \qquad B^G_f = \prod_{e \in \partial f} Z^G_e, \tag{G3}$$

where ∂f denotes the edges that contribute to the boundary of $f \in F$ and δv the edges that have v in their boundary. Note that this simply defines a conventional three-dimensional toric code on a cubic lattice.

We illustrate the stabilizer generators of S^{RBG} in Fig. 32. For the rest of this section, we denote the single-qubit Pauli P operator acting at the edge e of the A sublattice as P^R_e (and similarly for B and C).

S^{RBG} defined above has a transversal three-qubit controlled-controlled-Z gate, that acts on every triple of qubits assigned to the same edge via $CCZ|abc\rangle = (-1)^{abc}|abc\rangle$, where $a, b, c \in \{0, 1\}$ label the computational basis states on a system of three qubits. For the rest of this section, we denote the transversally applied CCZ gate with $\overline{CCZ} = \prod_e CCZ_e$. That $\overline{CCZ}$ is a logical gate of the 3D code defined above can be verified by conjugating the Pauli stabilizer group with the transversal gate, $\tilde{S}^{RBG} = \overline{CCZ} S^{RBG} \overline{CCZ}$. This maps the generators above to Clifford stabilizers where the X stabilizers acquire additional CZ terms. By direct calculation, we find that any such Clifford stabilizer $\tilde{A} \in \tilde{S}^{RBG}$ preserves the Pauli stabilizer group, i.e., $\tilde{A}^\dagger S^{RBG} \tilde{A} = S^{RBG}$, and, hence, is a Clifford stabilizer that acts like the logical identity on the code space. The code defined by S^{RBG} has a macroscopic distance proportional to the linear size of the system, and $\tilde{S}^{RBG}$ is locally generated. Any thin slice of a 3D model obtained by introducing boundaries perpendicular to one of the three spatial directions has the same type of Clifford stabilizers in the bulk (regardless of whether the full code has a logical non-Clifford gate); i.e., for any X stabilizer S_X that is unmodified when introducing the boundaries, there is an associated Clifford stabilizer obtained by conjugating S_X with a transversal CCZ gate.

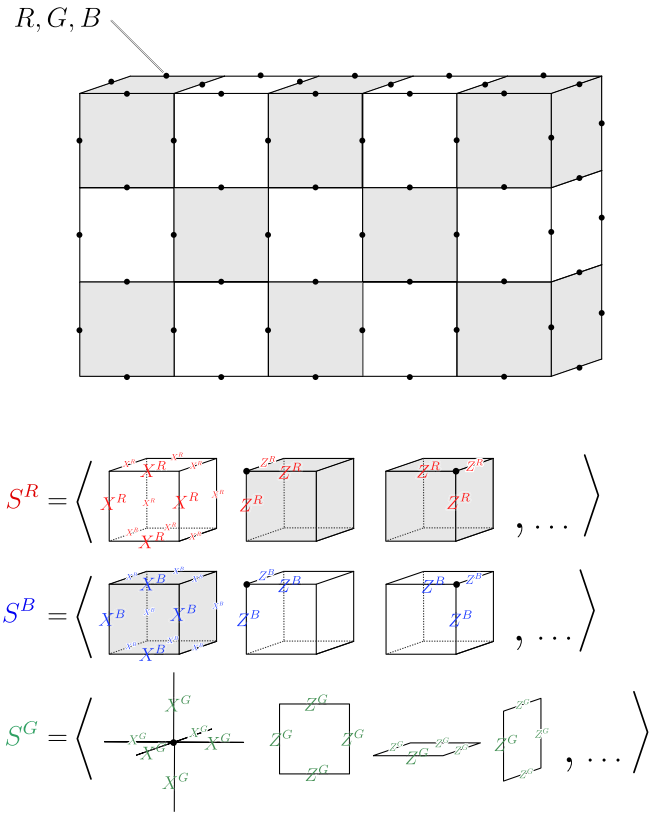


FIG. 32. We define three copies of toric codes on a square lattice on which we bicolor the volumes. On each edge, we place is a triple of qubits $R \times B \times G$ and define a stabilizer group $S^{RBG} = \langle S^R, S^B, S^G \rangle$, whose generators are shown on the right. The coloring is necessary only to define S^R and S^B. S^G defines a usual toric code associating X stabilizers to vertices and Z stabilizers to the faces of the cubic lattice.

1. The model in a thin 3D slab

We now consider the 3D model on a thin slab, namely, a manifold with the topology of $\mathbb{T}^2 \times [0,1]$ by introducing boundaries perpendicular to the 110 direction, as depicted in Fig. 33. Let Q be the qubits within the thin slab, which we assume to have thickness 1 (originating from a single layer of "rotated" cubes). The boundaries are introduced by measuring all qubits in the complement of the slab, Q^c in the single-qubit X basis. This creates two boundaries of the type labeled by $\langle r, g, b\rangle$ in Fig. 9. The stabilizer group of the resulting thin 3D model is generated by X-type stabilizers in the bulk, as well as truncated X-type stabilizers close to the boundary and all Z-type stabilizers that are fully supported in Q. Since we chose the 110 direction to be the "thin" direction, the cubes touch neighboring cubes along a face in one direction and along a single edge in the other direction; see Fig. 33.

The model also admits Clifford stabilizers inherited from the 3D code it was derived from. For each white cube c_0 in the thin slab, the code admits a Clifford stabilizer

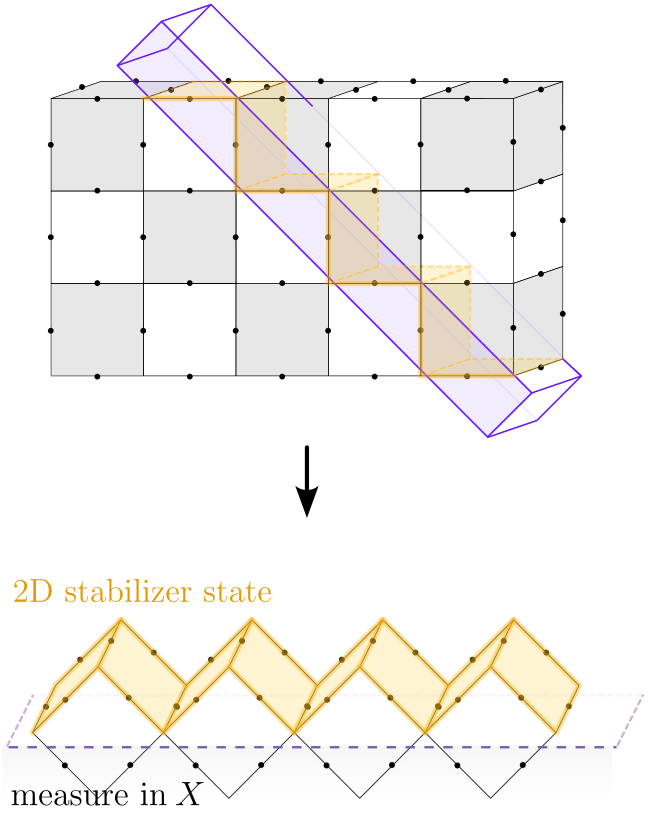


FIG. 33. We define a thin 3D model on the qubits in a thin slab that we choose perpendicular to the 110 direction of the cubic lattice, indicated in blue. Measuring all qubits outside of the thin slab in the X direction gives a code defined on a thin slab of qubits Q, shown on the right. The qubits on the orange edges are part of the top boundary of the thin model, which we label by Q_t. They form the qubits on which the non-Abelian code is supported after the initialization protocol.

$$\tilde{A}^R_{c_0} = \prod_{e \in c_0} X^R_e CZ^{BG}_e; \tag{G4}$$

for each gray cube c_1 in the thin slab, it admits a Clifford stabilizer

$$\tilde{A}^B_{c_1} = \prod_{e \in c_1} X^B_e CZ^{RG}_e; \tag{G5}$$

and for each six-valent vertex in the thin slab, the code admits the Clifford stabilizer

$$\tilde{A}^G_v = \prod_{e \in \delta v} X^G_e CZ^{RB}_e. \tag{G6}$$

We now consider two different examples of a 2D code obtained from this 3D code. In the first example, we simply perform single-qubit Pauli measurements on the qubits of the bulk and bottom boundary, which prepares a code state on the qubits of the top boundary equivalent to three independent 2D toric codes. In the second example, before measuring out the bulk and bottom qubits, we apply a transversal CCZ gate to the bottom and bulk qubits. This presents an alternative realization of the 2D code with non-Abelian topological order equivalent to the non-Abelian TQD discussed in the main text.

2. Initializing an Abelian 2D code from 3D toric codes

We start with a 3D toric code on a thin slab described in the previous section. We sort the qubits into two subsets: qubits that are part of the top boundary, Q_t, and qubits in the complement, Q^c_t. For the purposes of this section, we chose Q_t as indicated in Fig. 33. The qubits in Q_t live on the edges of a 2D square lattice. The bipartition of $Q = Q_t \sqcup Q^c_t$ is not canonical. There is some freedom in the assignment of qubits to the top boundary. For example, in the context of a repeating protocol that periodically switches between a thin 3D code and a 2D code on its boundary (as in Refs. [22,23]), a natural choice for Q_t would include only qubits that are measured out in the next period of switching from the 3D code to the 2D code. We expect that the protocol works for any choice of Q_t that is not cleanable in the 3D code.

Starting with the stabilizer group S^{RBG} of the thin model, we measure the qubits in Q^c_t in the single-qubit Pauli-X basis. The postmeasurement state is stabilized by $S^{RBG}_t = S^R_t \otimes S^B_t \otimes S^G_t$, which factorizes over the qubits $R \cap Q_t$, $B \cap Q_t$, and $G \cap Q_t$, respectively.

Recall that the lattice formed by the boundary qubits Q_t is a square lattice, consisting of horizontal and vertical edges. We find that S^R_t contains single-qubit X-type stabilizers on every other vertical edge. Since these qubits are in a product state, disentangled from the rest, we can remove these edges from the square lattice. This results in a

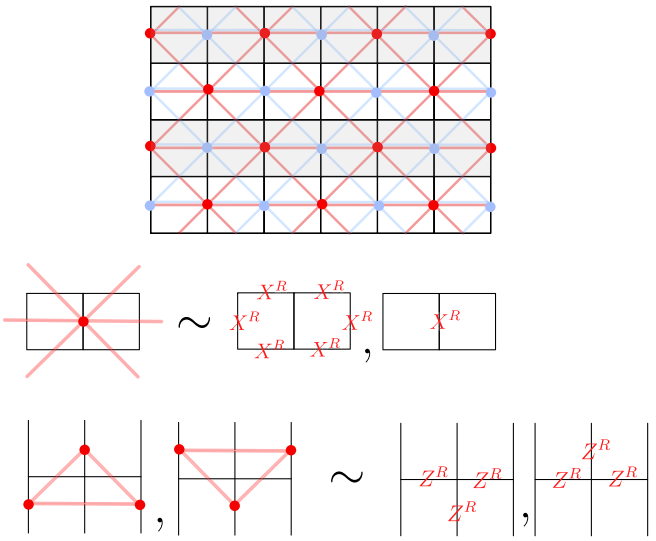


FIG. 34. Measuring all qubits except the top boundary qubits Q_t in the X basis yields a code defined on a square lattice. On both sets of qubits R and B, the resulting stabilizer group S^R and S^B is equivalent to toric codes defined on triangular lattices, since the stabilizer group contains some single-qubit Pauli operators. The lattices are shown in red and green, laid over the square lattice formed by Q_t. They are shifted horizontally with respect to each other, and we can place the qubits associated to the diagonal edges on top of each other. The code defined by $S^R \otimes S^B$ admits a nontrivial transversal logical gate $\overline{CZ}$ that acts with a CZ gate on all pairs of qubits on diagonal edges of the red (respectively, green) lattice. The G qubits are stabilized by a stabilizer group S^G of a toric code defined on the (black) square lattice with plaquette and vertex stabilizers of weight 4.

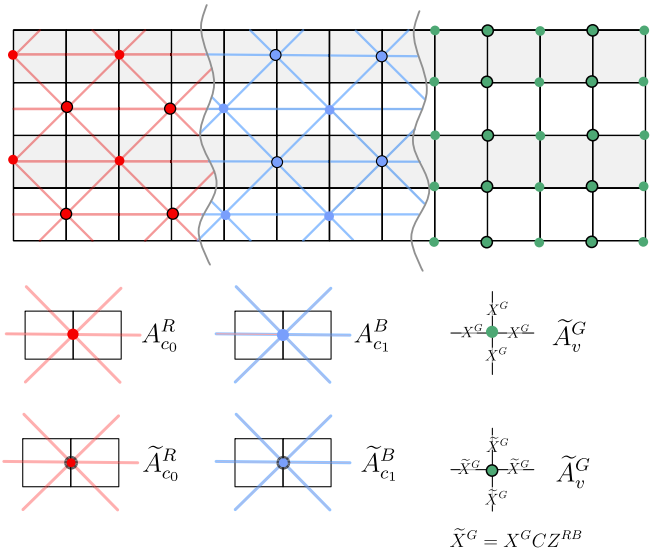


FIG. 35. The Clifford stabilizer generators of the non-Abelian code obtained using the transversal CCZ gate in three 3D toric codes. The model also has triangular plaquette Pauli-Z stabilizers for R and B qubits on respective sublattices and square plaquette Z stabilizers on G sublattice. This model can be viewed as a model obtained by gauging a $\mathbb{Z}_2$ symmetry in the two toric codes shown in Fig. 34. The product over all vertex stabilizers $\tilde{A}_v^G$ inherited from the stabilizers on the C qubits exactly reproduces the $\overline{CZ}$ symmetry of $S^R \otimes S^B$.

hexagonal lattice on which S_t^R reduces to the usual toric code stabilizer group on its dual, triangular lattice. We illustrate the effective toric code in Fig. 34. The effective toric code given by S_t^B is found analogously upon switching gray and white cubes and, thus, lives on a triangular lattice that is shifted by half a lattice site with respect to the lattice defining S_t^R.

Lastly, we find that after measuring Q_t^c in the X basis S_t^G is the stabilizer group of a 2D toric code on the square lattice formed by Q_t. Each (four-valent) vertex is associated with an X-type stabilizer A_v^G, and every (square) plaquette is associated with a Z-type stabilizer and B_f^G as defined in Eq. (G3).

Thus, we identified the 2D code on Q_t to be equivalent to three decoupled two-dimensional toric codes. This is useful when analyzing the non-Abelian model in the next subsection.

3. Initializing a non-Abelian 2D code from 3D toric codes

To initialize the non-Abelian code, we apply the $\overline{CCZ}$ gate to the qubits in Q_t^c before measuring them in the X basis; the resulting model has Clifford stabilizers. In Appendix G 4, we show that this code realizes the non-Abelian topological order corresponding to the type-III twisted TQD by identifying it as a code obtained from gauging an appropriate anyon-permuting $\mathbb{Z}_2$ symmetry in two 2D toric codes (which was discussed in Appendix C).

We start with a thin 3D code in the $+1$ eigenspace of all Pauli-Z stabilizers. This can be achieved either by post-selection or by error correction in the 3D code. In this case, the 3D model on the thin slab admits Clifford stabilizers $\tilde{A}_{c_0}^R$ and $\tilde{A}_{c_1}^B$ for every white cube c_0 and every gray cube c_1 that are fully contained in Q, as well as $\tilde{A}_v^G$ for every six-valent bulk vertex. Applying CCZ transversally to every triple of qubits on the same edge in Q_t^c maps these Clifford stabilizers to stabilizers that act with Pauli-X on the qubits in Q_c^t, namely,

$$\tilde{A}_{c_0}^R \overset{\overline{CCZ}|_{Q_t^c}}{\mapsto} \prod_{e \in c_0 \cap Q_t^c} X_e^R \prod_{e \in c_0 \cap Q_t} X_e^R CZ_e^{BG}, \tag{G7a}$$

$$\tilde{A}_{c_1}^B \overset{\overline{CCZ}|_{Q_t^c}}{\mapsto} \prod_{e \in c_1 \cap Q_t^c} X_e^B \prod_{e \in c_1 \cap Q_t} X_e^B CZ_e^{RG}, \tag{G7b}$$

$$\tilde{A}_v^G \overset{\overline{CCZ}|_{Q_t^c}}{\mapsto} \prod_{e \in \delta v \cap Q_t^c} X_e^G \prod_{e \in \delta v \cap Q_t} X_e^G CZ_e^{RB}. \tag{G7c}$$

In contrast, the X stabilizers that have overlap with Q_t^c are mapped to stabilizers that act with a Clifford unitary on these qubits.

Next, we measure the qubits in Q_t^c in the X basis. This preserves the stabilizers shown in Eq. (G7), since they commute with the measurements. By discarding the part supported on the qubits in Q_t^c (which can be done because these qubits are measured out in the X basis), we obtain Clifford stabilizers solely supported on Q_t. Together with the X- and Z-type stabilizers of the thin 3D slab that have already been supported only on Q_t, they generate the Clifford stabilizer group of the postmeasurement (2D) state. We show the stabilizer generators of the non-Abelian code in Fig. 35. In slight abuse of notation, we refer to the Clifford stabilizers that act only on Q_t as $\tilde{A}^R_{c_0}$, $\tilde{A}^B_{c_1}$, and $\tilde{A}^G_v$.

4. The phase of the non-Abelian code

In the following, we argue that the Clifford-stabilized code defined above can be obtained by gauging a $\mathbb{Z}_2$ symmetry on two toric codes that is realized by a product of on-site CZ gates; see Appendix C. This shows that the anyons of the code are equivalent to those in the quantum double of D_4 and the non-Abelian type-III twisted quantum double of $\mathbb{Z}_2^{\otimes 3}$ considered in this paper.

We start with the 2D Abelian model described in Appendix G 2. Consider the two decoupled codes defined by the stabilizer group $S_t^R \otimes S_t^B$. The stabilizer generators can be identified with vertex and plaquette terms of two toric codes defined on triangular lattices; see Fig. 34. We refer to these two triangular lattices as R and B sublattices. Since the lattices are shifted horizontally with respect to each other, the qubits on the diagonal edges of A and B can be identified one to one. Specifically, let D_A be the set of (qubits associated to) diagonal edges of the A lattice and, similarly for the D_B for the B lattice. For each edge $e \in D_A$, there exists a unique edge in D_B that crosses it. Hence, $e \in D_A$ can be considered as a label for an element in both D_A and D_B.

We define the global Clifford unitary

$$\overline{CZ} = \prod_{e \in D_A} CZ_e^{RB} \tag{G8}$$

that acts with a product of CZ gates on the pairs of qubits on the associated diagonal edges. We find that $\overline{CZ}$ and the X-type stabilizers commute up to Z stabilizers. This shows that $\overline{CZ}$ is a symmetry of the code $S_t^R \otimes S_t^B$. The action it has on the code space can be inferred from its commutation relation with the logical Pauli operators, which are the anyon string operators along nontrivial (dual) cycles of the R and B lattices.

Let $\bar{L}_m^R$ be an m-anyon string operator acting with products of Pauli-X operators along a nontrivial cycle of the dual A lattice. We find that

$$\overline{CZ}\bar{L}_m^R\overline{CZ} = \bar{L}_m^R\bar{L}_e^B, \tag{G9}$$

where $\bar{L}_e^B$ is an e string operator that acts with Pauli-Z operators along a cycle of the B lattice parallel to $\bar{L}_m^R$. By symmetry, the same holds for the string operators along inequivalent cycles and when A and B are interchanged. This shows that $\overline{CZ}$ acts as the anyon-permuting $\mathbb{Z}_2$ symmetry discussed in Appendix C.

Having identified an on-site symmetry in two triangular toric codes, we turn our attention back to the Clifford-stabilized code from Appendix G 3 and argue that it can be obtained from gauging this symmetry. Consider the vertex stabilizers of the non-Abelian code associated with the qubits of the G sublattice. We label them with $\tilde{A}_v^G$. If v is a bulk vertex of the thin 3D model, then $\tilde{A}_v^G$ is a Clifford stabilizer composed of a product of $X^G CZ^{RB}$. For v a boundary vertex of the thin 3D model, $\tilde{A}_v^G$ corresponds to a Pauli-X stabilizer. We find that

$$\prod_v \tilde{A}_v^G = \overline{CZ}. \tag{G10}$$

Hence, an eigenstate of all $\tilde{A}_v^G$ stabilizers individually is automatically an eigenstate of $\overline{CZ}$. We can initialize a given eigenstate of each $\tilde{A}_v^G$ by first fixing the $+1$ eigenstate of all Z stabilizers in S_t^{RBG}, by, e.g., measuring them and applying suitable X corrections (or via postselection) and then measuring each of the $\tilde{A}_v^G$ stabilizers. Given that the measurement was applied to the "flux-free" state in the $+1$ eigenspace of all Z-type stabilizers, we can apply Z-type Pauli corrections to resolve the -1 measurement outcomes. Thus, we obtain the non-Abelian model defined above by gauging the symmetry of the A and B toric codes. At the same time, as we know from Appendix C, this must produce the state with non-Abelian topological order whose anyons are described by the non-Abelian type-III twisted quantum double.

[1] B. M. Terhal, *Quantum error correction for quantum memories*, Rev. Mod. Phys. **87**, 307 (2015).

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

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

[4] M. A. Levin and X.-G. Wen, *String-net condensation: A physical mechanism for topological phases*, Phys. Rev. B **71**, 045110 (2005).

[5] C. Nayak, S. H. Simon, A. Stern, M. Freedman, and S. Das Sarma, *Non-Abelian anyons and topological quantum computation*, Rev. Mod. Phys. **80**, 1083 (2008).

[6] M. Barkeshli, P. Bonderson, M. Cheng, and Z. Wang, *Symmetry fractionalization, defects, and gauging of topological phases*, Phys. Rev. B **100**, 115147 (2019).

[7] Y. Hu, Y. Wan, and Y.-S. Wu, *Twisted quantum double model of topological phases in two dimensions*, Phys. Rev. B **87**, 125114 (2013).

[8] B. Yoshida, *Topological phases with generalized global symmetries*, Phys. Rev. B **93**, 155131 (2016).

[9] R. Dijkgraaf and E. Witten, *Topological gauge theories and group cohomology*, Commun. Math. Phys. **129**, 393 (1990).

[10] M. d. W. Propitius, *Topological interactions in broken gauge theories*, Ph.D. thesis, University of Amsterdam, 1995.

[11] 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).

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

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

[14] R. Raussendorf and J. Harrington, *Fault-tolerant quantum computation with high threshold in two dimensions*, Phys. Rev. Lett. **98**, 190504 (2007).

[15] H. Bombin and M. A. Martin-Delgado, *Quantum measurements and gates by code deformation*, J. Phys. A **42**, 095302 (2009).

[16] 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).

[17] H. Bombin and M. A. Martin-Delgado, *Topological quantum distillation*, Phys. Rev. Lett. **97**, 180501 (2006).

[18] B. Yoshida, *Gapped boundaries, group cohomology and fault-tolerant logical gates*, Ann. Phys. (Amsterdam) **377**, 387 (2017).

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

[20] A. C. Potter and T. Morimoto, *Dynamically enriched topological orders in driven two-dimensional systems*, Phys. Rev. B **95**, 155126 (2017).

[21] M. Barkeshli, Y.-A. Chen, S.-J. Huang, R. Kobayashi, N. Tantivasadakarn, and G. Zhu, *Codimension-2 defects and higher symmetries in* $(3+1)$D *topological phases*, SciPost Phys. **14**, 065 (2023).

[22] H. Bombin, *2D quantum computation with 3D topological codes*, arXiv:1810.09571.

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

[24] J. R. Wootton, J. Burri, S. Iblisdir, and D. Loss, *Error correction for non-Abelian topological quantum computation*, Phys. Rev. X **4**, 011051 (2014).

[25] A. Schotte, G. Zhu, L. Burgelman, and F. Verstraete, *Quantum error correction thresholds for the universal Fibonacci Turaev-Viro code*, Phys. Rev. X **12**, 021012 (2022).

[26] A. Schotte, L. Burgelman, and G. Zhu, *Fault-tolerant error correction for a universal non-Abelian topological quantum computer at finite temperature*, arXiv:2301.00054.

[27] G. Dauphinais and D. Poulin, *Fault-tolerant quantum error correction for non-Abelian anyons*, Commun. Math. Phys. **355**, 519 (2017).

[28] A. Hutter and J. R. Wootton, *Continuous error correction for Ising anyons*, Phys. Rev. A **93**, 042327 (2016).

[29] T. R. Scruby, D. E. Browne, P. Webster, and M. Vasmer, *Numerical implementation of just-in-time decoding in novel lattice slices through the three-dimensional surface code*, Quantum **6**, 721 (2022).

[30] T. R. Scruby, K. Nemoto, and Z. Cai, *Fault-tolerant quantum computation without distillation on a 2D device*, arXiv:2412.12529.

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

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

[33] A. M. Meier, B. Eastin, and E. Knill, *Magic-state distillation with the four-qubit code*, arXiv:1204.4221.

[34] E. T. Campbell, B. M. Terhal, and C. Vuillot, *Roads towards fault-tolerant universal quantum computation*, Nature (London) **549**, 172 (2017).

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

[36] F. Pastawski and B. Yoshida, *Fault-tolerant logical gates in quantum error-correcting codes*, Phys. Rev. A **91**, 012305 (2015).

[37] P. Webster, M. Vasmer, T. R. Scruby, and S. D. Bartlett, *Universal fault-tolerant quantum computing with stabilizer codes*, Phys. Rev. Res. **4**, 013092 (2022).

[38] M. H. Freedman, M. Larsen, and Z. Wang, *A modular functor which is universal for quantum computation*, Commun. Math. Phys. **227**, 605 (2002).

[39] K. Laubscher, D. Loss, and J. R. Wootton, *Universal quantum computation in the surface code using non-Abelian islands*, Phys. Rev. A **100**, 012338 (2019).

[40] N. E. Bonesteel and D. P. DiVincenzo, *Quantum circuits for measuring Levin-Wen operators*, Phys. Rev. B **86**, 165113 (2012).

[41] S. Bseiso, J. Pommerening, R. R. Allen, S. H. Simon, and L. Hormozi, *Minimal quantum circuits for simulating Fibonacci anyons*, arXiv:2407.21761.

[42] Z. K. Minev, K. Najafi, S. Majumder, J. Wang, A. Stern, E.-A. Kim, C.-M. Jian, and G. Zhu, *Realizing string-net condensation: Fibonacci anyon braiding for universal gates and sampling chromatic polynomials*, Nat. Commun. **16**, 6225 (2025).

[43] S. X. Cui and Z. Wang, *Universal quantum computation with metaplectic anyons*, J. Math. Phys. (N.Y.) **56**, 032202 (2015).

[44] I. Cong, M. Cheng, and Z. Wang, *Universal quantum computation with gapped boundaries*, Phys. Rev. Lett. **119**, 170504 (2017).

[45] C. Chamberland and A. W. Cross, *Fault-tolerant magic state preparation with flag qubits*, Quantum **3**, 143 (2019).

[46] C. Gidney, N. Shutty, and C. Jones, *Magic state cultivation: Growing t states as cheap as cnot gates*, arXiv:2409.17595.

[47] 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).

[48] D. J. Williamson and T. J. Yoder, *Low-overhead fault-tolerant quantum computation by gauging logical operators*, Nat. Phys. **22**, 598 (2026).

[49] A. W. Cross, Z. He, P. J. Rall, and T. J. Yoder, *Improved QLDPC surgery: Logical measurements and bridging codes*, arXiv:2407.18393.

[50] E. Swaroop, T. Jochym-O'Connor, and T. J. Yoder, *Universal adapters between quantum low-density parity check codes*, PRX Quantum **7**, 010324 (2026).

[51] B. Ide, M. G. Gowda, P. J. Nadkarni, and G. Dauphinais, *Fault-tolerant logical measurements via homological measurement*, Phys. Rev. X **15**, 021088 (2025).

[52] H. He, Y. Zheng, and C. von Keyserlingk, *Field theories for gauged symmetry-protected topological phases: Non-Abelian anyons with Abelian gauge group* $F_2^{\otimes 3}$, Phys. Rev. B **95**, 035131 (2017).

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

[54] E. Dennis, *Toward fault-tolerant quantum computation without concatenation*, Phys. Rev. A **63**, 052314 (2001).

[55] R. S. Gupta, N. Sundaresan, T. Alexander, C. J. Wood, S. T. Merkel, M. B. Healy, M. Hillenbrand, T. Jochym-O'Connor, J. R. Wootton, T. J. Yoder, A. W. Cross, M. Takita, and B. J. Brown, *Encoding a magic state with beyond break-even fidelity*, Nature (London) **625**, 259 (2024).

[56] M. Iqbal, N. Tantivasadakarn, R. Verresen, S. L. Campbell, J. M. Dreiling, C. Figgatt, J. P. Gaebler, J. Johansen, M. Mills, S. A. Moses, J. M. Pino, A. Ransford, M. Rowe, P. Siegfried, R. P. Stutz, M. Foss-Feig, A. Vishwanath, and H. Dreyer, *Non-Abelian topological order and anyons on a trapped-ion processor*, Nature (London) **626**, 505 (2024).

[57] X. Ni, O. Buerschaper, and M. Van den Nest, *A non-commuting stabilizer formalism*, J. Math. Phys. (N.Y.) **56**, 052201 (2015).

[58] J. C. Magdalena de la Fuente, N. Tarantino, and J. Eisert, *Non-pauli topological stabilizer codes from twisted quantum doubles*, Quantum **5**, 398 (2021).

[59] M. A. Webster, B. J. Brown, and S. D. Bartlett, *The XP stabiliser formalism: A generalisation of the Pauli stabiliser formalism with arbitrary phases*, Quantum **6**, 815 (2022).

[60] A. Lyons, C. F. B. Lo, N. Tantivasadakarn, A. Vishwanath, and R. Verresen, *Protocols for creating anyons and defects via gauging*, Phys. Rev. Lett. **135**, 200405 (2025).

[61] https://github.com/m-webster/XPFpackage/blob/main/CCZ2D/CCZ2D.ipynb.

[62] M. A. Webster, A. O. Quintavalle, and S. D. Bartlett, *Transversal diagonal logical operators for stabiliser codes*, New J. Phys. **25**, 103018 (2023).

[63] C. Vuillot, L. Lao, B. Criger, C. G. Almudéver, K. Bertels, and B. M. Terhal, *Code deformation and lattice surgery are gauge fixing*, New J. Phys. **21**, 033028 (2019).

[64] X. Ni, O. Buerschaper, and M. Van den Nest, *A non-commuting stabilizer formalism*, J. Math. Phys. (N.Y.) **56**, 052201 (2015).

[65] D. J. Williamson, Fractal symmetries: Ungauging the cubic code, Phys. Rev. B **94**, 155128 (2016).

[66] A. Kubica and B. Yoshida, *Ungauging quantum error-correcting codes*, arXiv:1805.01836.

[67] F. A. Bais and J. K. Slingerland, *Condensate-induced transitions between topologically ordered phases*, Phys. Rev. B **79**, 045316 (2009).

[68] D. J. Williamson and T. Devakul, *Type-II fractons from coupled spin chains and layers*, Phys. Rev. B **103**, 155140 (2021).

[69] N. Tantivasadakarn, R. Thorngren, A. Vishwanath, and R. Verresen, *Long-range entanglement from measuring symmetry-protected topological phases*, Phys. Rev. X **14**, 021040 (2024).

[70] N. P. Breuckmann and J. N. Eberhardt, *Quantum low-density parity-check codes*, PRX Quantum **2**, 040101 (2021).

[71] D. Aasen, D. Bulmash, A. Prem, K. Slagle, and D. J. Williamson, *Topological defect networks for fractons of all types*, Phys. Rev. Res. **2**, 043165 (2020).

[72] D. Williamson, *Spacetime topological defect networks and Floquet codes*, 10.26081/K69H2K (2023).

[73] V. G. Turaev and O. Y. Viro, *State sum invariants of 3-manifolds and quantum 6j-symbols*, Topology **31**, 865 (1992).

[74] R. Koenig, G. Kuperberg, and B. W. Reichardt, *Quantum computation with Turaev–Viro codes*, Ann. Phys. (Amsterdam) **325**, 2707 (2010).

[75] B. Sahinoglu, M. Walter, and D. Williamson, *A tensor network framework for topological order in higher dimensions*, in *APS March Meeting Abstracts* (American Physical Society, New York, 2016), Vol. 2016, pp. S20–002.

[76] A. Bauer, J. Eisert, and C. Wille, *Towards topological fixed-point models beyond gappable boundaries*, Phys. Rev. B **106**, 125143 (2022).

[77] A. Bauer, *Topological error correcting processes from fixed-point path integrals*, Quantum **8**, 1288 (2024).

[78] A. Bauer, *Low-overhead non-Clifford fault-tolerant circuits for all non-chiral Abelian topological phases*, Quantum **9**, 1673 (2025).

[79] A. Kitaev and L. Kong, *Models for gapped boundaries and domain walls*, Commun. Math. Phys. **313**, 351 (2012).

[80] L. Kong, *Anyon condensation and tensor categories*, Nucl. Phys. **B886**, 436 (2014).

[81] J. C. Magdalena de la Fuente, J. Eisert, and A. Bauer, *Bulk-to-boundary anyon fusion from microscopic models*, J. Math. Phys. (N.Y.) **64**, 111904 (2023).

[82] S. Beigi, P. W. Shor, and D. Whalen, *The quantum double model with boundary: Condensations and symmetries*, Commun. Math. Phys. **306**, 663 (2011).

[83] A. Bullivant, Y. Hu, and Y. Wan, *Twisted quantum double model of topological orders with boundaries*, Phys. Rev. B **96**, 165138 (2017).

[84] T. Lan, J. C. Wang, and X.-G. Wen, *Gapped domain walls, gapped boundaries, and topological degeneracy*, Phys. Rev. Lett. **114**, 076402 (2015).

[85] J. C. Bridgeman, D. Barter, and C. Jones, *Fusing binary interface defects in topological phases: The* $\mathrm{Vec}(\mathbb{Z}/p\mathbb{Z})$ *case*, J. Math. Phys. (N.Y.) **60**, 121701 (2019).

[86] J. C. Bridgeman and D. Barter, *Computing defects associated to bounded domain wall structures: The* $\mathrm{Vec}(\mathbb{Z}/p\mathbb{Z})$ *Case*, J. Phys. A **53**, 235206 (2020).

[87] J. C. Bridgeman and D. Barter, *Computing data for Levin-Wen with defects*, Quantum **4**, 277 (2020).

[88] A. Kubica, B. Yoshida, and F. Pastawski, *Unfolding the color code*, New J. Phys. **17**, 083026 (2015).

[89] A. Bauer, *The* $\mathrm{x}+\mathrm{y}$ *Floquet code: A simple example for topological quantum computation in the path integral approach*, Phys. Rev. A **111**, 032413 (2025).

[90] Y.-A. Chen and S. Tata, *Higher cup products on hypercubic lattices: Application to lattice models of topological phases*, J. Math. Phys. (N.Y.) **64**, 091902 (2023).

[91] N. E. Steenrod, *Products of cocycles and extensions of mappings*, Ann. Math. **48**, 290 (1947).

[92] M. Vasmer and D. E. Browne, *Three-dimensional surface codes: Transversal gates and fault-tolerant architectures*, Phys. Rev. A **100**, 012312 (2019).

[93] A. Bauer and J. C. Magdalena de la Fuente, *Planar fault-tolerant circuits for non-Clifford gates on the 2D color code*, Phys. Rev. A, 10.1103/p7c9-x1m9.

[94] R. Raussendorf, S. Bravyi, and J. Harrington, *Long-range quantum entanglement in noisy cluster states*, Phys. Rev. A **71**, 062313 (2005).

[95] H. Bombin, *Dimensional jump in quantum error correction*, New J. Phys. **18**, 043038 (2016).

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

[97] H. Bombin and M. A. Martin-Delgado, *Topological computation without braiding*, Phys. Rev. Lett. **98**, 160502 (2007).

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

[99] M. S. Kesselring, J. C. Magdalena de la Fuente, F. Thomsen, J. Eisert, S. D. Bartlett, and B. J. Brown, *Anyon condensation and the color code*, PRX Quantum **5**, 010342 (2024).

[100] H. Bombin, *Transversal gates and error propagation in 3D topological codes*, arXiv:1810.09575.

[101] G. Zhu, S. Sikander, E. Portnoy, A. W. Cross, and B. J. Brown, *Non-Clifford and parallelizable fault-tolerant logical gates on constant and almost-constant rate homological quantum LDPC codes via higher symmetries*, PRX Quantum **6**, 040361 (2025).

[102] B. Yoshida, *Topological color code and symmetry-protected topological phases*, Phys. Rev. B **91**, 245131 (2015).

[103] T. R. Scruby, M. Vasmer, and D. E. Browne, *Non-Pauli errors in the three-dimensional surface code*, Phys. Rev. Res. **4**, 043052 (2022).

[104] M. B. Hastings and J. Haah, *Dynamically generated logical qubits*, Quantum **5**, 564 (2021).

[105] M. Davydova, N. Tantivasadakarn, S. Balasubramanian, and D. Aasen, *Quantum computation from dynamic automorphism codes*, Quantum **8**, 1448 (2024).

[106] S. Balasubramanian, M. Davydova, and E. Lake, *A local automaton for the 2D toric code*, arXiv:2412.19803.

[107] S. Bravyi and J. Haah, *Quantum self-correction in the 3D cubic code model*, Phys. Rev. Lett. **111**, 200501 (2013).

[108] P. Gács, *Reliable computation with cellular automata*, in *Proceedings of the Fifteenth Annual ACM Symposium on Theory of Computing* (Association for Computing Machinery, New York, 1983), pp. 32–41.

[109] P. Gács, *Reliable cellular automata with self-organization*, J. Stat. Phys. **103**, 45 (2001).

[110] I. Çapuni and P. Gács, *A reliable turing machine*, arXiv:2112.02152.

[111] S. Bravyi, I. Kim, A. Kliesch, and R. Koenig, *Adaptive constant-depth circuits for manipulating non-Abelian anyons*, arXiv:2205.01933.

[112] N. Tantivasadakarn, A. Vishwanath, and R. Verresen, *Hierarchy of topological order from finite-depth unitaries, measurement and feedforward*, PRX Quantum **4**, 020339 (2023).

[113] Y. Ren, N. Tantivasadakarn, and D. J. Williamson, *Efficient preparation of solvable anyons with adaptive quantum circuits*, Phys. Rev. X **15**, 031060 (2025).

[114] S.-J. Huang and Y. Chen, *Generating logical magic states with the aid of non-Abelian topological order*, arXiv:2502.00998.

[115] R. Dijkgraaf, V. Pasquier, and P. Roche, *Quasi Hopf algebras, group cohomology and orbifold models*, Nucl. Phys. B, Proc. Suppl. **18**, 60 (1991).

[116] H. Bombin, G. Duclos-Cianci, and D. Poulin, *Universal topological phase of two-dimensional stabilizer codes*, New J. Phys. **14**, 073048 (2012).

[117] J. Haah, *Classification of translation invariant topological Pauli stabilizer codes for prime dimensional qudits on two-dimensional lattices*, J. Math. Phys. (N.Y.) **62**, 012201 (2021).

[118] T. D. Ellison, Y.-A. Chen, A. Dua, W. Shirley, N. Tantivasadakarn, and D. J. Williamson, *Pauli stabilizer models of twisted quantum doubles*, PRX Quantum **3**, 010353 (2022).

[119] N. Tarantino, N. H. Lindner, and L. Fidkowski, *Symmetry fractionalization and twist defects*, New J. Phys. **18**, 035006 (2016).

[120] J. C. Teo, T. L. Hughes, and E. Fradkin, *Theory of twist liquids: Gauging an anyonic symmetry*, Ann. Phys. (Amsterdam) **360**, 349 (2015).

[121] R. Verresen, N. Tantivasadakarn, and A. Vishwanath, *Efficiently preparing Schrödinger's cat, fractons and non-Abelian topological order in quantum devices*, arXiv:2112.03061.

[122] A. Prem and D. J. Williamson, *Gauging permutation symmetries as a route to non-Abelian fractons*, SciPost Phys. **7**, 068 (2019).