

# Quantum Computing with Majorana Fermion Codes

Daniel Litinski and Felix von Oppen

*Dahlem Center for Complex Quantum Systems and Fachbereich Physik,  
Freie Universität Berlin, Arnimallee 14, 14195 Berlin, Germany*

We establish a unified framework for Majorana-based fault-tolerant quantum computation with Majorana surface codes and Majorana color codes. All logical Clifford gates are implemented with zero time overhead. This is done by introducing a protocol for Pauli product measurements with tetrons and hexons which only requires local 4-Majorana parity measurements. An analogous protocol is used in the fault-tolerant setting, where tetrons and hexons are replaced by Majorana surface code patches, and parity measurements are replaced by lattice surgery, still only requiring local few-Majorana parity measurements. To this end, we discuss twist defects in Majorana fermion surface codes and adapt the technique of twist-based lattice surgery to fermionic codes. Moreover, we propose a new family of codes that we refer to as Majorana color codes, which are obtained by concatenating Majorana surface codes with small Majorana fermion codes. Majorana surface and color codes can be used to decrease the space overhead and stabilizer weight compared to their bosonic counterparts.

## I. INTRODUCTION

In an effort to construct robust qubits for quantum computation, Majorana zero modes (Majoranas) in topological superconductors [1–3] are currently being explored as potential building blocks of topological qubits. Even though no such topological qubit has been built to date, a range of precursor experiments exists [4–7], providing various signatures of Majorana zero modes. While it remains uncertain whether Majorana-based qubits will offer higher coherence times compared to other solid-state qubits, Majorana-based qubits have one distinct feature that sets them apart from non-topological qubits. Due to the nonabelian statistics of Majoranas, these qubits can be measured in all three Pauli bases  $X$ ,  $Y$ , and  $Z$ , and not just in a single computational basis as is the case with conventional qubits. As a consequence, Majorana-based qubits implement robust single-qubit Clifford gates with zero time overhead. These gates are products of Hadamard gates  $H$  and phase gates  $S$ , and are equivalent to the operations implemented by braiding Majoranas.

The coherence times of Majorana-based qubits are predicted to be long, but still limited. Thus, a quantum error-correcting code is necessary for large-scale quantum computation with arbitrarily long qubit survival times. Quantum error correction combines many physical qubits into more error-resilient logical qubits [8]. Errors are detected and corrected by measuring certain stabilizer operators. Two-dimensional (2D) topological codes are of particular interest, since their local stabilizers are compatible with the constraints of solid-state architectures.

Topological codes come as bosonic or fermionic codes. Bosonic codes are defined on a 2D lattice where vertices correspond to qubits and faces define stabilizers, products of Pauli operators with support on all qubits of the face. Fermionic codes are defined on a lattice where vertices correspond to Majoranas and stabilizers are products of all Majorana operators of a face. A bosonic code maps onto a fermionic code by replacing each qubit with four Majoranas in a fixed parity sector [9], but not all fermionic codes can be straightforwardly mapped back onto a bosonic code. This implies that Majorana-based qubits admit a wider range of topo-

logical codes than conventional qubits.

One practical problem is that fault-tolerant quantum computing requires logical operations on encoded qubits, which may be entirely different from operations on physical qubits. Indeed, various recent proposals for Majorana-based implementations of codes exist, based on, e.g., bosonic surface codes [10–13], bosonic color codes [14, 15], fermionic surface codes [9, 16, 17], and small Majorana fermion codes [18, 19], each featuring individual protocols for a universal set of logical gates. Moreover, except for twist-based encodings in surface codes [20–22] and transversal gates of bosonic color codes [14, 15, 23], logical single-qubit Clifford gates require a series of code operations in all aforementioned proposals, and the Majoranas' advantage of zero-overhead single-qubit Clifford gates is lost.

According to the Gottesman-Knill theorem [24], not only single-qubit Clifford gates, but all Clifford gates including the two-qubit controlled-NOT (CNOT) gate can be tracked using a classical computer. Therefore, all Clifford gates can, in principle, be implemented with zero time overhead. Operations on logical qubits need to be appropriately designed in order to take full advantage of the Gottesman-Knill theorem.

**Overview and main results.** In this work, we establish a unified framework for Majorana-based quantum computation with bosonic and fermionic surface and color codes, in all of which logical Clifford gates, including CNOT gates, are implemented with zero time overhead. To this end, we first discuss the construction of Majorana surface code patches in Sec. II. These surface code patches are essentially fault-tolerant versions of tetrons and hexons which can correct a certain number of errors. While tetrons and hexons were introduced in Refs. [27, 28] as quantum-wire-based [29, 30] constructions, we use these terms for any physical system with four or six Majoranas in a fixed parity sector. Next, in Sec. III, we describe a protocol to implement minimal-overhead Clifford gates with *physical* tetrons and hexons by measuring arbitrarily nonlocal Pauli product operators using only local operations. This is not fault-tolerant in the sense that the protocol assumes perfect measurements and error-free qubits. However, it is entirely analogous to the fault-tolerant protocol discussed in Sec. IV, where we extend the protocol for minimal-overhead Clifford gates to

Order	Code family (with minimal-overhead Clifford gates)	Majorana overhead for code distance $d$	Building blocks	Maximum stabilizer weight	Bulk weights	Max. stabilizer wt. for lattice surgery
surface codes (non-overlapping 2D-local topological codes)						
1	4.6.12 Majorana surface code	$12d^2 + \mathcal{O}(d)$	dodecons	6	4, 6	6
	bosonic subsystem surface code <sup>1</sup>	$12d^2 + \mathcal{O}(d)$	qubits*	6	6	10
	6.6.6 Majorana surface code <sup>2</sup>	$6d^2 + \mathcal{O}(d)$	hexons / octons	6	6	6
	4.8.8 Majorana surface code <sup>3</sup>	$4d^2 + \mathcal{O}(d)$	tetrans / hexons	8	8	8
	bosonic surface code <sup>4</sup>	$4d^2 + \mathcal{O}(d)$	qubits*	8	8	10
color codes (multiple layers of surface codes obtained by concatenation)						
2	4.8.8 ( $[[[6, 2, 2]]_m$ ) Majorana color code	$3d^2 + \mathcal{O}(d)$	hexons	8	8	8
	bosonic 6.6.6 color code <sup>5</sup>	$3d^2 + \mathcal{O}(d)$	qubits*	12	12	12
3	6.6.6 ( $[[[20, 4, 4]]_m^6$ ) Majorana color code	$2.5d^2 + \mathcal{O}(d)$	decons	12	10, 12	12
	bosonic 4.8.8 color code <sup>5</sup>	$2d^2 + \mathcal{O}(d)$	qubits*	16	8, 16	12
4	4.8.8 ( $[[[8, 3, 2]]_m$ ) Majorana color code	$\frac{8}{3}d^2 + \mathcal{O}(d)$	octons	10	8, 10	10
	4.8.8 ( $[[[16, 3, 4]]_m^6$ ) Majorana color code	$\frac{4}{3}d^2 + \mathcal{O}(d)$	octons	16	8, 16	12
k	4.8.8 Majorana surface code concatenated with $[[n_m, k, d_m]]_m$	$\frac{4n_m}{kd_m^2}d^2 + \mathcal{O}(d)$		at least $4d_m$		
	bosonic surface code concatenated with $[[n, k, d_b]]$	$\frac{4n}{kd_b^2}d^2 + \mathcal{O}(d)$	qubits*	at least $8d_b$		

first mentioned in: 1) [25], 2) [16], 3) [11], 4) [26], 5) [23], 6) [19] (albeit without minimal-overhead Clifford gates)

\*for Majorana-based hardware: tetrans

TABLE I. Comparison of different two-dimensional topological codes with respect to the following characteristics: number of Majoranas necessary to encode a logical qubit with code distance  $d$ , proposed number of Majoranas in a fixed parity sector used as a building block (tetrans/hexons/octons/decons/dodecons refer to 4/6/8/10/12 Majoranas in a fixed parity sector), maximum Majorana weight of the stabilizers that need to be measured, stabilizer weights in the bulk of the code (excluding the boundaries), and the maximum stabilizer weight for lattice surgery operations. For Majorana color codes, the  $[[n_m, k, d_m]]_m$  code in parentheses is the code that is concatenated with the corresponding surface code in order to obtain the color code. The order of each code indicates how many layers of Majorana surface codes it corresponds to. All codes listed in the table implement logical Clifford gates with zero time overhead.

logical tetrans and hexons, i.e., surface code patches. This is done by describing twist defects in Majorana surface codes and adapting twist-based lattice surgery [22] to fermionic codes. Finally, in Sec. V, we propose a construction procedure for Majorana color codes, i.e., the fermionic equivalent of bosonic color codes. Since bosonic color codes can be obtained by concatenating bosonic surface codes with small non-topological codes [31], we describe code concatenation for fermionic codes and construct Majorana color codes by concatenating Majorana surface codes with small Majorana fermion codes.

With fermionic and bosonic versions of surface and color codes, a plethora of topological codes is available, all of which can implement logical Clifford gates with zero time overhead. To decide which code to use for Majorana-based quantum computation, we show a comparison of 2D topological codes in Tab. I. The main differences between these codes lie in their Majorana overhead, i.e., the number of Majoranas required to encode a qubit with code distance  $d$ , and in their stabilizer weight, i.e., the number of Majorana operators contained in the stabilizers that need to be measured for error correction. It is desirable to keep both of these figures low, as a lower Majorana overhead implies a higher encoding rate and more

efficient space usage, and a lower stabilizer weight implies easier implementation due to lower hardware requirements. However, the general trend seen in bosonic surface and color codes indicates that codes featuring lower-weight stabilizers tend to have a higher space overhead (or Majorana overhead in the case of Majorana-based qubits).

Out of all known bosonic 2D topological codes, surface codes have the lowest stabilizer weight of four qubits, with subsystem surface codes [25] reducing the weight to three qubits. With Majorana-based hardware, this corresponds to 8-Majorana and 6-Majorana stabilizers. In order to implement minimal-overhead Clifford gates, twist-based lattice surgery requires the measurement of higher-weight 5-qubit operators corresponding to twist defects, which corresponds to 10-Majorana operators. Majorana surface codes decrease the stabilizer weight below the weight of bosonic surface codes, such that only 4- and 6-Majorana operators need to be measured. This might be useful if higher-weight stabilizers turn out to be difficult to measure in a given implementation. Majorana color codes, on the other hand, can encode quantum information with a lower space overhead compared to bosonic color codes with the same stabilizer weight.

In general, bosonic surface codes can be concatenated with

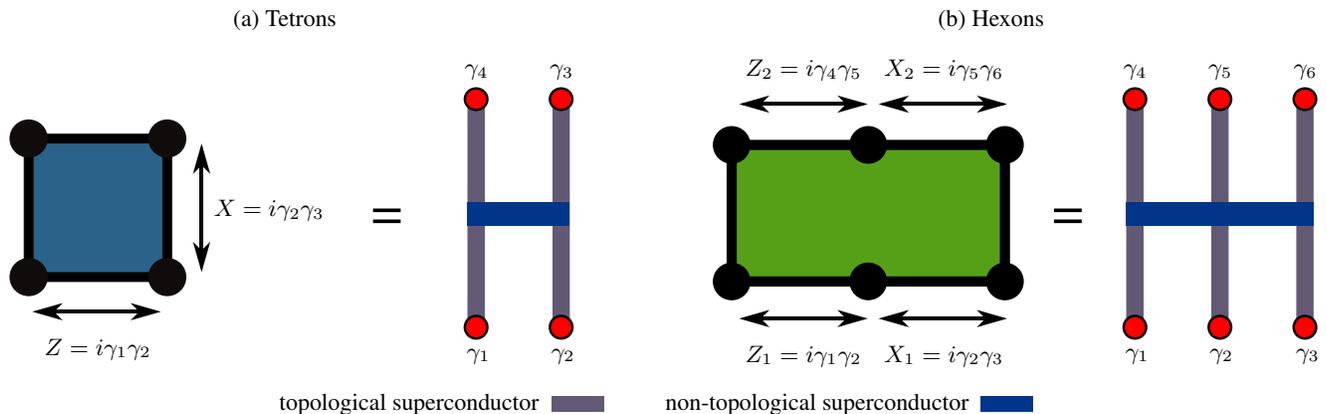


FIG. 1. Tetrons and hexons as the two smallest Majorana fermion codes, and their correspondence to four or six Majoranas in a fixed parity sector, e.g., via the quantum-wire construction in Ref. [28].

any  $[[n, k, d]]$  code, where  $n$  is the number of physical qubits,  $k$  is the number of encoded logical qubits, and  $d$  is the code distance, i.e., the minimum qubit weight of all logical operators. Similarly, Majorana surface codes can be concatenated with any  $[[n_m, k, d_m]]_m$  Majorana fermion code, where  $n_m$  is the number of Majoranas, and  $d_m$  is the Majorana distance, i.e., the minimum Majorana weight of all logical operators. Concatenating surface codes with high-distance codes can be used to obtain topological codes with arbitrarily low space overhead. As one uses higher-distance codes for concatenation, one increasingly sacrifices stabilizer weight and locality in favor of lower Majorana overhead. From a pragmatic point of view, if a given Majorana-based quantum computer has a maximum stabilizer weight that it can measure, a comparison in the spirit of Tab. I can be used to determine a suitable encoding, even though we note that our collection of codes is not exhaustive. In particular, it does not include Majorana surface codes based on non-uniform tilings.

## II. MAJORANA SURFACE CODES

A Majorana fermion code [9] encodes logical information in a set of Majorana zero modes placed on the vertices of 2D lattices, which are described by self-adjoint operators  $\gamma_i = \gamma_i^\dagger$ , and fulfill the fermionic anticommutation relations  $\{\gamma_i, \gamma_j\} = 2\delta_{i,j}$ . The code is defined by its stabilizers – more precisely, by its mutually commuting stabilizer generators – which are products of all Majorana operators

$$\mathcal{O}_{\text{face}} = \prod_{j \in \text{face}} i^{1/2} \gamma_j \quad (1)$$

associated with a face and have eigenvalues  $\pm 1$ . A code with  $n$  Majoranas and  $m$  stabilizers encodes  $n/2 - m$  logical qubits in the degenerate ground-state manifold of the Hamiltonian

$$H_{\text{code}} = - \sum_{i=1}^m \mathcal{O}_i. \quad (2)$$

Due to fermion parity superselection, the product of all  $n$  Majoranas is always a stabilizer, such that  $n$  Majoranas can at most encode  $n/2 - 1$  qubits.

### A. Smallest Majorana fermion codes

The smallest fermion code is the four-Majorana tetron code shown in Fig. 1a, which involves a single stabilizer  $\mathcal{O}_{\text{tetron}} = -\gamma_1\gamma_2\gamma_3\gamma_4$ . The qubit is encoded in the doubly degenerate ground-state space of the Hamiltonian

$$H_{\text{tetron}} = \gamma_1\gamma_2\gamma_3\gamma_4. \quad (3)$$

While it is possible to define a qubit in the Schrödinger picture by defining two computational states  $|0\rangle$  and  $|1\rangle$ , it is more convenient to use the Heisenberg picture. Since any single-qubit unitary operator can be written using the  $X$  and  $Z$  Pauli operators via the Euler decomposition, a qubit can be defined through these two operators. The  $X$  and  $Z$  operators need to square to the identity,  $X^2 = Z^2 = \mathbb{1}$ , anticommute,  $XZ = -ZX$ , and commute with all stabilizers of the code. For a tetron, one choice is  $Z = i\gamma_1\gamma_2 = i\gamma_3\gamma_4$  and  $X = i\gamma_2\gamma_3 = i\gamma_1\gamma_4$ . The remaining Pauli operator follows as the product  $Y = iXZ$ .

There are two ways to implement stabilizer terms such as  $\mathcal{O}_{\text{tetron}}$ . One way is to interpret stabilizers as physical parity-fixing constraints realized in the lab, e.g., via the charging energy of two topological superconducting nanowires. Alternatively, they can be interpreted as measurement prescriptions. Measuring all stabilizers of the code projects the system into the ground-state space as long as the measurement outcome is  $+1$  for all stabilizers. For tetrons, it is easy to see that both approaches are susceptible to errors. In particular, an error process described by the operator  $i\gamma_i\gamma_j$  involving two Majorana fermion operators will lead to errors that neither violate the parity-fixing constraint nor change the measurement outcome of the parity measurement. The fact that stabilizers can be interpreted either as physical constraints or measurement prescriptions will become important when we discuss larger codes with more than one stabilizer.

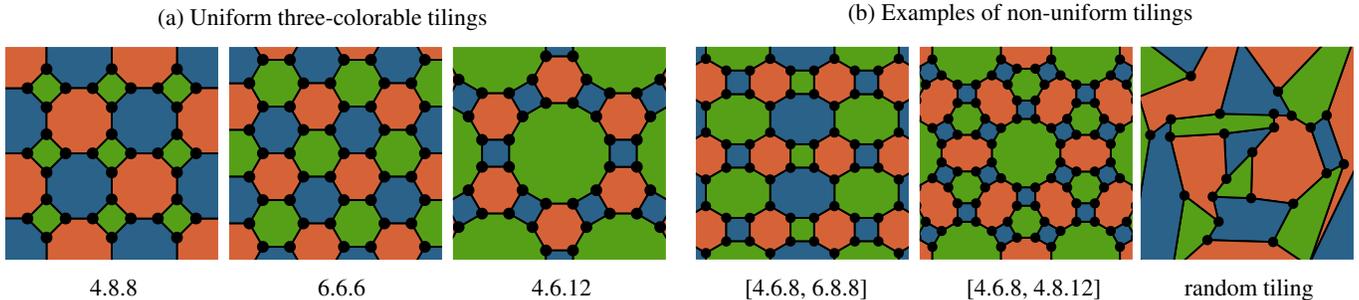


FIG. 2. The three uniform three-colorable tilings of the 2D plane (a), and three examples of non-uniform tilings (b).

The second smallest Majorana fermion code is the six-Majorana hexon code shown in Fig. 1b. It encodes two logical qubits in the ground-state space of the Hamiltonian

$$H_{\text{hexon}} = -i\gamma_1\gamma_2\gamma_3\gamma_4\gamma_5\gamma_6. \quad (4)$$

The logical Pauli operators of the two qubits can be chosen as  $Z_1 = i\gamma_1\gamma_2$ ,  $X_1 = i\gamma_2\gamma_3$ ,  $Z_2 = i\gamma_4\gamma_5$ ,  $X_2 = i\gamma_5\gamma_6$ .

### B. Logical tetrons and hexons

Tetrons and hexons encode qubits, but cannot correct any errors. We now describe a procedure to obtain *logical* tetrons and hexons based on 2D Majorana surface codes.

We define Majorana surface codes as fermionic topological codes with non-overlapping stabilizers. Thus, they can be defined by a tiling of the 2D plane with vertices corresponding to Majoranas and faces (or tiles) corresponding to stabilizers. Since stabilizers need to commute, neighboring faces should share two vertices (or, in fact, any even number). This implies that any valid tiling needs to be three-colorable [9], in the sense that faces can be colored in three colors with neighboring faces having different colors. Moreover, each stabilizer needs to contain an even number of Majoranas. Remarkably, these requirements are identical to the restrictions of bosonic color codes, such that all valid color code tilings with vertices as qubits are also valid Majorana surface code tilings with vertices as Majoranas.

There exist three uniform three-colorable tilings that can be used to define a Majorana surface code: 4.8.8 [11, 12, 16, 17], 6.6.6 [10], and 4.6.12, as shown in Fig. 2a. The numbers refer to the three polygons that meet at each vertex, e.g., for the 4.6.12 tiling, a square, a hexagon, and a dodecagon. Uniform means that the same types of polygons meet at each vertex. There exist also non-uniform three-colorable tilings, which have at least two different vertex types. Examples are shown in Fig. 2b, such as the [4.6.8, 6.8.8] tiling [9] which has two types of vertices, namely 4.6.8 and 6.8.8. In this work, we restrict ourselves to the three uniform tilings shown in Fig. 2a, although our constructions can be straightforwardly extended to non-uniform tilings. We stress that the colors bear no physical meaning, but are merely a useful bookkeeping tool.

A logical tetron encodes one logical qubit in  $n$  Majoranas with  $n/2 - 1$  stabilizers. It is constructed by appropriately in-

roducing boundaries to a three-colorable tiling, as shown in Fig. 3. At a boundary, Majoranas are no longer part of three differently-colored stabilizers, but only two. We then assign the remaining color to the boundary, e.g., on a red boundary, Majoranas are part of blue and green stabilizers. Two differently-colored boundaries meet at a corner, where a Majorana is part of only one stabilizer. A logical tetron is obtained by terminating the tiling by four boundaries, a pair of opposing red and blue boundaries each (or any other pair of colors). This procedure always yields a code with  $n$  Majoranas and  $n/2 - 1$  stabilizers.

One can also assign colors to the edges of faces, i.e., to the two-Majorana operators that lie between two faces. The color of an edge is then the third remaining color. For instance, an edge that lies between a red and a blue face is referred to as a green edge. The logical  $X$  ( $Z$ ) operators of a logical tetron are strings of red (blue) edges that connect the red (blue) boundaries, as shown in Fig. 3b. In particular, the product of all Majoranas along a blue boundary is a string of red edges, i.e., an  $X$  operator. Similarly, the product of all Majoranas along a red boundary is a  $Z$  operator. Logical  $Z$  and  $X$  operators anticommute, since they always share an odd number of Majoranas. The red and blue boundary operators, specifically, share one Majorana in the corner where the boundaries meet.

The four-corner surface codes depicted in Fig. 3 are indeed the higher-distance equivalents of four-Majorana tetrons. In Ref. [32], it was pointed out that the corners of surface code qubits correspond to twist defects, which feature the same nonabelian statistics as Majoranas [20]. This can be understood from the observation that since one type of surface code boundary can absorb  $e$  anyons, and the other type of boundary can absorb  $m$  anyons, corners can absorb  $\epsilon$  anyons. Therefore, according to Ref. [20], corners are twist defects. Alternatively, in Ref. [33], edges of surface codes are identified as flat bands of uncoupled Majoranas. The boundary stabilizers gap out the edge Majoranas, leaving unpaired Majoranas in the corners. Thus, the identification of four-boundary Majorana surface codes with tetrons is indeed justified, as their corners can be interpreted as *logical* Majoranas, i.e., twist defects.

Similarly, six-boundary Majorana surface codes can be identified as logical hexons, as shown in Fig. 4. Logical hexons encode two logical qubits on  $n$  Majoranas with  $n/2 - 2$  stabilizers. This can be achieved by introducing three red and blue boundaries each in an alternating fashion. Logical hex-

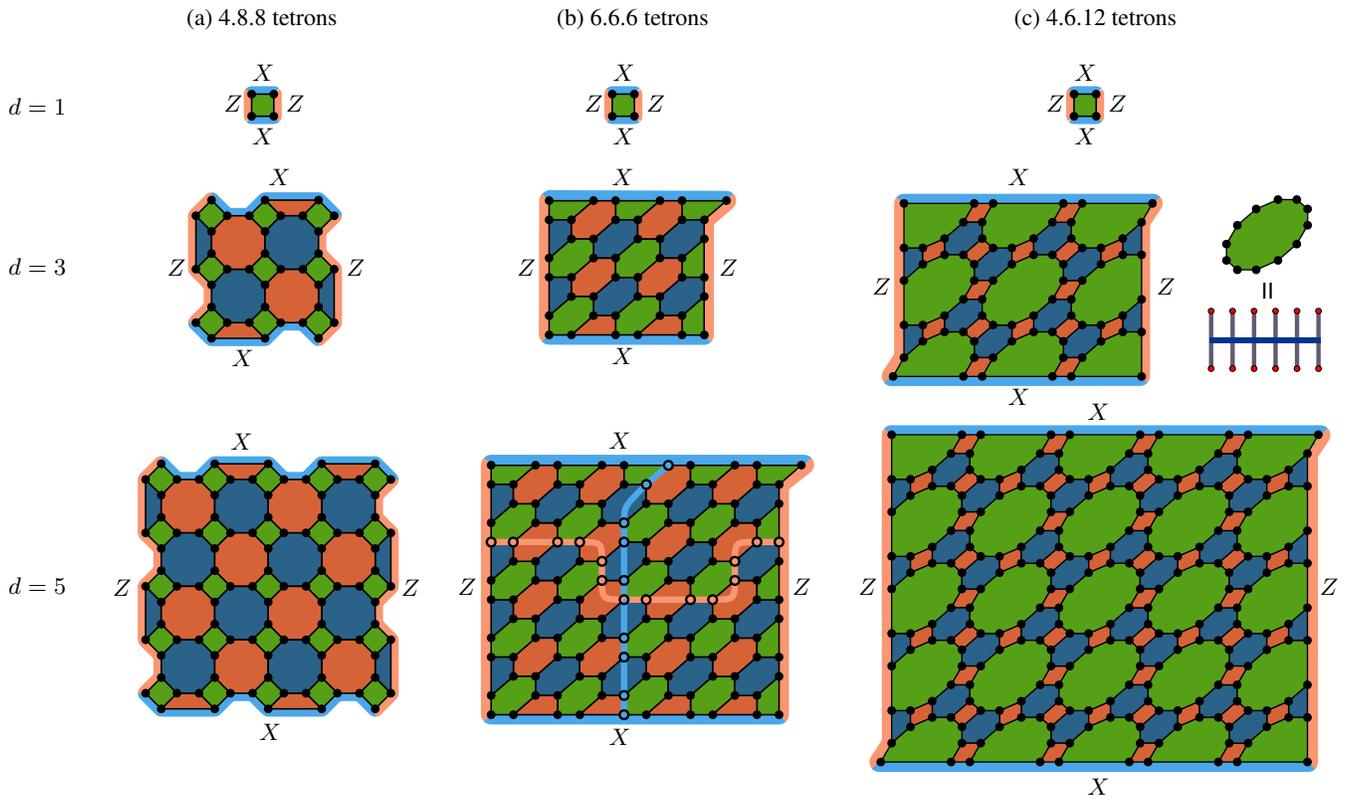


FIG. 3. Logical tetrons of the 4.8.8 (a), 6.6.6 (b), and 4.6.12 (c) Majorana surface code with code distances  $d = 1, 3,$  and  $5$ . The 6.6.6 Majorana surface code with  $d = 5$  shows an example of strings of red and blue edges through the bulk of the code. Products of Majoranas along the red and blue strings are equivalent to logical  $X$  and  $Z$  operators, respectively.

ons require approximately twice as many Majoranas as logical tetrons for a given code distance, but also encode twice as many qubits – one qubit at the two bottom and one qubit at the two top boundaries.

The error processes considered in this work correspond to quasiparticle poisoning and are described by the action of single Majorana operators  $\gamma_i$ . We refer to these errors as Majorana errors. A Majorana error flips the measurement outcome of the stabilizers that the affected Majorana is part of, which allows for detection of errors and their subsequent correction. Corrections do not need to be applied explicitly, but merely tracked by a classical computer. We define the Majorana code distance  $d_m$  as the minimum number of Majorana errors necessary to introduce a logical error that will go undetected, i.e.,  $d_m$  is the number of Majorana operators contained in the shortest logical operator. Codes with Majorana distance  $d_m$  tolerate  $d_m/2 - 1$  Majorana errors, as the syndrome of error strings that cover more than half the Majorana distance will be misinterpreted and lead to a correction that introduces a logical Pauli error. Since only even-number products of Majoranas are physically measurable,  $d_m$  is always even. For this reason,  $d_m$  is not the same quantity as the code distance  $d$  of bosonic codes. For a fair comparison between fermionic and bosonic codes, we label fermionic codes by their bosonic code distance  $d = d_m/2$ . This is justified, because Pauli operators of physical Majorana-based qubits, such as tetrons or

hexons, correspond to products of two Majorana operators, and therefore single-qubit Pauli errors always correspond to *two* Majorana errors. Thus, the Majorana surface code tetrons shown in Fig. 3 have Majorana distances  $d_m = 2, 6,$  and  $10$ , but code distances  $d = 1, 3,$  and  $5$ .

To minimize the number of operators that need to be measured, we implement stabilizers of one color as parity-fixing constraints, while the other two colors are interpreted as measurement prescriptions. The parity-fixed color should be the one color that is not a boundary. For instance, the green four-Majorana stabilizers of the 4.8.8 Majorana surface codes in Fig. 3a could be interpreted as physical tetrons, whereas the red and blue stabilizers are operators that need to be measured in order to actively detect and correct errors. This can be done with any three-colorable tiling, such that, say, red and blue stabilizers need to be measured, while green stabilizers are interpreted as Majoranas in a fixed parity sector. We then refer to the green stabilizers as the building blocks of the code. Note that Majorana errors that respect the parity-fixing constraints set by the green stabilizers always come in pairs.

4.8.8 Majorana surface codes can be defined on a square lattice of tetrons in the spirit of Ref. [11], such that the 8-Majorana stabilizers need to be measured. One qubit with code distance  $d$  requires  $4d^2$  Majoranas. Similarly, 6.6.6 Majorana surface codes can be defined on a square lattice of hexons. While they require more Majoranas per qubit

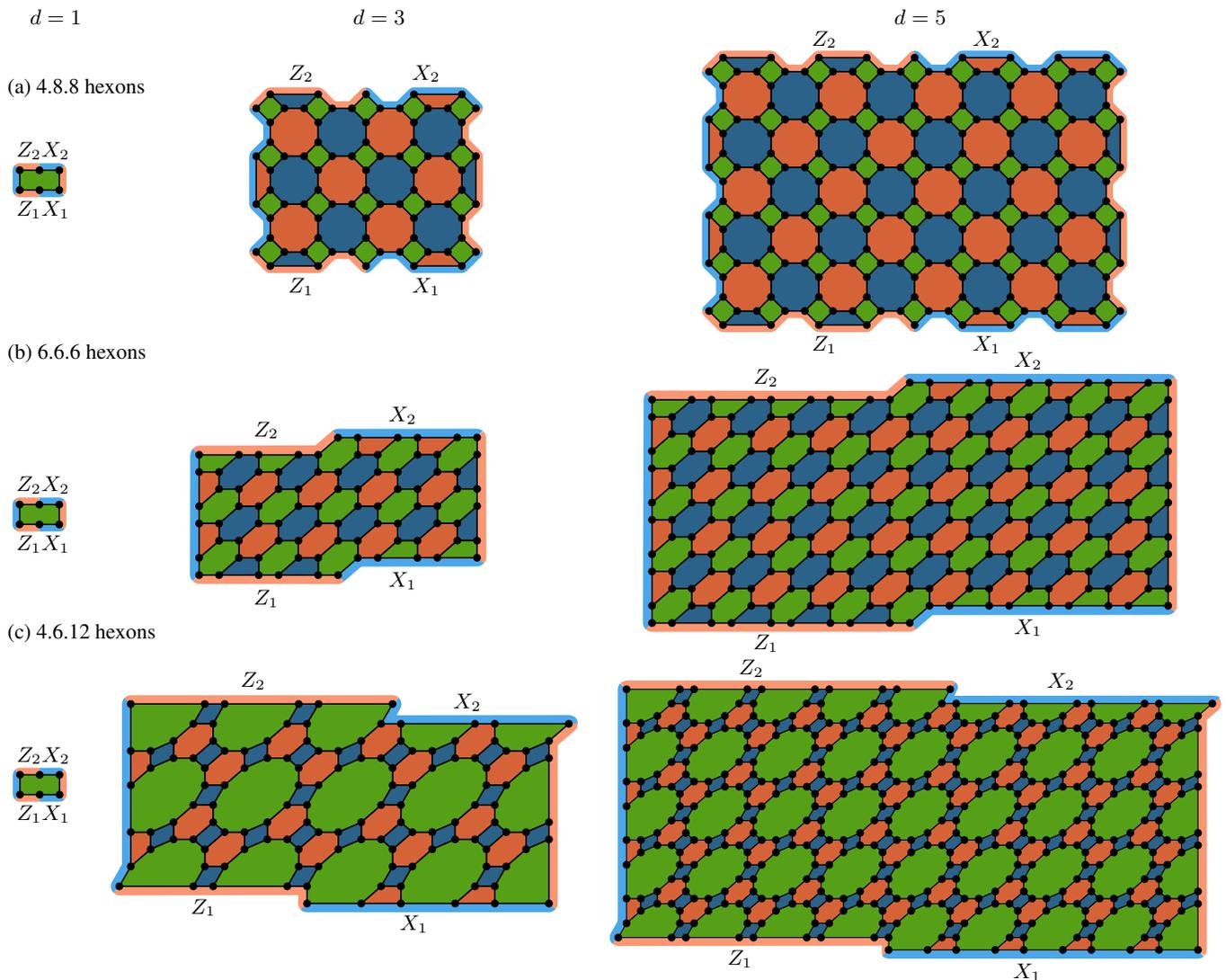


FIG. 4. Logical hexons of the 4.8.8 (a), 6.6.6 (b), and 4.6.12 (c) Majorana surface code with code distances  $d = 1, 3,$  and  $5$ .

$(6d^2 - 4d + 2)$ , the weight of the operators that need to be measured reduces to 6 Majoranas. We note that even though the codes in Fig. 3b display green tetrons at the top and right boundaries, these codes can indeed be defined on a square lattice of hexons, as we show in Appendix A 1. Finally, 4.6.12 Majorana surface codes can be defined on a square lattice of dodecons, i.e., 12 Majoranas in a fixed parity sector. In the context of color codes, 4.6.12 codes are often ignored, since they feature both a high space overhead and high-weight stabilizers. However, by interpreting the 12-Majorana operators as building blocks that satisfy parity-fixing constraints, only 4- and 6-Majorana operators need to be measured. Thus, the stabilizer weight has decreased to an average of 5 Majoranas, albeit at the price of an increased space overhead of  $12d^2 - 12d + 4$  Majoranas per qubit.

An important part of error correction is decoding. While the error syndrome is known from the stabilizer measurements, the most likely error configuration that causes this syndrome needs to be determined by a classical program called a de-

coder. Since Majorana surface codes are indeed surface codes, they can be decoded by any bosonic surface code decoder,<sup>1</sup> either by interpreting blue and red stabilizers as two types of anyons that need to be matched (analogous to  $e$  and  $m$  anyons in bosonic surface codes), or by first mapping the Majorana code onto a bosonic code and then decoding the bosonic surface code. As we show in Appendix A 2, Majorana surface codes can be mapped onto bosonic codes, but the mapping is not unique. In the case of 4.8.8 codes, replacing each tetron with a qubit yields exactly the bosonic surface code on a square lattice. Replacing each hexon of the 6.6.6 code with two qubits also yields a bosonic code on a square lattice, but the lattice is rotated. With a 4.6.12 code, replacing each of the

<sup>1</sup> We note that this is only true, if building blocks are interpreted as parity-fixed, such that, say, green stabilizers cannot be violated. If this is not the case, then the decoder needs to match all three colors, similar to the decoding of bosonic color codes.

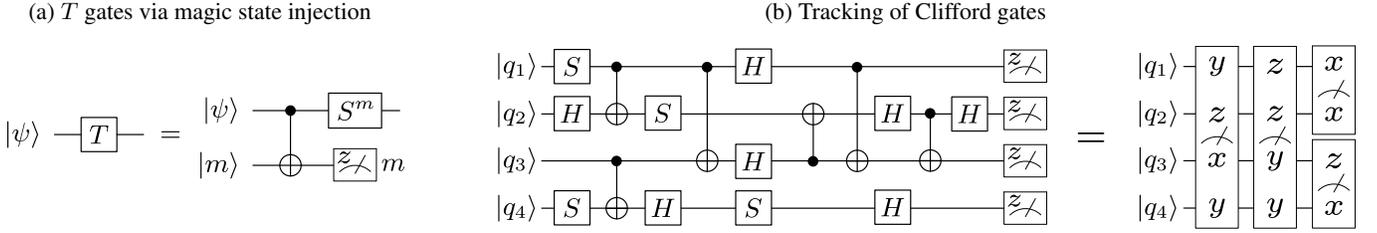


FIG. 5. (a) Applying a  $T$  gate to a qubit  $|\psi\rangle$  is equivalent to a CNOT between the qubit and a magic state  $|m\rangle = \frac{1}{\sqrt{2}}(|0\rangle + e^{i\pi/4}|1\rangle)$ , a measurement of the magic state with outcome  $m \in \{0, 1\}$ , and a corrective operation  $S^m$  on the qubit. (b) Example of a circuit of Clifford gates and four  $Z$  measurements that is compressed into four Pauli product measurements.

four-Majorana operators with a qubit yields a bosonic surface code on a tri-hexagonal lattice.

### III. MINIMAL-OVERHEAD CLIFFORD GATES WITH TETRONS AND HEXONS

In this section, we show how to perform universal quantum computation with *physical* tetrons and hexons. This is not fault-tolerant, since the protocols assume perfect measurements and error-free qubits. Closely analogous fault-tolerant protocols based on logical tetrons and hexons will be discussed in Sec. IV.

We start with general considerations concerning the classical tracking of Clifford gates. These gates map Pauli operators onto other Pauli operators and are products of Hadamard gates  $H$ , phase gates  $S$ , and controlled-NOT gates CNOT. Any quantum circuit can be expressed in terms of Clifford gates,  $T$  gates (where  $T = \sqrt{S}$ ) and single-qubit measurements in the  $Z$  basis [34]. If magic states  $|m\rangle = (|0\rangle + e^{i\pi/4}|1\rangle)/\sqrt{2}$  are available as a resource,  $T$  gates can be rewritten as a CNOT gate and a measurement of the magic-state qubit, see Fig. 5a. In realistic architectures, only faulty magic states are available. Using magic state distillation [35], multiple faulty magic states can be converted into fewer low-error magic states. This is yet another circuit of Clifford gates and measurements. Thus, any quantum computation can be expressed as a quantum circuit consisting only of Clifford gates and measurements.

According to the Gottesman-Knill theorem, gate operations for Clifford gates need not be performed, as they can be tracked classically. Mapping Pauli operators onto other Pauli operators, they merely change the basis of the  $Z$  measurements:  $H$  gates map  $Z \rightarrow X$  and  $X \rightarrow Z$ ,  $S$  gates map  $Z \rightarrow Z$  and  $X \rightarrow Y$ , and CNOTs map  $\mathbb{1} \otimes Z \rightarrow Z \otimes Z$ ,  $Z \otimes \mathbb{1} \rightarrow Z \otimes \mathbb{1}$ ,  $X \otimes \mathbb{1} \rightarrow X \otimes X$ , and  $\mathbb{1} \otimes X \rightarrow \mathbb{1} \otimes X$ . The action of Clifford gates on all other Pauli operators can be inferred from these rules. Consequently, any circuit of Clifford gates and  $n$  measurements can be contracted to  $n$  measurements of Pauli product operators. Thus, apart from the generation of (faulty) magic states, a quantum computer merely needs to be able to fault-tolerantly measure nonlocal products of Pauli operators. An example of such a contraction is shown in Fig. 5b.

The state-injection circuit in Fig. 5a then corresponds to

a  $Z \otimes Z$  measurement between the qubit  $|\psi\rangle$  and the magic state, and a tracked  $S$  gate correction. In order to discard the magic state qubit, it needs to be disentangled via an  $X$  measurement and a subsequent  $Z$  correction on the qubit  $|\psi\rangle$ , as we discuss in Appendix A 3. We note that if one prefers to explicitly perform CNOT operations, this can also be done using Pauli product measurements via the circuit identity shown in Appendix A 3.

#### A. Pauli product measurements with tetrons and hexons

We now describe a protocol for Pauli product measurements with tetrons and hexons using only local two-qubit operations. Our scheme uses hexons to encode qubits used for quantum computation (referred to as data qubits), and tetrons to encode ancilla qubits that enable long-range communication. The hexons form blocks of data qubits in a 2D array, with tetrons as ancillas between blocks. A possible arrangement of 24 data qubits is shown in Fig. 6. We describe the protocol using the example in Fig. 7.

The goal is to measure the nonlocal  $n$ -qubit Pauli product  $Z_1 \otimes X_3 \otimes Z_8 \otimes Y_9$  without measuring any of the  $n$  individual Pauli operators  $Z_1$ ,  $X_3$ ,  $Z_8$  or  $Y_9$ . The first step is to initialize an  $n + m$ -ancilla GHZ state  $|0\rangle^{\otimes n+m} + |1\rangle^{\otimes n+m}$ , where  $m$  ancillas are used to bridge long distances. In our example, we use  $n + m = 4 + 2$  ancilla qubits. Each ancilla is measured in the  $X$  basis by measuring the corresponding two-Majorana operator (labelled  $X_{a1} \cdots X_{a6}$ ) and thereby prepared in the  $X$

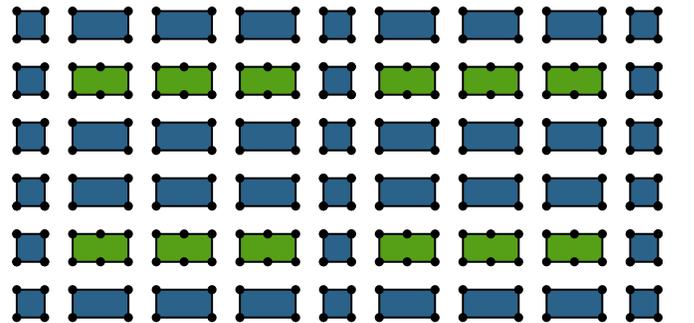


FIG. 6. Array of tetrons and hexons, where hexons are used to encode data qubits, and tetrons are used as ancillas for Pauli product measurements.

eigenstate  $|+\rangle = (|0\rangle + |1\rangle)/\sqrt{2}$ . For measurement outcomes  $-1$ , a corrective  $Z$  operation is necessary, which can also be tracked. Measuring  $Z \otimes Z$  of neighboring pairs of  $|+\rangle$  ancillas (corresponding to the red four-Majorana operators shown in step 2 of Fig. 7) prepares the ancillas in the desired GHZ state, assuming that all  $Z \otimes Z$  measurements yield  $+1$ . Again, measurement outcomes  $-1$  require corrective Pauli  $X$  operations.

The GHZ state is an entangled state with known global properties, but unknown local properties. It is a  $+1$  eigenstate of  $\tilde{X} = X^{\otimes n+m}$ , but the measurement outcome of each individual  $X$  operator is entirely random. Thus, we can use the GHZ state to measure the desired Pauli product  $\tilde{X} \otimes Z_1 \otimes X_3 \otimes Z_8 \otimes Y_9$  without measuring any of the individual operators. By measuring the six operators shown in step 3 of Fig. 7, the desired operator is obtained as the product of the six measurements.

In the general case of the measurement of a Pauli product  $P_1 \otimes \dots \otimes P_n$ , the  $n$  ancilla qubits adjacent to the corresponding data qubits are used to measure  $P_n \otimes X_{a,n}$ , while the remaining  $m$  ancilla qubits are read out in the  $X$  basis. The product of all  $n + m$  measurements is  $\tilde{X} \otimes P_1 \otimes \dots \otimes P_n$ . Because  $\tilde{X} = 1$ , but each individual  $X_{a,n}$  is random, this measures the Pauli product without measuring any of the individual Pauli operators.

The  $n$  ancilla qubits adjacent to data qubits still need to be disentangled from the data qubits in step 4 of the protocol, before they can be discarded. This is done by measuring the ancilla qubits in the  $Z$  basis with outcome  $m \in \{0, 1\}$ , leading to a  $P^m$  Pauli correction on the adjacent data qubit, where  $P$  is the Pauli operator that was part of the two-qubit measurement in step 3, e.g.,  $Z_1$  for qubit 1 in Fig. 7.

When combined with the preparation of (faulty) magic states, this is sufficient to implement universal quantum computation. Specifically, this protocol can be straightforwardly used with the quantum-wire constructions of Ref. [28]. In Appendix B 1, we explicitly show the tunnel coupling configurations to implement Fig. 7 in a network of topological superconducting nanowires.

To summarize, the following operations are necessary to implement a Majorana-based universal quantum computer with minimal-overhead Clifford gates:

- Op1 Measurements of tetrons in the bases  $X$  and  $Z$ ,
- Op2 Measurements of  $Z \otimes Z$  between adjacent tetrons,
- Op3 Measurements of  $X \otimes X$ ,  $Y \otimes X$ , and  $Z \otimes X$  between adjacent hexons and tetrons,
- Op4 Application of (potentially faulty)  $T$  gates on tetron qubits.

In the following section, we show how to implement these four operations fault-tolerantly with *logical* tetrons and hexons.

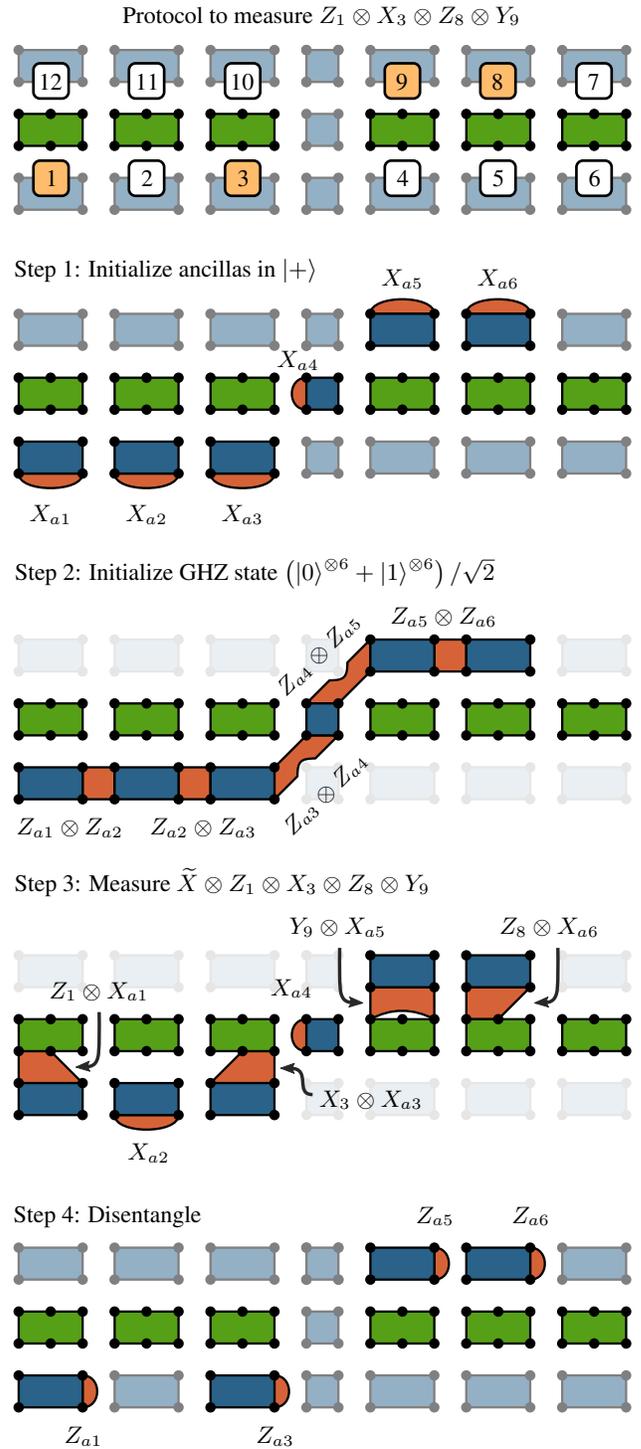


FIG. 7. Protocol for the measurement of the Pauli product operator  $Z_1 \otimes X_3 \otimes Z_8 \otimes Y_9$ .

#### IV. TWISTS IN MAJORANA SURFACE CODES

Remarkably, a completely analogous scheme can be implemented with logical tetrons and hexons, allowing for fault-tolerant quantum computing with tracked Clifford gates.

Specifically, we implement the four operations Op1-Op4 with the logical tetrons and hexons from Figs. 3 and 4. The first operation is the measurement of logical tetrons in the  $X$  and  $Z$  basis. With bosonic surface codes, logical qubits are read out in the  $X$  or  $Z$  basis by measuring all physical qubits in the  $X$  or  $Z$  basis, and performing classical error correction. Similarly, Majorana surface code tetrons are read out in the  $X$  or  $Z$  basis by measuring all two-Majorana operators corresponding to red or blue edges, and performing classical error correction. The measurement outcome corresponds to the product of all Majoranas along the corresponding boundary of the tetron. In analogy to bosonic surface codes, Majorana surface code tetrons can be initialized in the  $|+\rangle$  state by measuring all red edges, and then measuring the stabilizers and correcting the errors.

The second and third operations require measurements of two-qubit Pauli products. For logical tetrons with code distance  $d$ , this corresponds to products of  $4d$  Majoranas, which are highly nonlocal operators. In order to measure these operators using only measurements of low-weight local operators, we adapt the technique of lattice surgery [36], and in particular twist-based lattice surgery [22], to fermionic codes. While we only explicitly show lattice surgery protocols for the three uniform Majorana surface codes, similar constructions can also be used for non-uniform tilings.

#### A. 4.8.8 Majorana surface codes

We start with 4.8.8 Majorana surface codes. The second operation Op2 requires the measurement of  $Z \otimes Z$  between two tetrons, such as the two  $d = 5$  tetrons in Fig. 8. In this example, the operator  $Z \otimes Z$  is the product of the 20 Majorana fermions along the two neighboring red boundaries. Lattice surgery is a protocol that temporarily changes the stabilizer configuration in order to measure this operator fault-tolerantly using only local measurements. Pairs of blue 4-Majorana stabilizers are merged to form 8-Majorana stabilizers (light blue in step 2). The measurement outcome of these 8-Majorana stabilizers is trivial, as it is simply the product of previously known stabilizers. In addition, new stabilizers are introduced along the boundary (light red). In this new stabilizer configuration in step 2, all stabilizers commute, but the number of stabilizers is increased by one compared to step 1. Thus, the number of degrees of freedom is reduced by one, and one bit of information is measured. Since the light blue stabilizers are trivial, this bit of information is given by the product of the light red stabilizers. Because the stabilizers involve each red-boundary Majorana exactly once, their product is exactly  $Z \otimes Z$ . This is the fault-tolerant generalization of the 4-Majorana parity measurement between two physical tetrons (with  $d = 1$ ) that is shown in the inset in Fig. 8. Not only is this protocol fault-tolerant in the sense that it can correct qubit errors such as quasiparticle poisoning, but by repeating rounds of syndrome measurement, erroneous stabilizer measurements can also be corrected.

A very similar protocol can be used for the third operation to measure  $X \otimes X$ ,  $Y \otimes X$ , and  $Z \otimes X$  between a tetron and a

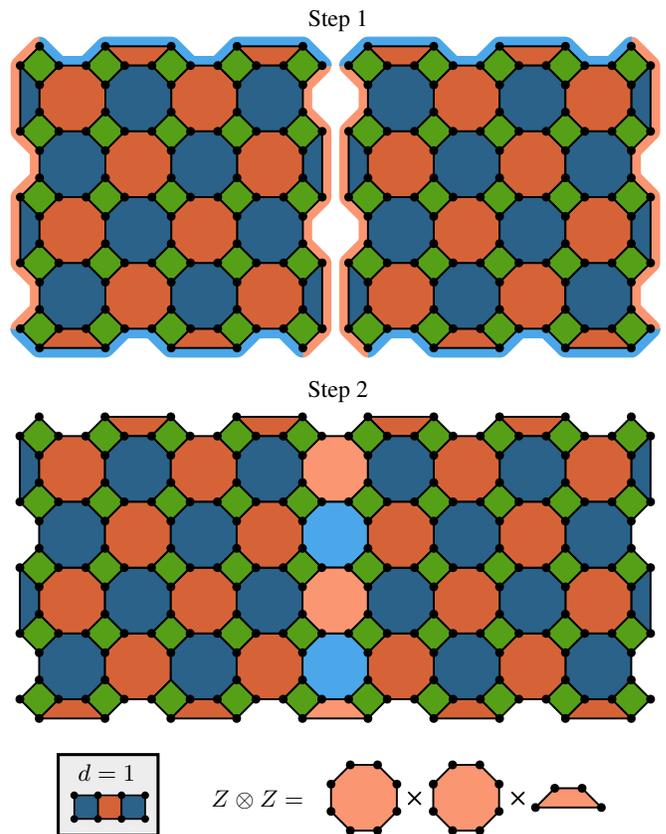


FIG. 8. Lattice surgery between two 4.8.8 Majorana surface code tetrons with code distance  $d = 5$  measuring  $Z \otimes Z$ . The box in the bottom left corner shows the corresponding measurement with  $d = 1$ , where  $Z \otimes Z$  is the red 4-Majorana operator.

hexon. In Fig. 9a, an  $X \otimes X$  measurement is shown for  $d = 5$ . Here,  $X \otimes X$  is the product of the 20 Majoranas along the two adjacent blue boundaries. Again, the 4-Majorana operators are merged to 8-Majorana operators (light red), and the product of the new light blue operators yields precisely  $X \otimes X$ . The protocol for the  $Z \otimes X$  measurement in Fig. 9b measures the product of 20 Majoranas that are located on a red boundary on the hexon, but on a blue boundary on the tetron. While again stabilizers are merged and new stabilizer are introduced to yield  $Z \otimes X$ , the stabilizer configuration is different from the situation in Fig. 9a. Because the two boundaries have different colors, the stabilizers in step 2 form what is called a *dislocation line*. If flipped red and blue stabilizers are interpreted as anyons in analogy to  $e$  and  $m$  anyons of bosonic surface codes, then anyons passing this dislocation line will change color.

The measurement of  $Y \otimes X$  is shown in Fig. 9c. Because  $Y = iXZ$ , the  $Y$  operator of the (bottom) hexon qubit is the product of 18 Majoranas on both the red and blue bottom boundary, excluding the center Majorana where the two boundaries meet (since it is part of both boundaries and  $\gamma^2 = \mathbb{1}$ ). Thus,  $Y \otimes X$  is the product of 36 Majoranas on the long blue boundary of the tetron, and the adjacent red and blue boundaries of the hexon. The product is again measured

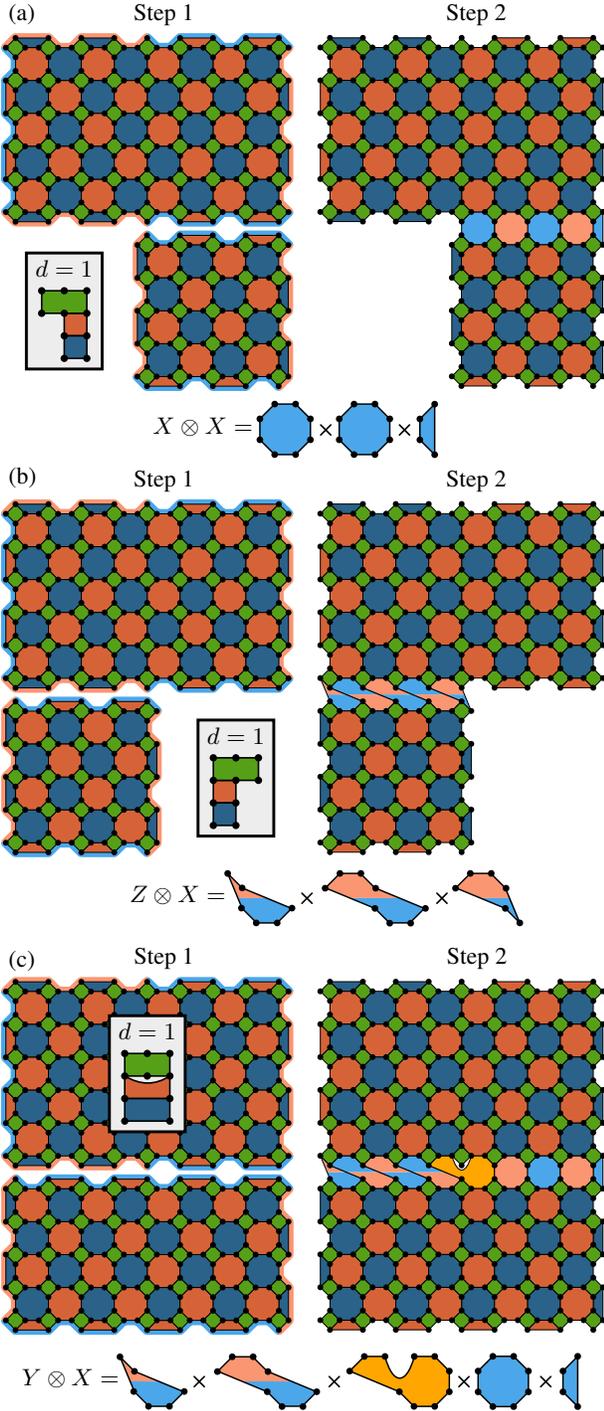


FIG. 9. Lattice surgery protocols for the measurements of the operators  $X \otimes X$  (a),  $Z \otimes X$  (b), and  $Y \otimes X$  (c) between a 4.8.8 Majorana surface code hexon and tetron. The insets show the equivalent operation with  $d = 1$ .

by merging stabilizers and introducing new ones whose product is precisely the product of the 36 Majoranas. The stabilizer configuration in step 2 corresponds to a dislocation line that is terminated by a 10-Majorana stabilizer (orange). This 10-Majorana stabilizer is what is called a twist defect [20]. Twists are found at the ends of dislocation lines. Incidentally,

corners of surface code qubits – i.e., the meeting points of two different boundaries – can also be interpreted as ends of dislocation lines, and therefore as twists [32]. The protocols in Figs. 8 and 9 further illustrate the equivalence between twists and Majoranas: What for  $d = 1$  is a measurement of a product of four Majoranas, for higher code distances becomes a measurement of the four twists in the corners that are part of the lattice surgery protocols. Twist defects become visible when the twist is not located in a corner during lattice surgery. This is what happens in Fig. 9c for the  $Y \otimes X$  measurement, where the 10-Majorana twist defect is revealed as it is in the center of the qubit corresponding to the one Majorana that is not part of the parity measurement in the  $d = 1$  case.

The 10-Majorana twist defect of the 4.8.8 Majorana surface code corresponds to the 5-qubit twist defect of bosonic surface codes. What is more, all of the aforementioned lattice surgery protocols with 4.8.8 Majorana surface codes are exactly equivalent to the protocols with bosonic surface codes [22]. In fact, no Majorana fermion code that is purely based on tetron building blocks (such as 4.8.8 codes) can display any features that are different from bosonic codes, as any bosonic code can be mapped onto a fermionic code by replacing each qubit with a tetron [9]. However, by replacing only a few of the tetrans with hexons, the 4.8.8 Majorana surface code can display some Majorana-specific characteristics. In particular, the maximum stabilizer weight necessary for lattice surgery can be reduced from 10 Majoranas (corresponding to 5 qubits in the bosonic case) to 8 Majoranas.

Consider the modified  $d = 5$  hexons shown in Fig. 10. The tetrans along the bottom and top red boundaries have been replaced by hexons. Since two of the Majoranas of the hexons are in a fixed parity sector due to the two-Majorana plaquettes, they are equivalent to tetrans. In order to measure  $Y \otimes X$  between the hexon and a tetron via lattice surgery, stabilizers are merged and new stabilizers are introduced to yield the desired operator. The stabilizer configuration again features a dislocation line that is terminated by a twist defect, but the weight of the twist defect has decreased to just 8 Majoranas. Due to the three-colorability of Majorana surface code lattices, their twist defects closely resemble the so-called color twists of bosonic color codes [37].

Thus, 4.8.8 Majorana surface codes feature the same Majorana overhead as bosonic surface codes of  $\sim 4d^2$  Majoranas per qubit, but a lower stabilizer weight for twist-based lattice surgery. As we show in the remainder of the section, 6.6.6 and 4.6.12 Majorana surface codes can be used to reduce the stabilizer weight even further, albeit at the cost of a higher Majorana overhead.

Note that, in contrast to the Pauli product measurement protocol for physical tetrans, the fault-tolerant protocol features tetrans of different lengths. Specifically, the tetrans used for  $Y \otimes X$  measurements are almost twice as long as tetrans used for  $X \otimes X$  or  $Z \otimes X$  measurements. The protocol of Fig. 7 can still be used in the fault-tolerant setting, but additional tetrans may be required to bridge distances between ancillas. For concreteness, we show an implementation of the protocol of Fig. 7 using 4.8.8 Majorana surface codes with  $d = 3$  in Appendix A 4.

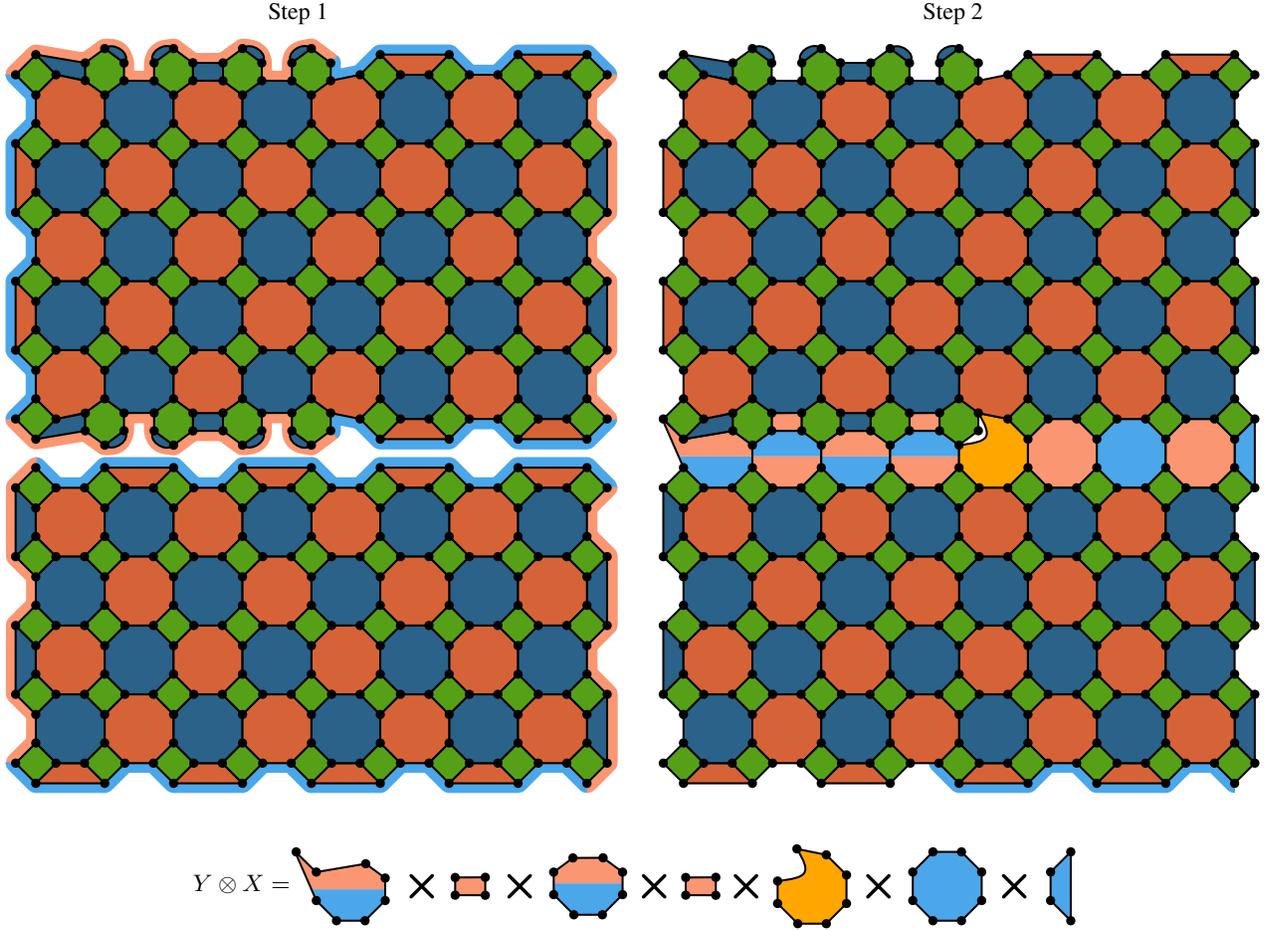


FIG. 10. Lattice surgery protocol for the measurement of  $Y \otimes X$  between a 4.8.8 Majorana surface code hexon and tetron. The hexon is modified to feature hexons instead of tetrons at the bottom and top red boundaries. As a consequence, the weight of the twist defect (orange) reduces to 8 Majoranas.

### B. 6.6.6 and 4.6.12 Majorana surface codes

6.6.6 codes only require the measurement of 6-Majorana operators in the bulk of the code, as opposed to 8-Majorana operators for 4.8.8 codes. In Fig. 11a, we show the protocol to measure  $Y \otimes X$  between a 6.6.6 hexon and tetron. This case is the most instructive, as it features standard lattice surgery, a dislocation line, and a twist defect. The measurement protocols for the other operators are shown in Appendix A 5. Even though 6.6.6 codes reduce the stabilizer weight to 6 Majoranas for most stabilizers, twist-based lattice surgery requires the measurement of some 8-Majorana operators that are part of dislocation lines. However, we can use the same trick as for 4.8.8 Majorana surface codes to reduce the weight of the lattice surgery operators. As we show in Appendix A 6, the weight of the dislocation line operators can be reduced to 6 Majoranas by replacing some of the boundary hexons with octons (8 Majoranas in a fixed parity sector), such that at most 6-Majorana parity measurements are required for quantum computing with 6.6.6 Majorana surface codes.

Figure 11b shows the same scenario for 4.6.12 codes. Here, both the dislocation line and the twist defect consist of oper-

ators that have a weight of six Majoranas. While the green building blocks along the dislocation line are drawn as decons (10-Majorana building blocks) in Fig. 11b, as opposed to dodecons, this protocol can in fact be realized in a square lattice of dodecons. This is shown in Appendix A 5, where we also present the remaining lattice surgery protocols with 4.6.12 codes.

Thus, 4.6.12 Majorana surface codes require at most 6-Majorana parity measurements for quantum computation. Compared to bosonic surface codes, this comes at the price of a threefold increased space overhead of  $\sim 12d^2$  Majoranas per qubit. This is reminiscent of bosonic subsystem surface codes [25], which only require 3-qubit operator measurements (corresponding to 6 Majoranas), and also feature an increased space overhead of  $\sim 4d^2$  physical qubits (corresponding to  $\sim 12d^2$  Majoranas) per logical qubit. However, since these are bosonic codes, twist-based lattice surgery features a 5-qubit twist defect, corresponding to 10-Majorana operators. Since there exists no work on lattice surgery with these codes, we show the twist-based lattice surgery protocol with subsystem surface codes in Appendix A 7, even though it is exactly the same as with standard bosonic surface codes.

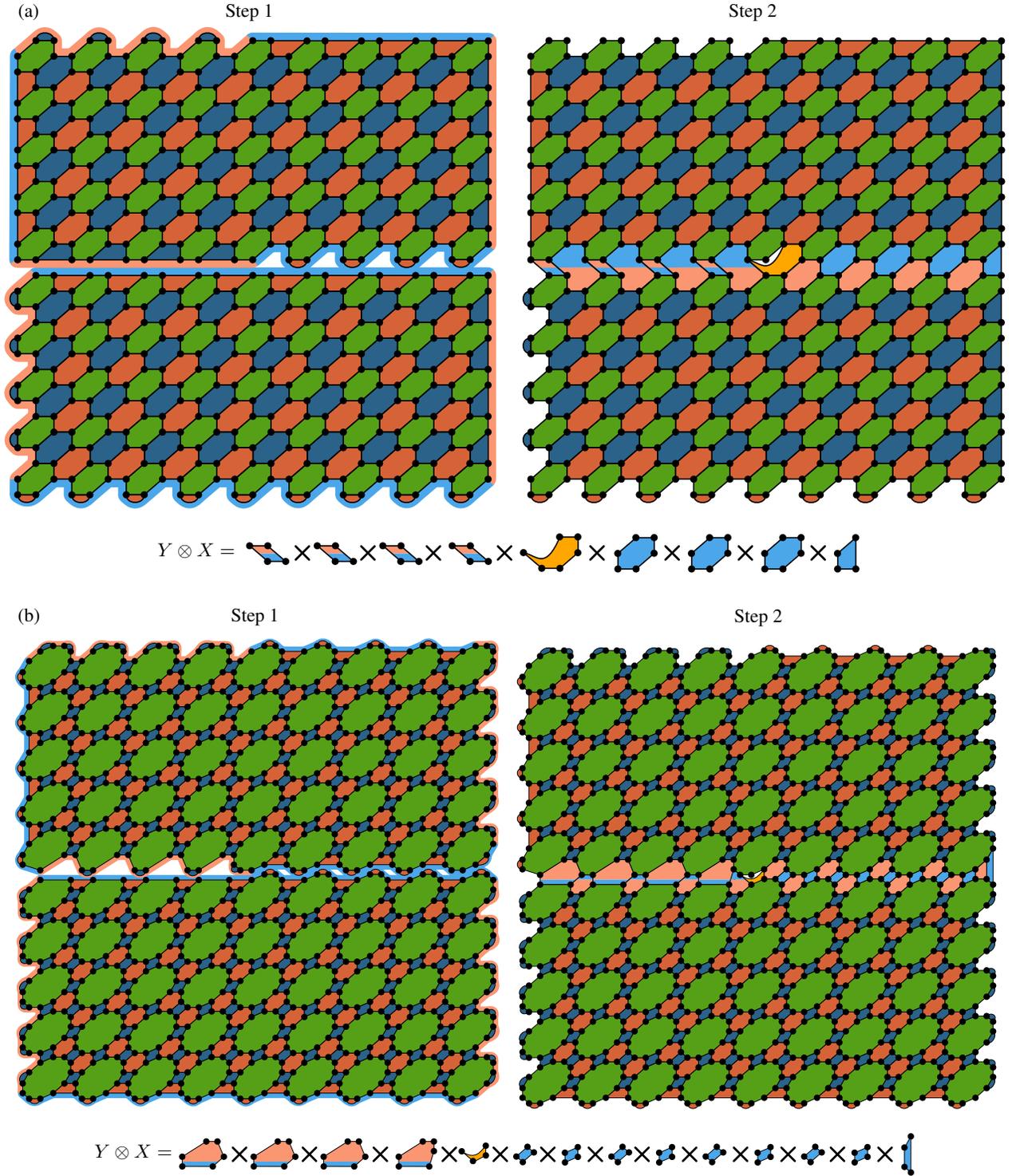


FIG. 11. Lattice surgery protocol for the measurement of  $Y \otimes X$  between 6.6.6 (a) and 4.6.12 (b) Majorana surface code hexons and tetrons. The maximum weight of the operators during lattice surgery is 8 Majoranas for 6.6.6 codes, and 6 Majorana for 4.6.12 codes.

While all of our Majorana surface code constructions can be applied to any Majorana platform, they can in particular be implemented using the quantum-wire constructions of Ref. [28]. We show nanowire implementation of the three uniform Majorana surface codes in Appendix B 2.

### C. State injection

So far, we have shown how to implement the operations Op1, Op2, and Op3 with 4.8.8, 6.6.6, and 4.6.12 Majorana surface codes, which allow tracking of Clifford gates, thus

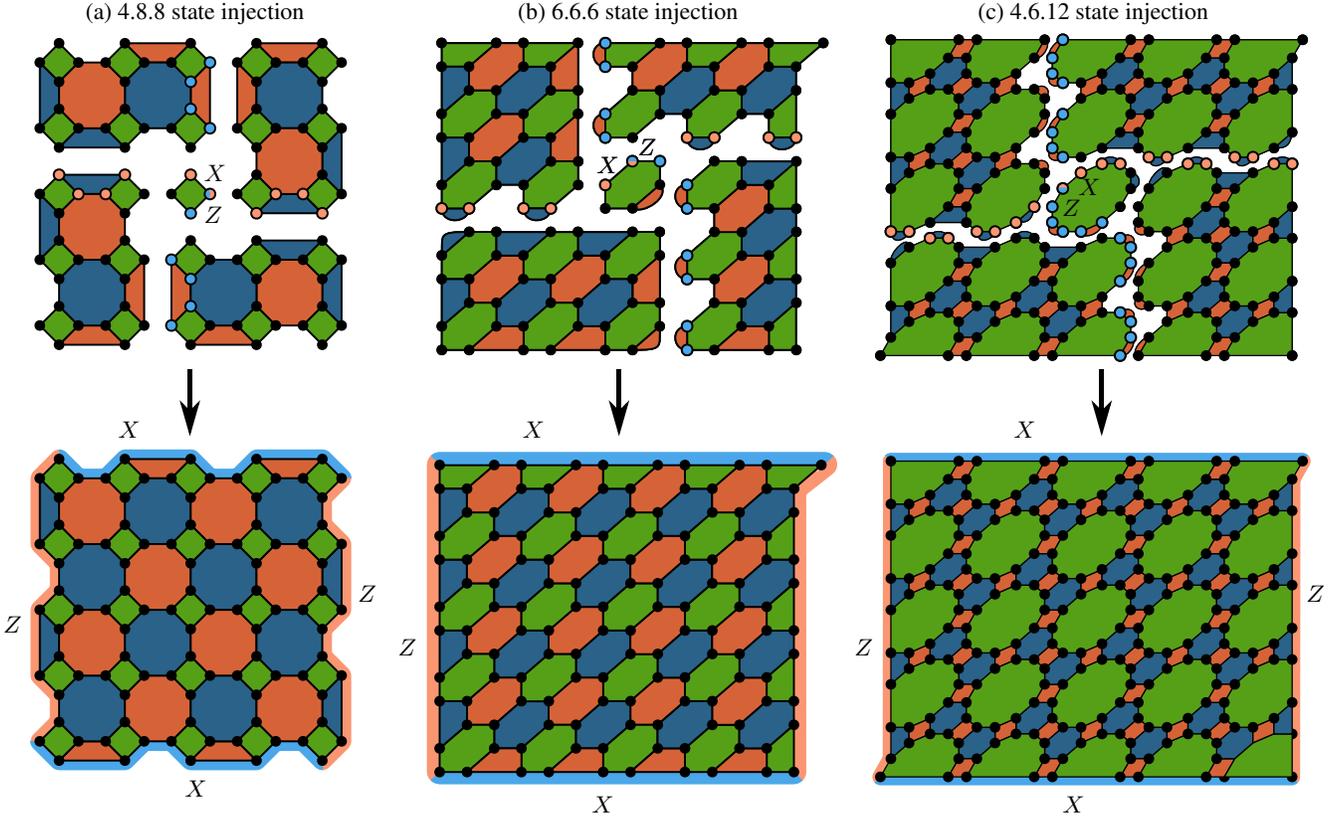


FIG. 12. State injection protocols to convert a state encoded in a physical tetron into a logical 4.8.8 (a), 6.6.6 (b), or 4.6.12 (c) tetron for the example of  $d = 5$ .

implementing them with zero time overhead. The remaining operation required for universality is the generation of (faulty) magic states that are encoded in tetrons. Unfortunately, even under the assumption of error-free  $d = 1$  tetrons, there is no robust way to initialize a magic state. One possibility to prepare a faulty magic state is to initialize a  $|+\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$  state by measuring  $i\gamma_2\gamma_3$ , and to split the degeneracy between the two states  $|0\rangle$  and  $|1\rangle$  by  $\Delta E$  for a time  $\tau = \pi/4 \cdot \hbar/\Delta E$ . The dynamical phase difference between the states  $|0\rangle$  and  $|1\rangle$  will yield a magic state  $|m\rangle = \frac{1}{\sqrt{2}}(|0\rangle + e^{i\pi/4}|1\rangle)$ . The degeneracy splitting can be implemented by coupling the two Majoranas  $\gamma_1$  and  $\gamma_2$ , as is done for the measurement of  $Z = i\gamma_1\gamma_2$ . This protocol is obviously not robust against perturbations, as it requires precise control of the coupling parameters to generate an exact phase difference of  $\pi/4$ . While there exist more sophisticated protocols for so-called Majorana magic gates [38] based on parity echoes, magic state distillation is still required to obtain a low-error magic state.

Majorana magic gates or the coupling of two Majoranas can be used to obtain a noisy magic state encoded in a ( $d = 1$ ) tetron. Fault-tolerant quantum computing requires us to convert this magic state into a magic state encoded in a *logical* tetron with a higher code distance. This is done via a protocol called state injection. There exist several protocols to inject an arbitrary state into a surface code [36, 39, 40]. In particular, Ref. [39] describes a two-step protocol that minimizes the

time that the magic state spends with code distance  $d = 1$ . Here, we adapt this protocol to fermionic codes.

Consider the stabilizer configuration shown in the top panel of Fig. 12a. The number of stabilizers is the same as in a 4.8.8 Majorana surface code with  $d = 5$ , i.e., exactly one qubit is encoded. Even though the four segments in the corners look like small surface codes, they do not encode any qubits, since they only have two different boundaries each and therefore only two corners. The encoded qubit is in fact the physical tetron in the center. Notice that the product of the Majoranas highlighted in red would correspond to a logical  $X$  operator in an actual  $d = 5$  code, since it is a string connecting the two red boundaries that commutes with all stabilizers. In the stabilizer configuration in the top panel, this string also commutes with all stabilizers. Since the product of the Majoranas that are not located at the center tetron is fixed to be  $+1$  by the stabilizers, the product of red Majoranas is equal to the  $X$  information encoded in the center tetron. Similarly, the product of blue Majoranas is equivalent to the  $Z$  information encoded in the center tetron, and would correspond to a blue-to-blue string of Majoranas in a  $d = 5$  code. By switching the stabilizer configuration from the top configuration to the bottom configuration, the state encoded in the physical tetron is converted into a logical tetron. As the highlighted logical operators commute with the stabilizers in both cases, they remain unchanged in the process. However, the measurement

outcomes of the new stabilizers straddling the corner segments are random. The errors need to be corrected in such a way that the correction operation commutes with the highlighted logical operators. Note that since we start out with a  $d = 1$  qubit, state injection is not fault-tolerant, which further emphasizes the need for magic state distillation.

The injection procedures for 6.6.6 and 4.6.12 Majorana surface codes are very similar. In the 6.6.6 case in Fig. 12b, the logical information is initially encoded in four of the six Majoranas of the center hexon. Again, the highlighted Majorana strings commute with both stabilizer configurations and correspond to the logical operators. For the 4.6.12 code in Fig. 12c, the logical information is initially encoded in four of the 12 Majoranas of the center dodecon.

Thus, we have implemented all four operations necessary for universal quantum computation with tetrons and hexons with minimal-overhead Clifford gates. The Pauli product measurement protocol of Fig. 7 can be implemented with *logical* tetrons and hexons exactly the same way as was described in Sec. III by replacing the measurements of four Majoranas by (twist-based) lattice surgery.

## V. MAJORANA COLOR CODES

In the previous section, we have shown that Majorana surface codes can be used to decrease the weight of the operators that need to be measured for fault-tolerant quantum computing. However, they come with a higher space overhead compared to bosonic surface codes. What about the opposite direction? How does one design topological codes with higher stabilizer weights, but, in turn, lower space overhead?

The answer lies in color codes. Bosonic color codes [23, 41] are known to feature higher stabilizer weights compared to bosonic surface codes, but they can encode qubits more compactly, using fewer physical qubits per logical qubit of a given code distance. Color codes are closely related to surface codes, as they can be obtained by concatenating surface codes with small non-topological codes [31]. In an alternative construction, color codes are obtained by folding surface codes in half [42]. Here, we adapt the concatenation scheme to Majorana surface codes, in order to obtain Majorana color codes.<sup>2</sup>

For bosonic codes, concatenation means that each physical qubit of a code is replaced by another logical qubit. For instance, a 4.8.8 bosonic color code can be obtained by concatenating a bosonic surface code with a  $[[4, 2, 2]]$  code [31]. This means that two bosonic surface codes are stacked on top of each other, and pairs of stacked qubits are replaced by  $[[4, 2, 2]]$  codes, as shown in Fig. 14a/b. Similarly, we propose that Majorana color codes can be obtained by concatenating Majorana surface codes with  $[[n_m, k, d_m]]_m$  codes, where  $[[n_m, k, d_m]]_m$  labels Majorana fermion codes that use  $n$  Majoranas to encode  $k$  qubits with Majorana distance  $d_m$ .

We suggest to concatenate fermionic codes by stacking Majorana surface codes on top of each other and replacing stacked *building blocks* of the code by  $[[n_m, k, d_m]]_m$  codes. The simplest example of a Majorana color code is a 4.8.8 Majorana surface code concatenated with a  $[[6, 2, 2]]_m$  code, which is shown in Fig. 13. The  $[[6, 2, 2]]_m$  code uses six Majoranas to encode two qubits with Majorana distance  $d_m = 2$ , which means that it is a hexon. In the concatenation procedure shown in Fig. 13 for a small segment of a 4.8.8 surface code, two Majorana surface codes are placed on top of each other. Next, each stack of two tetrons is replaced by a single hexon. Hexons encode twice as many qubits as tetrons, but use only 1.5 times as many Majoranas to do so. Thus, a logical tetron built from the Majorana color codes shown in Fig. 13 encodes two qubits as two layers of surface codes on top of each other, but uses only  $3d^2$  Majoranas per qubit with distance  $d$ , while the stabilizer weight remains the same. Note that even though we draw Majorana color codes as three-dimensional stacks, they can also be implemented in 2D architectures. In particular, the 4.8.8 ( $[[6, 2, 2]]_m$ ) Majorana color code can be implemented in a square lattice of hexons, but, in contrast to Majorana surface codes, the stabilizers now overlap.

More generally, we define Majorana color codes as any number of Majorana surface code layers obtained by concatenating Majorana surface codes with any other code. We refer to a code corresponding to  $n$  layers of surface codes as an order- $n$  code. Stacking  $k$  4.8.8 Majorana surface codes and replacing each stack of  $k$  tetrons with an  $[[n_m, k, d_m]]_m$  code yields an order- $k$  Majorana color code. For instance, the previously discussed 4.8.8 ( $[[6, 2, 2]]_m$ ) Majorana color code is an order-2 Majorana color code. Majorana surface codes are order-1 codes. One way to obtain an order-3 code is to place three 4.8.8 Majorana surface codes on top of each other, and replace each stack of three tetrons with an octon (an  $[[8, 3, 2]]_m$  code). Since one of the logical operators of the octon is a 4-Majorana operator, the maximum stabilizer weight increases to 10 Majoranas, as we discuss in Appendix C 1.

Bosonic color codes in their usual definition [23] are order-2 codes, as they correspond to two surface code layers.

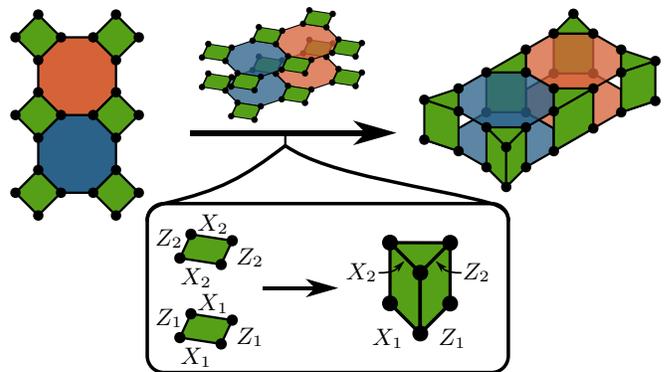


FIG. 13. Scheme to obtain a 4.8.8 ( $[[6, 2, 2]]_m$ ) Majorana color code by concatenating the 4.8.8 Majorana surface code with a  $[[6, 2, 2]]_m$  code, i.e., by stacking two surface codes and replacing each pair of tetrons with a hexon.

<sup>2</sup> Note that in Ref. [9], the term *Majorana color code* was used to refer to Majorana surface codes.

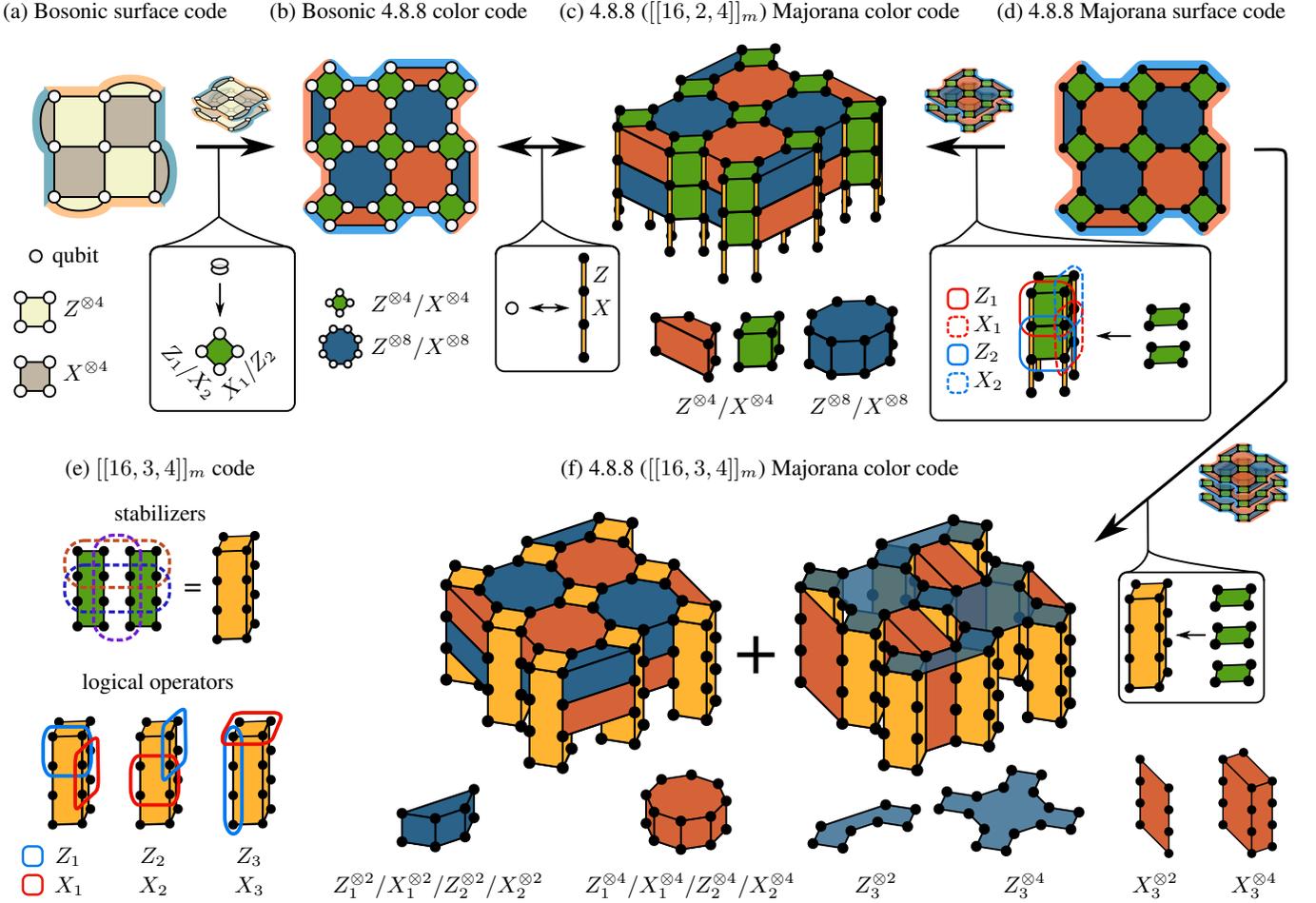


FIG. 14. A bosonic surface code (a) concatenated with a  $[[4, 2, 2]]$  code yields a bosonic 4.8.8 code (b). This can also be drawn as a Majorana fermion code (c) by replacing each qubit with a tetron (yellow lines). This Majorana color code can also be obtained by replacing each tetron of the 4.8.8 Majorana surface code with a  $[[16, 2, 4]]_m$  code. (e) Stabilizers and logical operators of the  $[[16, 3, 4]]_m$  code presented in Ref. [19]. (f) Order-3 Majorana color code obtained by concatenating a 4.8.8 Majorana surface code with the  $[[16, 3, 4]]_m$  code. Each yellow 16-Majorana box corresponds to a  $[[16, 3, 4]]_m$  code. Since the stabilizers overlap, they are shown in two separate figures.

One such code is the previously mentioned bosonic 4.8.8 color code, which is obtained by replacing each qubit of a bosonic surface code (Fig. 14a) with a  $[[4, 2, 2]]$  code. This is a code that uses four qubits and has two stabilizers  $Z \otimes Z \otimes Z \otimes Z$  and  $X \otimes X \otimes X \otimes X$ . It encodes two logical qubits with logical operators  $Z_1 = Z \otimes Z \otimes \mathbb{1} \otimes \mathbb{1}$ ,  $X_1 = \mathbb{1} \otimes X \otimes X \otimes \mathbb{1}$ , and  $Z_2 = \mathbb{1} \otimes Z \otimes Z \otimes \mathbb{1}$ ,  $X_2 = X \otimes X \otimes \mathbb{1} \otimes \mathbb{1}$ . Thus, a bosonic 4.8.8 color code tetron (see Fig. 14b) is equivalent to two surface codes on top of each other. It can be converted to a Majorana fermion code by replacing each qubit with a tetron. In Fig. 14c, these tetron building blocks are the yellow strings of four Majoranas. Another way of obtaining this Majorana fermion code is by stacking two 4.8.8 Majorana surface codes and replacing pairs of tetrons with  $[[16, 2, 4]]_m$  codes, which is the Majorana representation of the bosonic  $[[4, 2, 2]]$  code. The bosonic 4.8.8 color code uses four times as many physical qubits (or Majoranas) to encode twice as many logical qubits with twice the code distance. Thus, the Majorana overhead of bosonic 4.8.8 color codes is  $2d^2$  per qubit, which is half the overhead of bosonic surface codes. However, the stabilizer

weight doubles from 8 Majoranas to 16 Majoranas.

The  $[[16, 2, 4]]_m$  code used to obtain the bosonic 4.8.8 color code by concatenation is a bosonic code in the sense that its building blocks are tetrons. With Majoranas, there is also the possibility to use Majorana fermion codes for concatenation. These codes can encode information more compactly. In Ref. [19], a collection of small Majorana fermion codes is presented. The smallest  $d_m = 4$  Majorana fermion code is the  $[[16, 3, 4]]_m$  code shown in Fig. 14e, which uses 16 Majoranas to encode three instead of just two logical qubits with a Majorana distance of  $d_m = 4$ . It is based on two octons with three additional 8-Majorana operators as stabilizers. All of its logical operators are weight-4 Majorana operators. The order-3 Majorana color code obtained by concatenating the 4.8.8 surface code with the  $[[16, 3, 4]]_m$  code is shown in Fig. 14f. It essentially corresponds to three layers of Majorana surface codes and has a reduced Majorana overhead of  $\frac{4}{3}d^2$  with octons as building blocks, while still featuring a maximum stabilizer weight of 16 Majoranas.

In general, concatenating a bosonic surface code with an

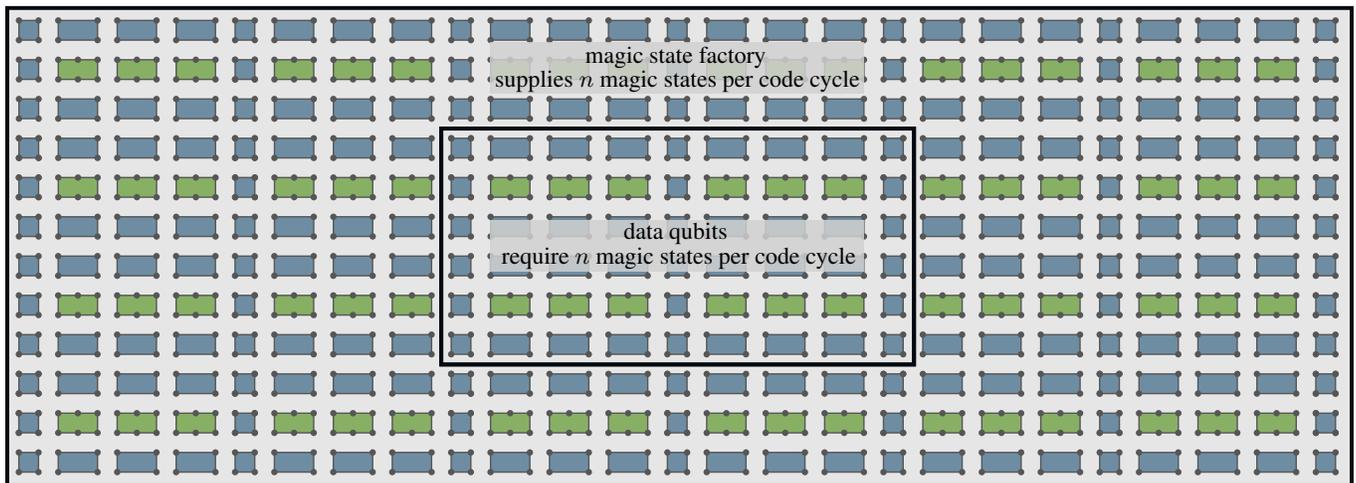


FIG. 15. Scheme for quantum computation in an array of tetrons and hexons, where the hexons are used to encode qubits, and tetrons are used as ancillas for Pauli product measurements. One part of the array is used to encode the data qubits for quantum computation, whereas the rest of the quantum computer is used to distill magic states.

$[[n, k, d_b]]$  bosonic code increases the number of physical qubits by a factor of  $n$ , the number of logical qubits by a factor of  $k$ , and the code distance by a factor of  $d_b$ . Thus, the number of physical qubits per logical qubit goes from  $d^2$  to  $\frac{n}{kd_b^2}d^2$ . The maximum stabilizer weight of the code may increase from 4 to  $4w_{\max}$ , where  $w_{\max}$  is the maximum weight of the logical operators of the  $[[n, k, d_b]]$  code. Similarly, a 4.8.8 Majorana surface code concatenated with an  $[[n_m, k, d_m]]_m$  code decreases the Majorana overhead from  $4d^2$  to  $\frac{4n_m}{kd_m^2}d^2$  Majoranas for a logical qubit with code distance  $d$ .

The second smallest  $d_m = 4$  Majorana fermion code is the  $[[20, 4, 4]]_m$  code. Its stabilizers and logical operators are shown in Appendix C 2. The code is based on two decons (10 Majoranas in a fixed parity sector) and four 10-Majorana stabilizers, and encodes four qubits whose logical operators all have a weight of 4 Majoranas. Thus, the order-4 Majorana color code obtained by concatenating a 4.8.8 surface code with a  $[[20, 4, 4]]_m$  has a Majorana overhead of  $1.25d^2$  while still only featuring a maximum stabilizer weight of 16 Majoranas. Since some of the logical operators of the third-smallest  $d_m = 4$  code, the  $[[24, 6, 4]]_m$  code, have a weight of 6 Majoranas, the stabilizer weight of a Majorana code obtained by concatenating this code with a 4.8.8 Majorana surface code would exceed 16 Majoranas. In general, Majorana surface codes can be concatenated with arbitrary Majorana fermion codes to obtain codes with a smaller Majorana overhead, while increasingly sacrificing stabilizer weight and locality of the code.

Majorana color codes can be used for quantum computation the same way as Majorana surface codes, i.e., by encoding tetrons and hexons, and performing lattice surgery. In fact, the stabilizer weight and space overhead of the lattice surgery protocol can be reduced by using surface-to-color code lattice surgery [15], such that the ancilla tetron is not a color code, but a surface code. For the previously discussed codes with 16-Majorana stabilizers, surface-to-color code lattice surgery

has a maximum stabilizer weight of 12 Majoranas. We show an example of this in Appendix C 3.

In summary, we have described Majorana color codes, which are multiple layers of Majorana surface codes obtained by concatenation. The advantage of Majorana color codes compared to bosonic color codes is that they can encode logical qubits more compactly with the same stabilizer weight. In particular, the Majorana color code obtained by concatenating the 4.8.8 Majorana surface code with the  $[[20, 4, 4]]_m$  code has a Majorana overhead of  $1.25d^2$ , compared to the Majorana overhead of  $2d^2$  of the bosonic 4.8.8 color code.

## VI. CONCLUSION

We have described a unified framework for fault-tolerant quantum computation with Majorana-based qubits. A full quantum computer could look like the array of tetrons and hexons shown in Fig. 15. Hexons are used to encode qubits, while tetrons are ancillas that are used during the Pauli product measurement protocol. One part of the quantum computer consists of the data qubits that are used for quantum computation, which requires a certain number of magic states  $n$  per code cycle as a resource for  $T$  gates, where  $n$  depends on the given quantum computation. In order for magic state distillation not to be a bottleneck of the quantum computer, the resources devoted to magic state distillation need to be large enough to supply  $n$  magic states per code cycle. The main operation in addition to the preparation of faulty magic states is the measurement of Pauli product operators according to the protocol in Fig. 7. Taking full advantage of the Gottesman-Knill theorem, this enables the classical tracking of all Clifford gates, which means that these gates require zero time overhead.

In principle, the scheme with  $d = 1$  tetrons and hexons shown in Fig. 15 can be implemented in a nanowire array in

the spirit of Ref. [28], as shown in Appendix B 1. However, this only allows for quantum computation on time scales of the order of the coherence times of the physical tetrons and hexons. For fault-tolerant quantum computation, each tetron and hexon needs to be replaced by a *logical* tetron and hexon with an appropriate code distance, and 4-Majorana parity measurements need to be replaced by lattice surgery.

There is a wide variety of codes that one can choose for this purpose, a collection of which is shown in Tab. I. Surface codes only require the measurement of low-weight stabilizers, but feature a high space overhead. Majorana surface codes feature lower-weight stabilizers compared to bosonic surface codes. In particular, 4.8.8 Majorana surface codes have the same bulk stabilizer weights as bosonic surface codes, but a lower stabilizer weight for lattice surgery of 8 Majoranas compared to 10 Majoranas. 6.6.6 and 4.6.12 Majorana surface codes reduce the bulk stabilizer weights to an average of 6 and 5 Majoranas, respectively, but at the same time increase the Majorana overhead by a factor of 1.5 and 3, respectively.

Color codes can reduce the space overhead at the price of higher-weight stabilizers. Majorana color codes encode qubits more compactly compared to bosonic color codes. The 4.8.8  $([[6, 2, 2]]_m)$  Majorana color code has the same stabilizer weights as a bosonic surface code, but features a lower Majorana overhead. Similarly, the 4.8.8  $([[20, 4, 4]]_m)$  Majorana color code has the same stabilizer weights as the bosonic 4.8.8 color code, but a lower Majorana overhead by a factor of 5/8.

Our entire scheme can also be realized with non-topological (e.g., superconducting) qubits, but is then limited to the use of bosonic codes. From the perspective of Majorana fermion codes, non-topological qubits can only implement logical tetrons and hexons with physical tetrons as building blocks, whereas Majorana-based qubits can also implement codes with other building blocks such as physical hexons, octons, decons, or dodecons.

There are many questions that remain unanswered. Even though our protocols allow the classical tracking of CNOT gates, it remains uncertain whether tracking is always advantageous compared to actually executing these gates. After all, if a layer of Clifford gates is followed by a layer of  $T$  gates, all of these  $T$  gates can be executed simultaneously. However, if CNOTs are tracked, these  $T$  gates require spatially overlapping Pauli product measurements. While our measurement protocol shows how arbitrary Pauli products can be measured, it is unclear how multiple (commuting) Pauli products can be read out simultaneously. If it turns out that it is more advantageous to execute CNOT gates explicitly, our Pauli product measurement protocol can be straightforwardly used for multi-target CNOT gates using the protocol in Appendix A 3.

Moreover, while bosonic surface-code decoders can be used to decode Majorana surface codes, one could also devise decoding schemes that are tailored towards Majorana fermion codes. In particular, in a setting where parity-fixing constraints can be violated and stabilizers of all colors are measured, a decoder needs to be able to match three types of anyons, as opposed to just two. Furthermore, one could adapt the scheme of Majorana-based fermionic quantum computa-

tion [43] to the fault-tolerant setting by replacing arrays of Majoranas with arrays of twist defects in Majorana surface codes. Based on twists in Majorana surface codes, it would also be interesting to see whether it is possible to come up with a Majorana version of the twist-based triangle code presented in Ref. [44]. Also, we did not investigate the performance of Majorana fermion codes, i.e., the logical error rate under the assumption of a realistic error model. We hope that the framework presented in this work will prove useful for Majorana-based quantum computation.

## ACKNOWLEDGMENTS

The authors would like to thank Markus Kesselring, Jens Eisert, Theodore Yoder, Michael Beverland, Benjamin Brown, and Aleksander Kubica for insightful discussions. This work has been supported by the Deutsche Forschungsgemeinschaft (Bonn) within the network CRC TR 183.

## Appendix A: Majorana surface codes

**1. 6.6.6 and 4.6.12 Majorana surface codes in square lattices of hexons and dodecons.** Some of the logical surface code tetrons and hexons in Figs. 3 and 4 feature smaller building blocks (green stabilizers) at the boundary compared to the building blocks in the bulk. In particular, some of the green stabilizers of 6.6.6 surface codes are tetrons at the boundary, whereas they are hexons in the bulk. Similarly, 4.6.12 surface codes feature hexons, octons, and decons at the boundary, as opposed to dodecons. Still, all of these codes can be implemented on a square lattice of building blocks (green stabilizers), where each building block is the same. The equivalence in Fig. 16 shows that 6.6.6 Majorana surface codes can be implemented on a square lattice of hexons, and 4.6.12 codes can be implemented on a square lattice of dodecons. The two-qubit plaquettes at the boundary fix the parity of some of the Majoranas of the hexons and dodecons, such that they effectively become tetrons, hexons, octons or decons.

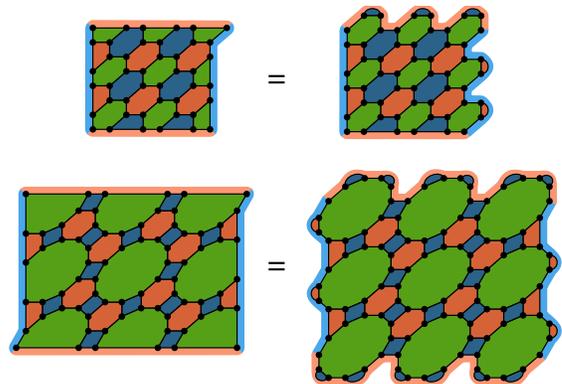


FIG. 16. 6.6.6 and 4.6.12 surface code tetrons in a square lattice of hexons and dodecons, respectively.

**2. Mapping Majorana surface codes onto bosonic surface codes.** While every bosonic code can be uniquely mapped onto a Majorana fermion code by replacing each qubit with a tetron, the reverse mapping is not unique. For Majorana surface codes, one prescription to map them onto a bosonic code is to interpret each stabilizer of a certain color of the three-colorable tiling as a number of qubits. For instance, for the 4.8.8 tiling, one has the choice of either interpreting the 8-Majorana stabilizers of one color as three qubits encoded in an octon, or of interpreting all 4-Majorana stabilizers as tetrons. The latter mapping precisely yields the bosonic surface code on a square lattice, as shown in Fig. 17a. Note that in this mapping, neighboring tetrons have different orientations.

For the 6.6.6 Majorana surface code in Fig. 17b, we interpret each green stabilizer as a hexon, where one qubit is encoded in the bottom left three Majoranas, and the other qubit in the top right three Majoranas. Thus, we replace each green hexon with two qubits, and obtain a bosonic surface code on a rotated square lattice. For the 4.6.12 Majorana surface code there are again several possibilities. In Fig. 17c, we show a mapping where each 4-Majorana stabilizer is replaced by a

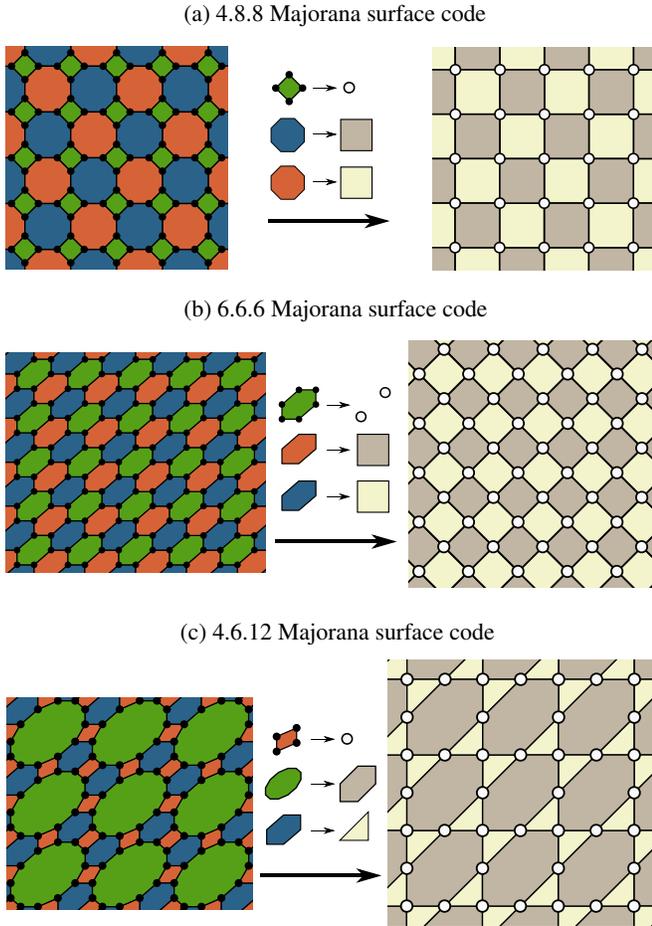


FIG. 17. One choice of mapping of 4.8.8, 6.6.6, and 4.6.12 Majorana surface codes onto bosonic surface codes. For bosonic surface codes, the dark plaquettes are products of  $X$  operators, and the light plaquettes are products of  $Z$  operators.

tetron, which yields a bosonic surface code on a Kagome lattice.

**3. State injection and CNOT gates with Pauli product measurements.** In a quantum computing architecture where all Clifford gates are tracked and only Pauli product measurements are explicitly performed, the state injection circuit from Fig. 5a is replaced by Fig. 18. In particular, the tracked CNOT gate maps the  $Z$  measurement of the magic state onto a  $Z \otimes Z$  measurement of  $|\psi\rangle \otimes |m\rangle$ . However, this leaves the qubits  $|\psi\rangle$  and  $|m\rangle$  in an entangled state, which means that  $|m\rangle$  cannot be discarded right away after the state injection. With  $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$  and  $|m\rangle = \frac{1}{\sqrt{2}}(|0\rangle + e^{i\pi/4}|1\rangle)$ , the initial state before any measurement is

$$|\Psi\rangle = \frac{1}{\sqrt{2}} \left( \alpha|00\rangle + \alpha e^{i\pi/4}|01\rangle + \beta|10\rangle + \beta e^{i\pi/4}|11\rangle \right). \quad (\text{A1})$$

After the  $Z \otimes Z$  measurement, there are two possible outcomes

$$|\Psi_0\rangle = \alpha|00\rangle + \beta e^{i\pi/4}|11\rangle \quad (\text{A2})$$

for outcome  $m_1 = 0$ , and

$$|\Psi_1\rangle = \alpha e^{i\pi/4}|01\rangle + \beta|10\rangle \approx \alpha|01\rangle + \beta e^{-i\pi/4}|10\rangle \quad (\text{A3})$$

for outcome  $m_1 = 1$ , where “ $\approx$ ” is an equality up to a global phase. After an  $S^{m_1}$  correction,  $|\Psi_0\rangle$  remains unchanged, whereas  $|\Psi_1\rangle$  is transformed to

$$S|\Psi_1\rangle = \alpha|01\rangle + \beta e^{i\pi/4}|10\rangle. \quad (\text{A4})$$

Both  $|\Psi_0\rangle$  and  $S|\Psi_1\rangle$  are entangled states. If one wants to discard the second qubit in order to continue to use it for magic state distillation, it first needs to be disentangled from the rest of the system. This can be done by measuring the magic state in the  $X$  basis with outcome  $m_2$ , followed by a  $Z^{m_2}$  Pauli correction on the qubit  $|\psi\rangle$ . For  $|\Psi_0\rangle$ , an  $X$  measurement leaves the state in

$$|\Psi_{0,0}\rangle = \left( \alpha|0\rangle + \beta e^{i\pi/4}|1\rangle \right) \otimes |+\rangle \quad (\text{A5})$$

for outcome  $m_2 = 0$ , and

$$|\Psi_{0,1}\rangle = \left( \alpha|0\rangle - \beta e^{i\pi/4}|1\rangle \right) \otimes |-\rangle \quad (\text{A6})$$

for outcome  $m_2 = 1$ . A  $Z^{m_2}$  correction leaves  $|\Psi_{0,0}\rangle$  unchanged and maps  $|\Psi_{0,1}\rangle$  to

$$Z|\Psi_{0,1}\rangle = \left( \alpha|0\rangle + \beta e^{i\pi/4}|1\rangle \right) \otimes |-\rangle. \quad (\text{A7})$$

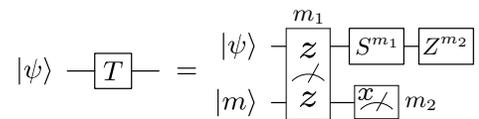


FIG. 18. A  $T$  gate on a qubit  $|\psi\rangle$  is equivalent to a  $Z \otimes Z$  measurement between  $|\psi\rangle$  and a magic state  $|m\rangle$  with outcome  $m_1$ , which results in a corrective  $S^{m_1}$  operation on  $|\psi\rangle$ . In order to disentangle  $|m\rangle$  from the qubit, it is measured in the  $X$  basis, prompting a  $Z^{m_2}$  Pauli correction on  $|\psi\rangle$ .



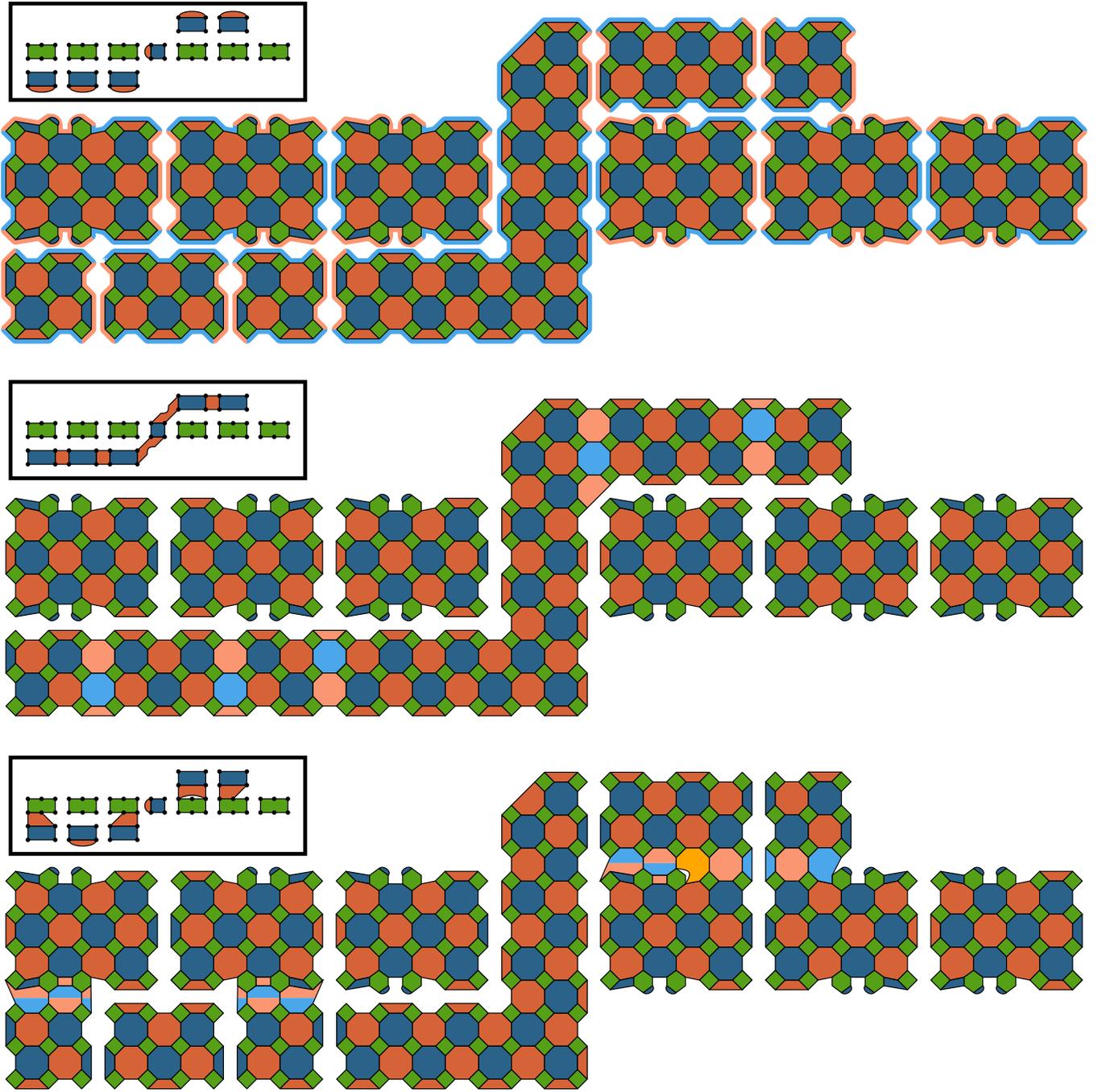


FIG. 21. Pauli product measurement protocol from Fig. 7 realized with  $d = 3$  4.8.8 Majorana surface codes. Only steps 1, 2, and 3 are shown.

line between the  $X$  boundary of the tetron and the  $Z$  boundary of the hexon no longer features any 8-Majorana operators, but only 6-Majorana operators. This reduces the maximum Majorana weight of quantum computing with 6.6.6 Majorana surface codes to 6 Majoranas.

**7. Twist-based lattice surgery with subsystem surface codes.** In Fig. 25, we show the twist-based lattice surgery protocol [22] for a  $d = 3$  subsystem surface code [25]. While all the check operators in the bulk of the code are weight-3, the measurement of the twist defect requires a 5-qubit mea-

surement. By replacing each qubit of the subsystem surface code with a tetron, one can see that the subsystem surface code resembles a 4.6.12 Majorana surface code, as is shown in Fig. 25.

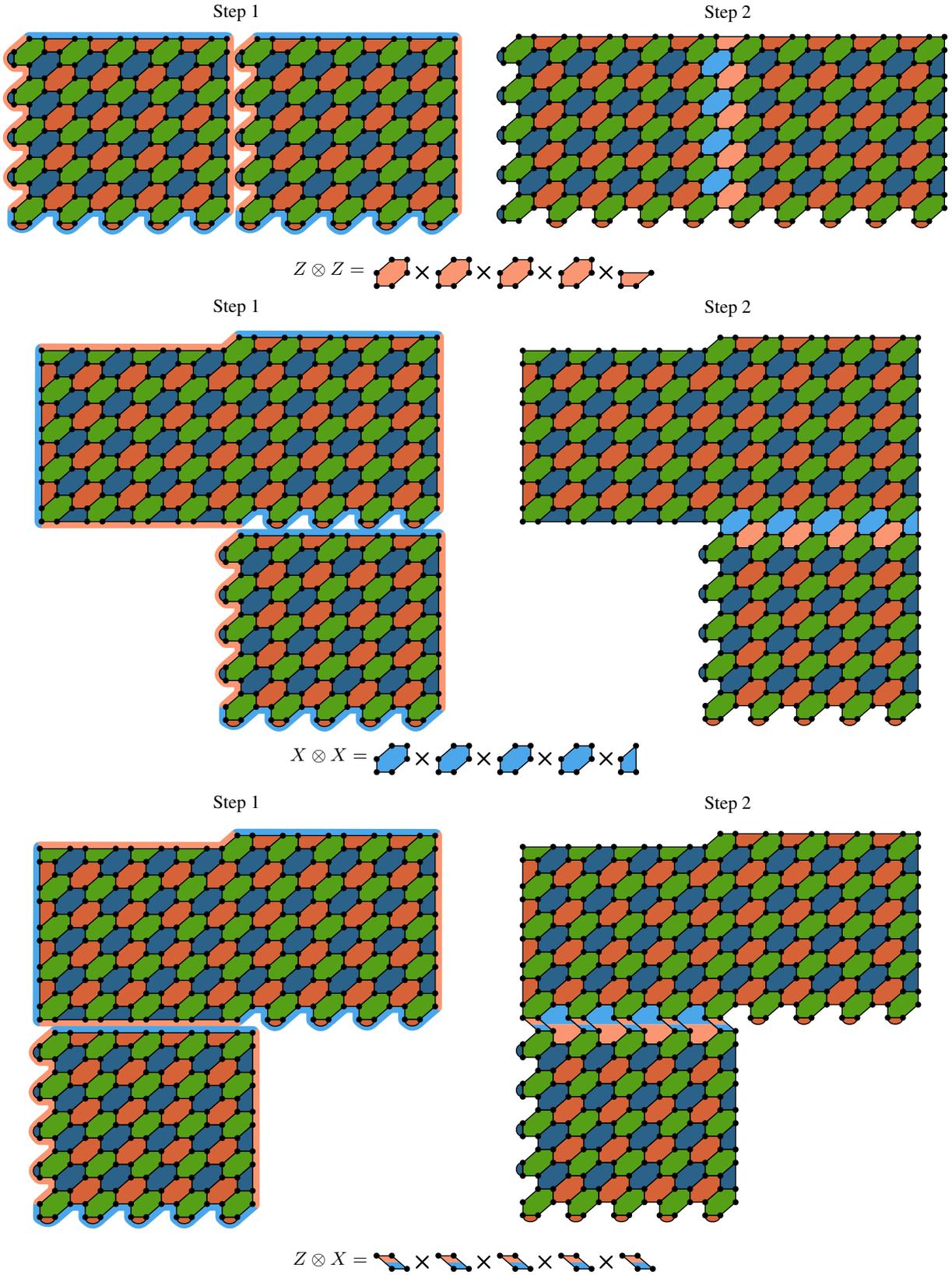


FIG. 22. Stabilizer configuration for  $Z \otimes Z$  measurements between two 6.6.6 Majorana surface code tetrons, and for  $X \otimes X$  and  $Z \otimes X$  measurements between a 6.6.6 hexon and a 6.6.6 tetron.

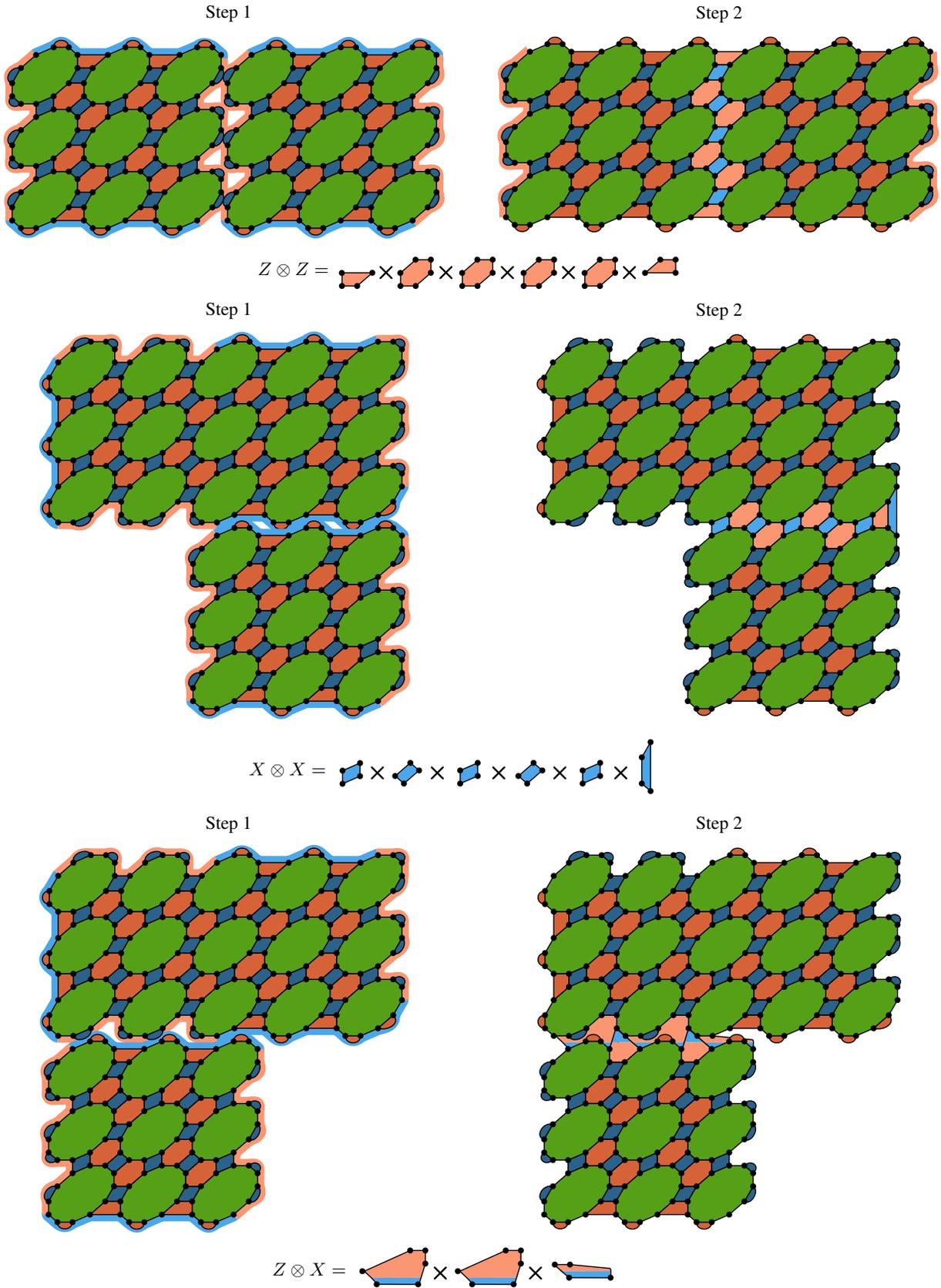


FIG. 23. Stabilizer configuration for  $Z \otimes Z$  measurements between two 4.6.12 Majorana surface code tetrons, and for  $X \otimes X$  and  $Z \otimes X$  measurements between a 4.6.12 hexon and a 4.6.12 tetron.

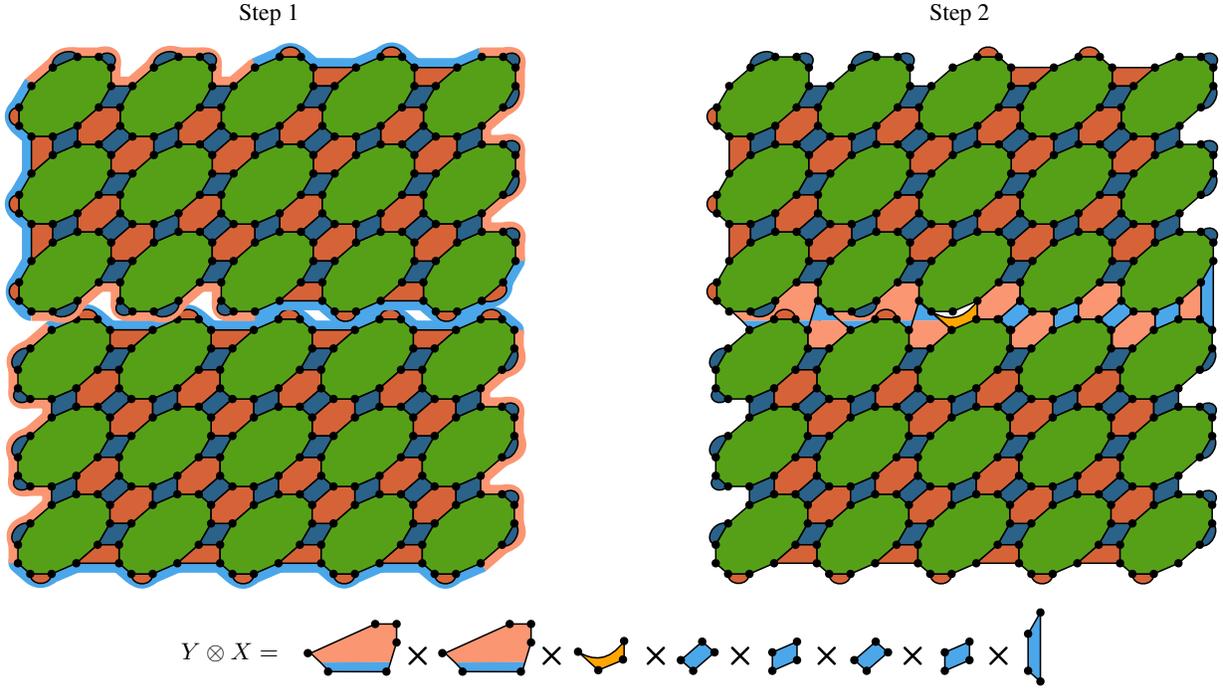


FIG. 24. Stabilizer configuration for  $Y \otimes Z$  measurements between a 4.6.12 Majorana hexon and tetron.

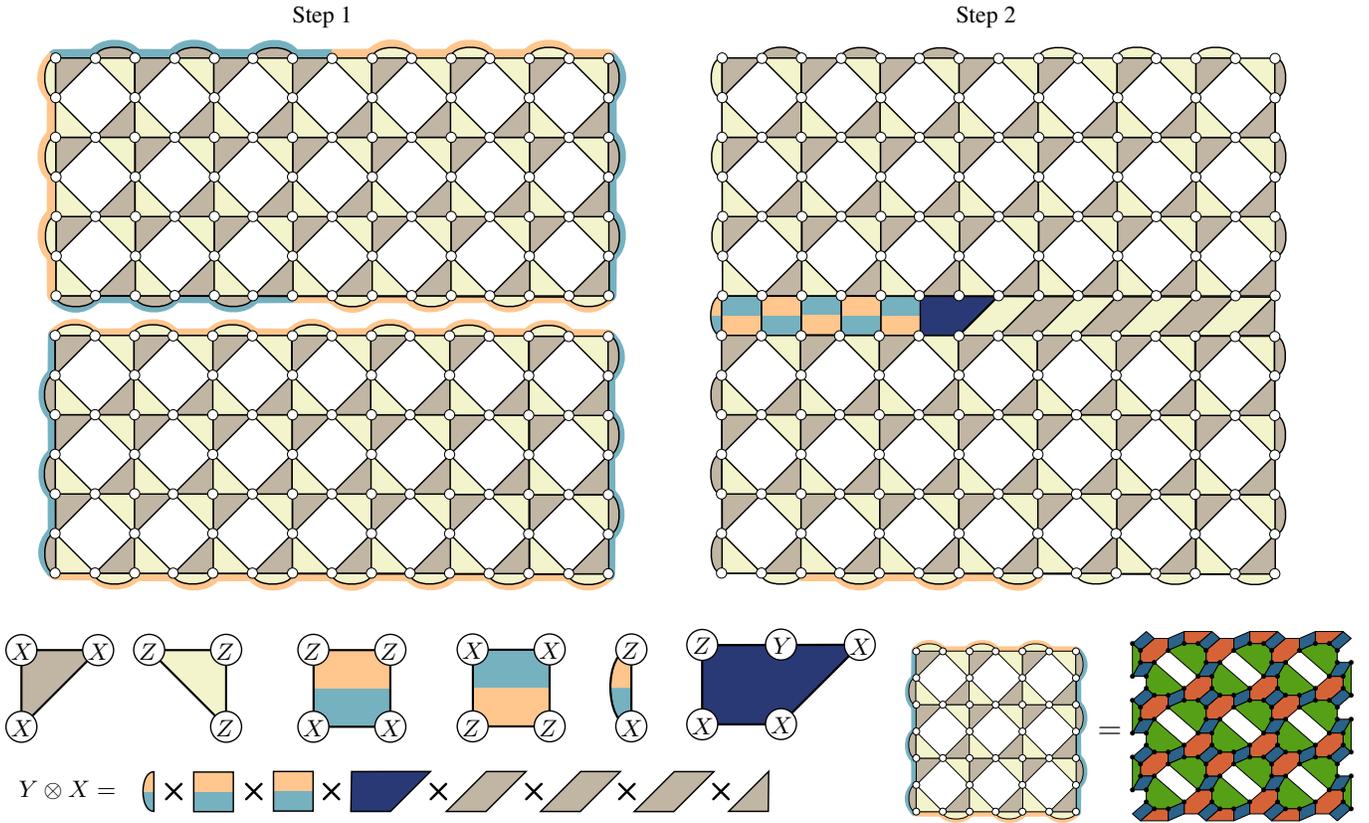


FIG. 25. Twist-based lattice surgery between a logical hexon and a logical tetron encoded in a  $d = 3$  bosonic subsystem surface code in order to measure  $Y \otimes Z$ . The Majorana fermion code obtained by replacing each qubit of a bosonic subsystem surface code tetron with four Majoranas in a fixed parity sector resembles a 4.6.12 Majorana surface code.

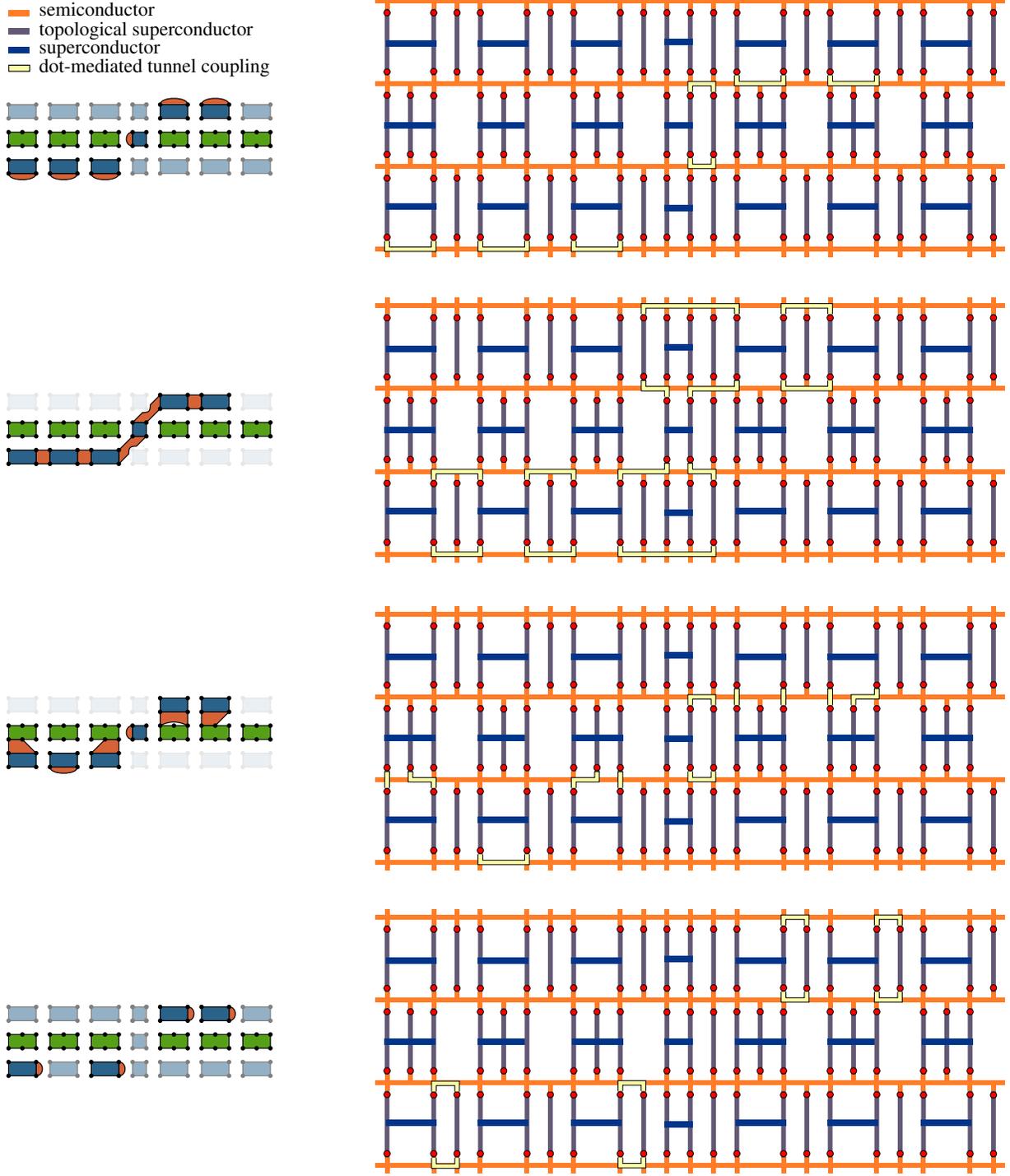


FIG. 26. Tunnel coupling configurations for the Pauli product measurements protocol from Fig. 7 realized with the wire-based tetrons and hexons proposed in Ref. [28].

## Appendix B: Quantum-wire-based implementations

**1. Pauli product measurement protocol in a nanowire array.** The Pauli product measurement protocol in Fig. 7 can be straightforwardly used to implement minimal-overhead Clifford gates in the wire-based architectures proposed in

Ref. [28]. Essentially, hexons can be implemented as three topological superconducting nanowires in a parity sector that is fixed by a non-topological superconductor bridging the three wires. Moreover, in Fig. 26, tetrons correspond to two topological superconducting nanowires in a fixed parity sector with an additional topological nanowire serving as a co-

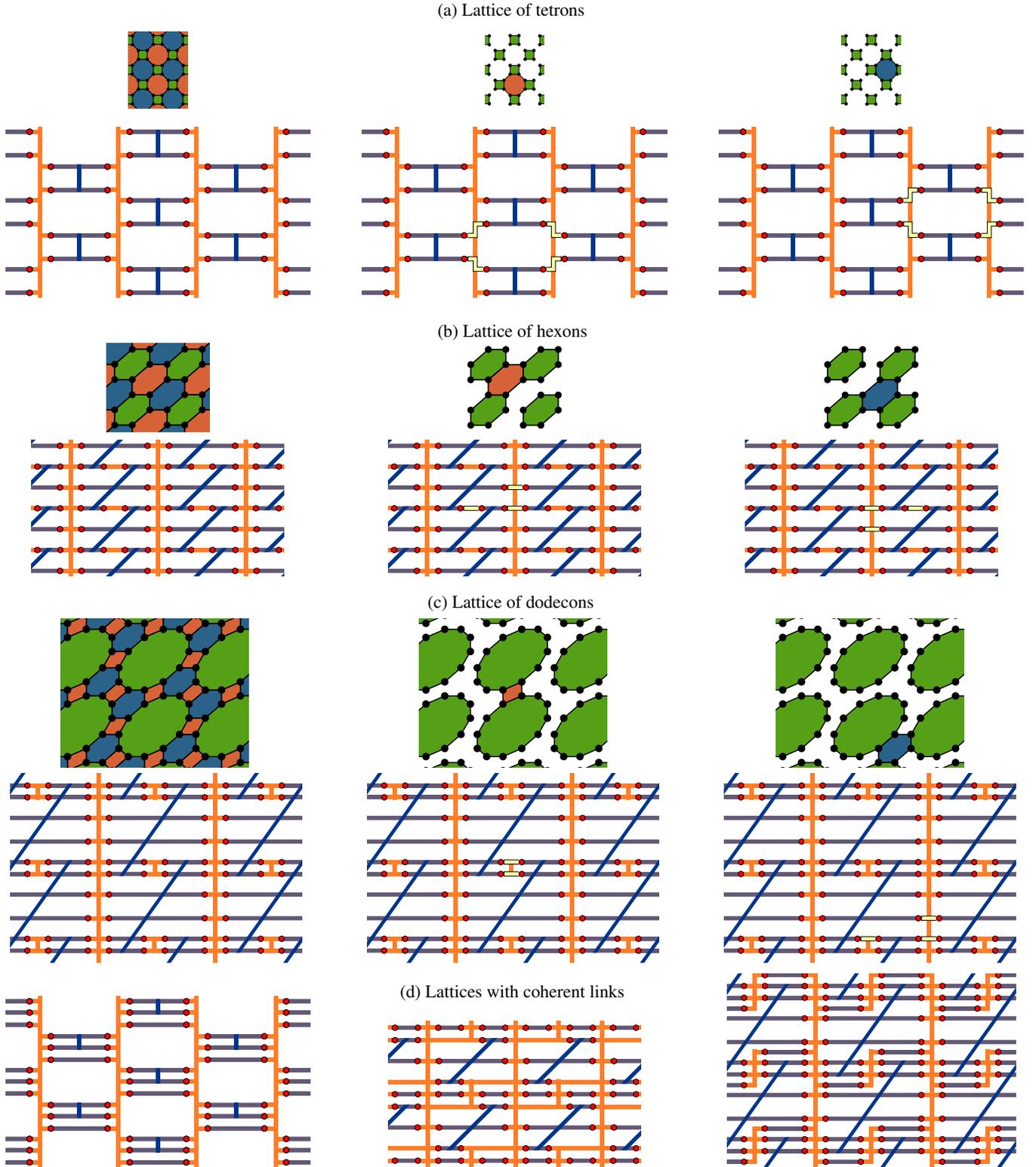


FIG. 27. Nanowire-based implementations of 4.8.8 (a), 6.6.6 (b), and 4.6.12 (c), with examples of tunnel-coupling configurations for red and blue stabilizer measurements. Coherent links (d) may be necessary for the measurement of other operators.

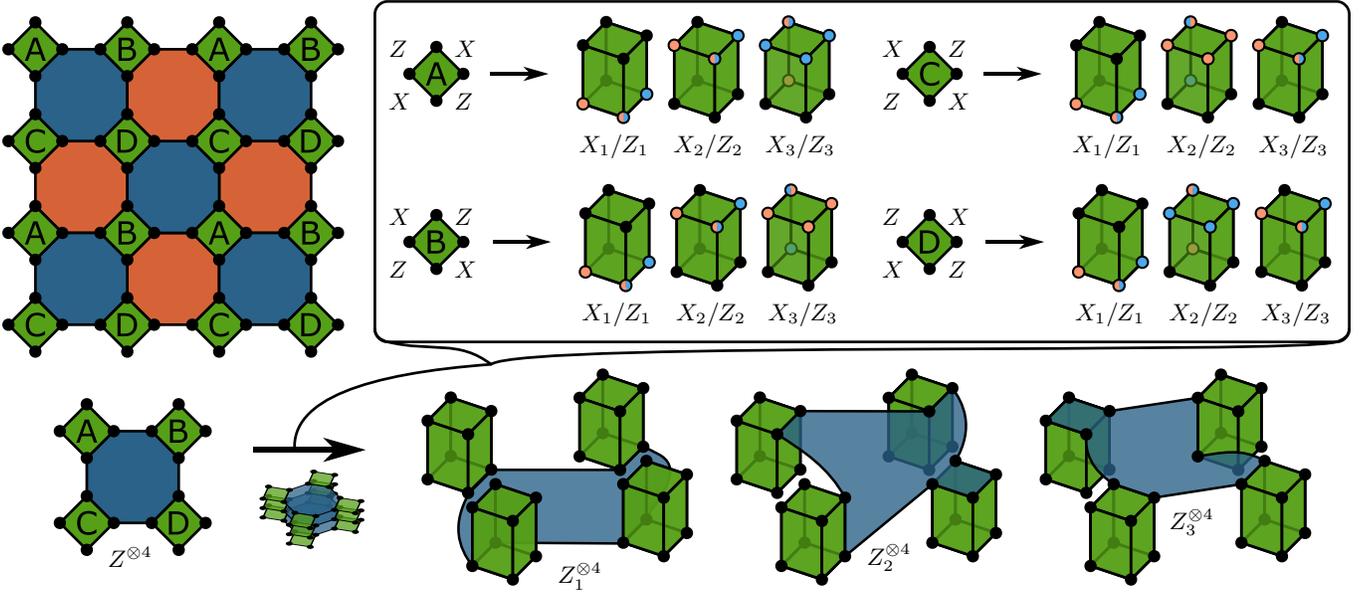


FIG. 28. Procedure to obtain a 4.8.8 ( $[[[8, 3, 2]]_m$ ) Majorana color code. In order to prevent stabilizer weights from increasing beyond 10 Majoranas, the tetrons of a 4.8.8 Majorana surface code are periodically labelled A, B, C, and D. Each tetron is replaced by a single octon, but the definition of the octon’s logical operators depends on the label of the corresponding tetron. In the figure, the octon’s  $Z$  ( $X$ ) operators correspond to products of blue (red) Majoranas. The resulting stabilizers have a maximum weight of 10 Majoranas, as shown for the three overlapping blue  $Z$  type stabilizers.

herent link that is used for certain parity measurements. Products of Majorana operators are measured by opening tunnel couplings between topological superconductors and segments of a semiconducting nanowire network, such that the tunnel couplings form closed paths. These semiconducting nanowire segments can be interpreted as quantum dots whose energy levels are shifted by the virtual process of an electron tunneling around the path, picking up all Majorana operators along the way. Therefore, suitable spectroscopy on the dots can be used to measure the product of these Majorana operators, as detailed in Ref. [28]. In Fig. 26, we show the tunnel coupling configurations that implement the measurements that are part of the Pauli product measurement protocol. Note that some of the measurements require the use of coherent links, such that the Majorana weight of some measurements is increased. In particular, some of the four-Majorana measurements in the second step become six-Majorana measurements in this particular implementation.

**2. Majorana surface code implementations in a nanowire array.** In Fig. 27, we show wire-based implementations of square lattices of tetrons (a), hexons (b), and dodecons (c), which can be used to implement 4.8.8, 6.6.6, and 4.6.12 Majorana surface codes, respectively. While we show the tunnel-coupling configurations to measure red and blue stabilizers, some irregular stabilizers – such as boundary stabilizers, dislocation lines, twist defect, or two-Majorana operators for initialization and readout – may require the use of coherent links. Nanowire arrays that include coherent links are shown in Fig. 27d. Note that operator measurements that require the use of coherent links have an increased Majorana weight compared to the optimal weight discussed in the main

text. We also remark that, in principle, all stabilizers of one color can be measured simultaneously.

### Appendix C: Majorana color codes

**1. 4.8.8 ( $[[[8, 3, 2]]_m$ ) Majorana color code.** A 4.8.8 ( $[[[8, 3, 2]]_m$ ) Majorana color code can be obtained by replacing each tetron of a 4.8.8 Majorana surface code with an  $[[8, 3, 2]]_m$  code, i.e., an octon. Even though this code has a Majorana distance of  $d_m = 2$ , one of its logical operators is a 4-Majorana operator. In order to prevent the maximum stabilizer weight after concatenation from increasing to 16 Majoranas, the procedure shown in Fig. 28 can be used. Here, each surface code tetron is assigned a label A, B, C, or D. While each tetron is still replaced by an octon, the definition of the octon’s logical operators depends on the label. For A, B, C, or D type octons, the 4-Majorana logical operator is chosen to be  $Z_3$ ,  $X_3$ ,  $X_2$ , or  $Z_2$ , respectively. This guarantees that the stabilizers have a maximum weight of 10 Majoranas after concatenation. The figure shows an example of a  $Z^{\otimes 4}$  stabilizer, which is replaced by three overlapping  $Z_1^{\otimes 4}$ ,  $Z_2^{\otimes 4}$ , and  $Z_3^{\otimes 4}$  stabilizers after concatenation. The operators  $Z_2^{\otimes 4}$  and  $Z_3^{\otimes 4}$  have a weight of 10 Majoranas.

**2. Stabilizers and logical operators of the  $[[20, 4, 4]]_m$  code.** Figure 30 shows the stabilizers and logical operators of the  $[[20, 4, 4]]_m$  code presented in Ref. [19]. The building blocks of the code are two decons, which are 10 Majoranas in a fixed parity sector. All logical operators have a weight of 4 Majoranas. This means that a code obtained by concatenating a Majorana surface code with the  $[[20, 4, 4]]_m$  code is

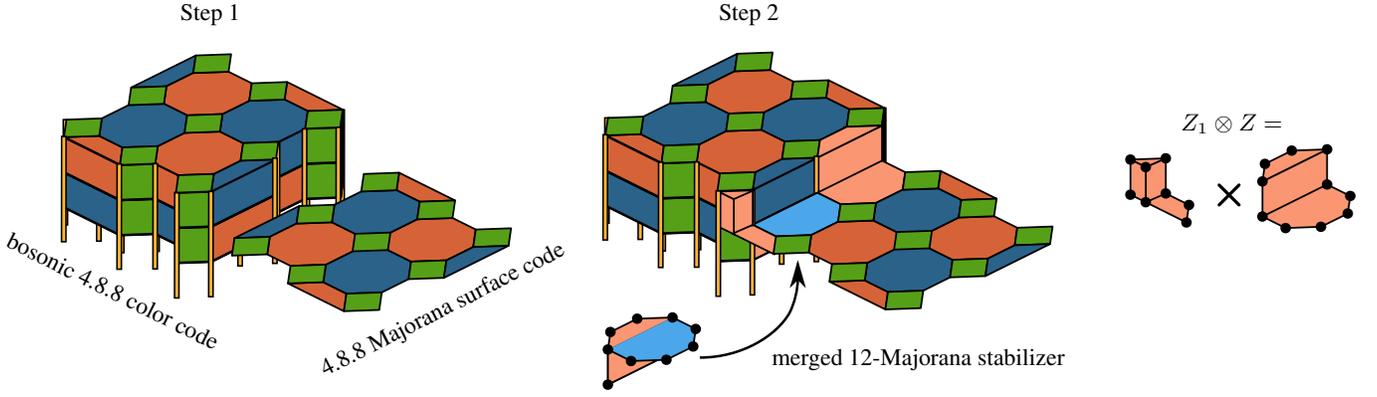


FIG. 29. Lattice surgery protocol for the measurement of the operator  $Z_1 \otimes Z$  between a bosonic 4.8.8 color code tetron and a 4.8.8 Majorana surface code tetron.  $Z_1$  is the  $Z$  operator of the first qubit encoded in the bosonic 4.8.8 color code tetron.

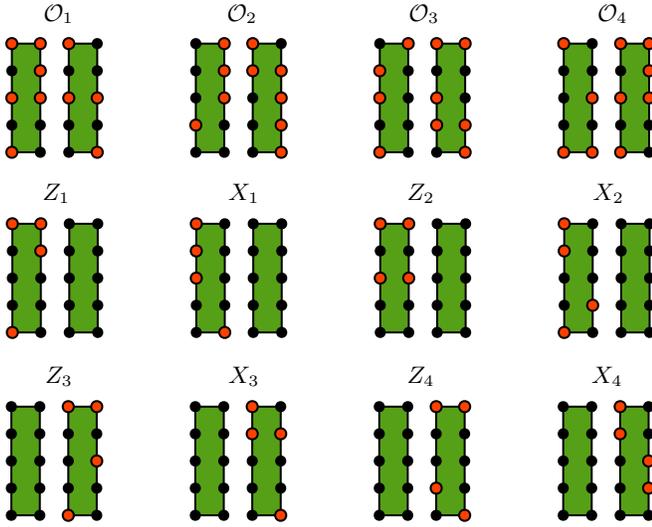


FIG. 30. Stabilizers  $\mathcal{O}_1$ - $\mathcal{O}_4$  and logical operators of the  $[[20, 4, 4]]_m$  code presented in Ref. [19].

an order-4 Majorana color code with a maximum stabilizer weight of 16 Majoranas.

**3. Surface-to-color code lattice surgery.** Figure 29 shows the an example of a surface-to-color code lattice surgery in the spirit of Ref. [15]. The example shows the measurement of the product  $Z_1 \otimes Z$  between the first qubit encoded in a bosonic 4.8.8 color code tetron and a 4.8.8 Majorana surface code. The blue 4-Majorana operator at the boundary of the surface code is merged with a red 8-Majorana operator at the boundary of the color code to yield the light blue 12-Majorana operator. New 8- and 12-Majorana operators (light red) are introduced whose product is precisely  $Z_1 \otimes Z$ . Such a scheme can always be used to measure the product between a Pauli operator encoded in a color code qubit and a Pauli operator of a surface code. By using color codes for data qubits encoded in logical hexons and surface codes for ancillary tetron qubits, the Pauli product measurement scheme of Sec. III can be implemented with a lower space overhead compared to surface codes due to the more compact encoding of color codes, while at the same time benefiting from the low Majorana weight of surface-to-color code lattice surgery.

- 
- [1] A. Y. Kitaev, *Unpaired Majorana fermions in quantum wires*, *Sov. Phys. Usp.* **44**, 131 (2001).
- [2] A. Kitaev, *Fault-tolerant quantum computation by anyons*, *Ann. Phys.* **303**, 2 (2003).
- [3] R. M. Lutchyn, E. P. A. M. Bakkers, L. P. Kouwenhoven, P. Krogstrup, C. M. Marcus, and Y. Oreg, *Realizing Majorana zero modes in superconductor-semiconductor heterostructures*, *arXiv:1707.04899* (2017).
- [4] V. Mourik, K. Zuo, S. M. Frolov, S. R. Plissard, E. P. a. M. Bakkers, and L. P. Kouwenhoven, *Signatures of Majorana fermions in in hybrid superconductor-semiconductor nanowire devices*, *Science* **336**, 1003 (2012).
- [5] S. M. Albrecht, A. P. Higginbotham, M. Madsen, F. Kuemmeth, T. S. Jespersen, J. Nyg, P. Krogstrup, and C. M. Marcus, *Exponential protection of zero modes in Majorana islands*, *Nature* **531**, 206 (2016).
- [6] M. Deng, S. Vaitiekėnas, E. Hansen, J. Danon, M. Leijnse, K. Flensberg, J. Nygård, P. Krogstrup, and C. Marcus, *Majorana bound state in a coupled quantum-dot hybrid-nanowire system*, *Science* **354**, 1557 (2016).
- [7] H. Zhang, C.-X. Liu, S. Gazibegovic, D. Xu, J. A. Logan, G. Wang, N. van Loo, J. D. S. Bommer, M. W. A. de Moor, D. Car, R. L. M. O. het Veld, P. J. van Veldhoven, S. Koelling, M. A. Verheijen, M. Pendharkar, D. J. Pennachio, B. Shojajei, J. S. Lee, C. J. Palmstrom, E. P. A. M. Bakkers, S. Das Sarma, and L. P. Kouwenhoven, *Quantized Majorana conductance*, *arXiv:1710.10701* (2017).
- [8] B. M. Terhal, *Quantum error correction for quantum memories*, *Rev. Mod. Phys.* **87**, 307 (2015).
- [9] S. Bravyi, B. M. Terhal, and B. Leemhuis, *Majorana fermion codes*, *New J. Phys.* **12**, 083039 (2010).
- [10] S. Vijay and L. Fu, *Physical implementation of a Majorana fermion surface code for fault-tolerant quantum computation*, *Phys. Scr.* **T168**, 014002 (2016).

- [11] L. A. Landau, S. Plugge, E. Sela, A. Altland, S. M. Albrecht, and R. Egger, *Towards realistic implementations of a majorana surface code*, *Phys. Rev. Lett.* **116**, 050501 (2016).
- [12] S. Plugge, L. A. Landau, E. Sela, A. Altland, K. Flensberg, and R. Egger, *Roadmap to majorana surface codes*, *Phys. Rev. B* **94**, 174514 (2016).
- [13] Y. Li, *Noise threshold and resource cost of fault-tolerant quantum computing with Majorana fermions in hybrid systems*, *Phys. Rev. Lett.* **117**, 120403 (2016).
- [14] D. Litinski, M. S. Kesselring, J. Eisert, and F. von Oppen, *Combining topological hardware and topological software: Color-code quantum computing with topological superconductor networks*, *Phys. Rev. X* **7**, 031048 (2017).
- [15] D. Litinski and F. von Oppen, *Braiding by Majorana tracking and long-range CNOT gates with color codes*, *Phys. Rev. B* **96**, 205413 (2017).
- [16] S. Vijay, T. H. Hsieh, and L. Fu, *Majorana Fermion surface code for universal quantum computation*, *Phys. Rev. X* **5**, 041038 (2015).
- [17] Y. Li, *Fault-tolerant fermionic quantum computation based on color code*, [arXiv:1709.06245](https://arxiv.org/abs/1709.06245) (2017).
- [18] S. Vijay and L. Fu, *Quantum error correction for complex and Majorana fermion qubits*, [arXiv:1703.00459](https://arxiv.org/abs/1703.00459) (2017).
- [19] M. B. Hastings, *Small Majorana fermion codes*, [arXiv:1703.00612](https://arxiv.org/abs/1703.00612) (2017).
- [20] H. Bombin, *Topological order with a twist: Ising anyons from an abelian model*, *Phys. Rev. Lett.* **105**, 030403 (2010).
- [21] M. B. Hastings and A. Geller, *Reduced space-time and time costs using dislocation codes and arbitrary ancillas*, *Quantum Info. Comput.* **15**, 962 (2015).
- [22] D. Litinski and F. von Oppen, *Lattice surgery with a twist: Simplifying Clifford gates of surface codes*, [arXiv:1709.02318](https://arxiv.org/abs/1709.02318) (2017).
- [23] H. Bombin and M. A. Martin-Delgado, *Topological quantum distillation*, *Phys. Rev. Lett.* **97**, 180501 (2006).
- [24] D. Gottesman, *The Heisenberg representation of quantum computers*, *Proc. XXII Int. Coll. Group. Th. Meth. Phys.* **1**, 32 (1999).
- [25] S. Bravyi, G. Duclos-Cianci, D. Poulin, and M. Suchara, *Subsystem surface codes with three-qubit check operators*, *Quantum Info. Comput.* **13**, 963 (2013).
- [26] S. B. Bravyi and A. Y. Kitaev, *Quantum codes on a lattice with boundary*, [arXiv:quant-ph/9811052](https://arxiv.org/abs/quant-ph/9811052) (1998).
- [27] S. Plugge, A. Rasmussen, R. Egger, and K. Flensberg, *Majorana box qubits*, *New J. Phys.* **19**, 012001 (2017).
- [28] T. Karzig, C. Knapp, R. M. Lutchyn, P. Bonderson, M. B. Hastings, C. Nayak, J. Alicea, K. Flensberg, S. Plugge, Y. Oreg, C. M. Marcus, and M. H. Freedman, *Scalable designs for quasiparticle-poisoning-protected topological quantum computation with Majorana zero modes*, *Phys. Rev. B* **95**, 235305 (2017).
- [29] R. M. Lutchyn, J. D. Sau, and S. Das Sarma, *Majorana fermions and a topological phase transition in semiconductor-superconductor heterostructures*, *Phys. Rev. Lett.* **105**, 077001 (2010).
- [30] Y. Oreg, G. Refael, and F. von Oppen, *Helical liquids and Majorana bound states in quantum wires*, *Phys. Rev. Lett.* **105**, 177002 (2010).
- [31] B. Criger and B. Terhal, *Noise Thresholds for the  $[[4, 2, 2]]$ -concatenated Toric Code*, *Quant. Inf. Comp.* **16**, 1261 (2016).
- [32] 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).
- [33] X.-G. Wen, *Quantum orders in an exact soluble model*, *Phys. Rev. Lett.* **90**, 016803 (2003).
- [34] P. O. Boykin, T. Mor, M. Pulver, V. Roychowdhury, and F. Vatan, *New universal and fault-tolerant quantum basis*, *Inf. Proc. Lett.* **75**, 101 (2000).
- [35] S. Bravyi and A. Kitaev, *Universal quantum computation with ideal Clifford gates and noisy ancillas*, *Phys. Rev. A* **71**, 022316 (2005).
- [36] C. Horsman, A. G. Fowler, S. Devitt, and R. V. Meter, *Surface code quantum computing by lattice surgery*, *New J. Phys.* **14**, 123011 (2012).
- [37] M. S. Kesselring, B. J. Brown, F. Pastawski, and J. Eisert, *Boundaries and domain walls of the color code*, in preparation.
- [38] T. Karzig, Y. Oreg, G. Refael, and M. H. Freedman, *Universal geometric path to a robust Majorana magic gate*, *Phys. Rev. X* **6**, 031019 (2016).
- [39] A. J. Landahl and C. Ryan-Anderson, *Quantum computing by color-code lattice surgery*, [arXiv:1407.5103](https://arxiv.org/abs/1407.5103) (2014).
- [40] Y. Li, *A magic states fidelity can be superior to the operations that created it*, *New J. Phys.* **17**, 023037 (2015).
- [41] A. J. Landahl, J. T. Anderson, and P. R. Rice, *Fault-tolerant quantum computing with color codes*, [arXiv:1108.5738](https://arxiv.org/abs/1108.5738) (2011).
- [42] F. Pastawski and B. Yoshida, *Fault-tolerant logical gates in quantum error-correcting codes*, *Phys. Rev. A* **91**, 012305 (2015).
- [43] T. O'Brien, P. Rožek, and A. Akhmerov, *Majorana-based fermionic quantum computation*, [arXiv:1712.02353](https://arxiv.org/abs/1712.02353) (2017).
- [44] T. J. Yoder and I. H. Kim, *The surface code with a twist*, *Quantum* **1**, 2 (2017).
- [45] O. Zilberberg, B. Braunecker, and D. Loss, *Controlled-NOT gate for multiparticle qubits and topological quantum computation based on parity measurements*, *Phys. Rev. A* **77**, 012327 (2008).