Flag Gadgets Based on Classical Codes

Benjamin Anker[1,2,*] and Milad Marvian[1,2]

[1] *Department of Electrical and Computer Engineering, University of New Mexico, Albuquerque, New Mexico 87106, USA*

[2] *Center for Quantum Information and Control, University of New Mexico, Albuquerque, New Mexico 87106, USA*

(Received 20 February 2024; revised 22 August 2024; accepted 12 November 2024; published 10 December 2024)

Fault-tolerant syndrome extraction is a key ingredient in implementing fault-tolerant quantum computation. While conventional methods use a number of extra qubits that are linear in the weight of the syndrome, several improvements have been introduced using flag gadgets. In this work, we develop a framework to design flag gadgets using classical codes. Using this framework, we show how to perform fault-tolerant syndrome extraction for any stabilizer code with arbitrary distance using exponentially fewer qubits than conventional methods when qubit measurement and reset are relatively slow compared to a round of error correction. In particular, our method requires only $(2t+1)t\lceil \log_2(w)\rceil$ flag qubits to fault-tolerantly measure a weight-w stabilizer. We further take advantage of the saving provided by our construction to fault-tolerantly measure multiple stabilizers using a single gadget and show that it maintains the same exponential advantage when it is used to fault-tolerantly extract the syndromes of quantum low-density parity-check codes. Using the developed framework, we perform computer-assisted search to find several small examples where our constructions reduce the number of qubits required. These small examples may be relevant to near-term experiments on small-scale quantum computers.

DOI: 10.1103/PRXQuantum.5.040340

I. INTRODUCTION

A key challenge in designing fault-tolerant circuits is to control the spread of faults in the computation. The spread of faults in performing certain quantum operations can be controlled by the transversal implementation of the encoded operation, i.e., performing encoded operations using parallel local operations. Incorporating this strategy to perform syndrome measurement requires using many ancilla qubits and preparing certain less noisy highly entangled quantum states, both of which need to be done using faulty quantum operations. To satisfy these requirements, the traditional strategy has been to take a conservative approach to detecting errors in any circuit location by introducing many ancillary qubits and gates, and discarding and restarting the fault-tolerant subroutine any time that an error is detected. Since syndrome extraction is the most frequently needed subroutine in fault-tolerant constructions, this strategy leads to significant overhead in the number of qubits and gates to make a quantum algorithm fault tolerant. For example, Shor's fault-tolerant error-correction scheme [1,2] requires at least as many ancilla qubits as the weight of the syndrome measurement, Steane's error-correction scheme [3] requires as many ancilla qubits as the size of the code block, and recent progress has shown how to interpolate between the cost of these two schemes [4,5]. Recently, it has been shown that carefully designing gadgets that can "flag" the location of errors and adopting a more relaxed strategy in the error correction procedure can lead to substantial savings in the overhead cost of fault-tolerant schemes [6,7].

The flag-qubit paradigm [6,8–10] allows low-weight errors during syndrome extraction to propagate to high-weight correlated errors on the data. However, extra structure is added to the syndrome extraction circuit so that the low-weight errors can be identified and the correlated errors can be corrected. As originally presented [6], the flag-qubit technique only applies to the extraction of a handful of different syndromes and may require adaptive measurements, where syndrome measurements depend upon the flag pattern received. Since its inception, the technique has been extended in many directions [9,11–22] and has enabled several recent experiments [23–27], allowing for realizations of fault tolerance using current hardware in some cases.

In particular, flag gadgets have been shown to apply to any stabilizer code [8]. In architectures that allow for relatively fast qubit measurement and reset, syndrome extraction can be performed fault-tolerantly using only

*Contact author: banker@unm.edu

a constant number of ancilla qubits [8]. However, when qubit measurement and reset is slow or unavailable, this construction uses a number of ancilla qubits that scales linearly with the weight of the syndrome to be extracted, similar to conventional methods such as Shor's method [1]. Most relevant to this work, it has been shown that flag gadgets can be used for fault-tolerant syndrome extraction of any distance-3 code, using only a number of ancilla qubits logarithmic in the weight of the stabilizer, without requiring fast qubit measurement or reset [9].

An open question relating to flag gadgets is whether a qubit saving over conventional methods can be realized for general codes, without restricting the form or distance and without requiring fast measurement and reset. We answer this open question affirmatively and constructively.

In this work, we develop a framework to use classical codes in designing flag gadgets. This framework protects the qubit used for syndrome extraction with the parity checks used by a classical code with the same distance as the quantum code. The measurement of each flag qubit used corresponds to the result of one of the parity checks in the classical code. The resulting flag-qubit measurements can then be used to locate propagating errors on the syndrome, in analogy to the way in which parity checks for a classical code locate errors on a codeword. We show that locating propagating errors on the syndrome is sufficient to ensure fault-tolerant syndrome extraction. In addition, we show how the parity checks can be approximately implemented under physically relevant constraints such as limitations on the number of gates that can act on a qubit simultaneously and that the effect of imprecision can be handled by repeating the parity checks in space.

Using this framework, we present a construction that can be applied to any stabilizer code, with arbitrary distance. For a distance $d = 2t + 1$ quantum code, this construction uses the parity-check matrix for the distance-d Bose-Chaudhuri-Hocquenghem (BCH) code, repeated in space d times. The BCH code uses only $t\lceil \log_2(w+1)\rceil$ parity checks for a w-bit codeword, so our construction uses only $(2t+1)t\lceil \log_2 w\rceil$ flag qubits for extracting a weight-w stabilizer of a distance-d quantum code with slow reset, which is an exponential saving comparing to the conventional methods that use $O(w)$ many qubits.

Additionally, we show that we can apply our framework to a sequence of stabilizer measurements, instead of a single stabilizer measurement. Because of the logarithmic scaling of the proposed method, this technique allows for constructions that use vastly fewer qubits than conventional methods, when qubit measurement and reset is relatively slow. In particular, we show that in the absence of qubit reset, our proposed scheme can extract the syndrome of any quantum low-density parity-check (qLDPC) code [28] with exponentially fewer ancilla qubits compared to Shor-style syndrome extraction.

The mathematical framework proposed in this work provides a systematic approach to designing fault-tolerant gadgets, by enabling computer-assisted search to optimize the resources. We provide many examples of optimized small gadgets that are potentially suitable for near-term experiments. In addition, we provide three algorithms to decode the flags, i.e., to identify the locations of errors based on the flag patterns. We also discuss the savings provided using our constructions when the qubit reset is available but is slow compared to the two-qubit gates.

Our framework focuses on protecting the measurement of one or more stabilizer measurements against propagating errors. Because of this focus, our construction can straightforwardly replace any instance of Shor syndrome extraction, without requiring modification of the rest of the error-correction procedure, and therefore allows our construction to introduce qubit savings for alternative fault-tolerant error-correction procedures [29–31].

The organization of this paper is as follows. In Sec. II, we outline the main ideas of the flag-qubit paradigm. In Sec. III, we introduce our framework to represent flag gadgets and syndrome-extraction circuits using a binary matrix and state the fault-tolerance requirements in this language. In Sec. IV, we first present the intuition in the design of our framework under some simplifying assumptions on the connectivity of the device. In Sec. V, we remove the simplifying assumption and impose physically relevant constraints on the connectivity of the device and provide a proof that our construction is fault tolerant. The cost analysis is provided in Sec. V D and in Sec. VI we discuss alternative decoding schemes. Section VII outlines a technique to apply our construction to multiple syndrome-extraction circuits as if they were one larger syndrome-extraction circuit, and Sec. VIII analyzes the resources required by this technique, as opposed to flagging each measurement separately. In Sec. IX, we present the results of numerical simulations verifying fault tolerance for certain codes, in addition to providing examples of computer-assisted designs of syndrome measurements for small codes. Section X analyzes the trade-offs between our construction and other methods for fault-tolerant syndrome extraction as measurement and reset times vary.

II. FLAG GADGETS

The idea of using flag gadgets is to attach a simple quantum circuit to the syndrome qubit, such that we can identify the location of low-weight errors on the syndrome and correct the propagated errors on the data qubits.

This idea is illustrated in Fig. 1, using a simple example to measure $X^{\otimes 3}$. Flag qubits are connected to the syndrome qubit by controlled-NOT (CNOT) gates. Given the structure of the circuit, there are two types of faults that we need to consider: syndrome faults and flag faults. Syndrome faults have nontrivial support on the syndrome qubit, while flag

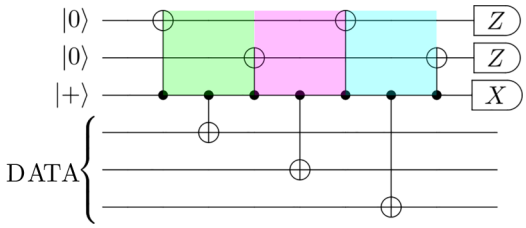

FIG. 1. An illustration of a flag-qubit circuit for measuring $X^{\otimes 3}$. The two qubits prepared in the $|0\rangle$ state are the flags and the qubit prepared in the $|+\rangle$ state is the syndrome qubit. If an error that can propagate to the data (an X error on the syndrome) occurs in the green (leftmost) region, only the top flag will produce a -1 result when measured. If an error occurs in the violet (middle) region, both flags will produce -1. If an error occurs in the aqua (rightmost) region, only the second flag will produce -1. Therefore, considering only errors on syndrome qubit, using the flag patterns we can distinguish and correct errors that propagate to the data.

faults have nontrivial support on a flag qubit. The errors on the syndrome qubit can spread to data qubits. Faults with trivial support on the syndrome qubit do not spread to data but can cause a "fake" flag pattern, which can trigger a correction step that introduces more errors on the data. The flags should identify syndrome faults while remaining resistant to flag faults.

If a flag qubit produces -1 upon being measured, we know that an error on the syndrome qubit has occurred before an odd number of CNOTs connecting the syndrome qubit to this flag qubit (or that a flag error occurred). We can then think of a pair of CNOTs connected to a single flag qubit as a "flag," which causes the flag qubit to switch sign whenever an error occurs on the syndrome qubit in the region it flags (the region between the two CNOTs). This is illustrated in Fig. 1. The error on the syndrome qubit is then inferred from the pattern of $+1$ and -1 given by the flag qubits (i.e., from the *flag pattern*).

Note that flag gadgets have been used to enable two distinct error-correction strategies. Flag gadgets were first used to protect error correction using *adaptive* syndrome extraction, where the choice of stabilizers measured is conditional on the results of the previous flag and stabilizer measurements [6,11,14,16,32]. However, subsequent generalizations have removed this requirement and instead focus on inferring errors during a static set of stabilizer measurements [8,9]. It is the latter picture on which we focus.

The other consideration in designing flag gadgets is the qubit reset time. Some flag-gadget constructions, such as in Ref. [8,10], assume that flag qubits can be reused within one stabilizer measurement. This requires that qubit measurement and reset are relatively fast, or requires idling while qubits are measured and reset (allowing more errors to be introduced). Note also that Ref. [8] gives the same asymptotic scaling as Shor's method in the absence of fast measurement and reset. Our construction in general does not require qubits to be reset quickly but may benefit from a fast reset. The resources required are analyzed for different reset time scales in Sec. X.

The idea of using flag gadgets has been extended in various directions. Flag gadgets have been applied to measuring the syndrome for the seven-qubit Steane code [11], topological and subsystem codes [12,13], cyclic Calderbank-Shor-Steane (CSS) codes [14], color codes [15,16], concatenated codes [17], state preparation [9, 18–20], entanglement certification [21], and circuit verification [22]. The technique has also been applied to schemes for reducing the number of stabilizer measurements required for fault tolerance [33] and to measuring stabilizers for a distance-3 CSS code in parallel [34]. Flag gadgets have been shown to be equivalent to encoding the ancilla qubit as a logical ancilla in certain cases [32]. The use of flag gadgets to protect against hardware-specific noise models has also been explored [35–38].

III. FRAMEWORK FOR DESCRIBING FLAG GADGETS

In this section, we introduce notation to describe the flag gadgets using certain binary matrices and state the fault-tolerance requirement using this notation.

We assume that the stabilizer $\sigma_1 \otimes \cdots \otimes \sigma_n$, where $\sigma \in \{I, X, Y, Z\}$, is measured by performing a controlled-σ_i gate on the ith data qubit with the control prepared in the $|+\rangle$ state and then measuring the control (syndrome) qubit in the X basis. Then, the only errors on the syndrome qubit that can propagate to the data qubits are X errors. In the examples and simulations presented in this paper, we assume that the stabilizer is of the form $X^{\otimes w}$ but the results can be directly applied to measuring general stabilizer operators.

Flag gadgets are used to identify where on the syndrome qubit X errors have occurred by initializing flag qubits in the $|0\rangle$ state and allowing X errors to propagate to the flag qubits. By measuring the flag qubits, we can deduce the propagated error as a function of the form of the stabilizer.

Since we only need to identify the location of X errors on the syndrome qubit, we describe the error on the syndrome qubit e_s by a binary column vector of height l and weight t_s, where l is the number of possible locations for an error (which depends on the weight of the stabilizer and on the flag construction). Throughout this work, we call e_s the *syndrome error*. In our construction based upon classical codes, an X directly after preparing $|+\rangle$ on the syndrome qubit is a stabilizer and hence we often omit this location from e_s when convenient. A key ingredient in specifying the fault-tolerant property of a circuit, and also in our constructions, is a description of how error propagates from the syndrome qubit onto flag qubits.

Definition 1. Let F be the matrix that describes how the syndrome error propagates to the flag qubits, i.e., Fe_s is a binary column vector with $(Fe_s)_i = 1$ if and only if an odd number of errors from the syndrome qubit propagate to flag qubit i.

Similarly, describe the flag error e_f by a binary vector column of height f and weight t_f, corresponding to the errors occurring on the flag qubits, where f is the number of flag qubits that a construction uses. Since errors on the flag qubits do not propagate to the data, only the parity of the error on a flag qubit needs to be considered, justifying the definition of e_f as a binary vector. Therefore, measuring the flag qubits reveals $Fe_s \oplus e_f$. Based on the result of flag measurements, we infer the data qubits to which we need to apply a correction; we call the corresponding map R.

We also define a matrix describing how errors propagate to the data.

Definition 2. Define D such that $(De_s)_i = 1$ if and only if an odd number of syndrome errors has propagated to data qubit i.

The goal of using flag qubits is to fault-tolerantly extract a syndrome using a few qubits. In this work, we use fault tolerance in the strong sense [31,39].

Definition 3. Syndrome extraction is said to be (strongly) *fault tolerant* up to t errors if the following conditions hold:

(1) If the data qubits are a weight-r Pauli correction from codeword c before syndrome extraction, and s faults occur during syndrome extraction and $r + s \leq t$ after syndrome extraction, the data qubits are at most a weight-$(r + s)$ Pauli correction from codeword c.
(2) If the data qubits are a weight-r Pauli correction from codeword c before syndrome extraction and s faults occur during syndrome extraction, after syndrome extraction the data qubits are at most a weight-$(r + s)$ Pauli correction from *any* codeword c'.

This definition is justified by the fact that if $(t+1)^2$ rounds of strongly fault-tolerant syndrome extraction are followed by a recovery operation according to Shor error correction [40] (or $(t+3)^2/4 - 1$ rounds using the correction procedure according to Tansuwannont *et al.* [31]), then the entire error-correction procedure is strongly fault tolerant [8,31].

Given these definitions, we succeed in fault-tolerantly identifying errors on the syndrome qubit if, for any e_s, e_f such that $t_s + t_f \leq t$, the residual error

$$De_s \oplus R(Fe_s \oplus e_f) \tag{1}$$

has weight at most $t_f + t_s$ up to a stabilizer.

We can describe a flag-gadget circuit diagram using a $(f + 1) \times l$ binary matrix C as follows.

Definition 4. The last row of C specifies the location of the CNOTs connecting the syndrome qubit to the data qubits. This row has w many nonzero elements. The remaining f rows describe the location of CNOTs connecting the syndrome qubit to flag qubits. The matrix element $C_{i,j} = 1$ if and only if there is a CNOT between the syndrome qubit and ith flag qubit at location j.

Given such a circuit matrix C, we can define the physical parity-check matrix F^c.

Definition 5. The matrix F^c is $f \times l$ and has as element (i,j) the sum $\sum_{k=0}^{l-j} C_{i,l-k}$ (over $\mathbb{Z}_2$).

This matrix F^c encodes how the syndrome errors propagate to the flag qubits, just as F does, but corresponds to a concrete physically implementable circuit C.

In this definition, we do not keep the row that encodes the location of the CNOTs connecting to the data. However, other than neglecting the data qubits, F^c completely characterizes C, so we can work with the two representations interchangeably. In our construction, we start with a matrix F based upon some classical code and modify it to produce F^c so that C (the circuit matrix) satisfies some set of physically motivated constraints (described in Sec. V). Analogously, we define D^c as the matrix that describes how errors on the syndrome qubit propagate to the data relative to a given circuit C. This is summarized in Fig. 2.

The fault-tolerant requirement of the syndrome-measurement circuit can be summarized [in a similar form to Eq. (1)] using the circuit diagram matrix C by requiring

$$\min\left(\left|D^c e_s \oplus R(F^c e_s \oplus e_f)\right|, w - \left|D^c e_s \oplus R(F^c e_s \oplus e_f)\right|\right) \leq k \tag{2}$$

for $k := t_f + t_s \leq t$, where we have included the fact that the syndrome that we measure is a stabilizer. Of course, in the case that $F^c = F$ and $D^c = D$, this reduces to Eq. (1).

IV. FLAG GADGETS BASED ON CLASSICAL CODES

We now show that by constructing the flag gadgets according to the parity checks of appropriate classical codes, we can fault-tolerantly identify and correct syndrome faults in measuring any stabilizer. For simplicity, in this section, we assume that multiple CNOTs can be performed simultaneously and only suffer weight-1 faults—we will remove these assumptions in Sec. V.

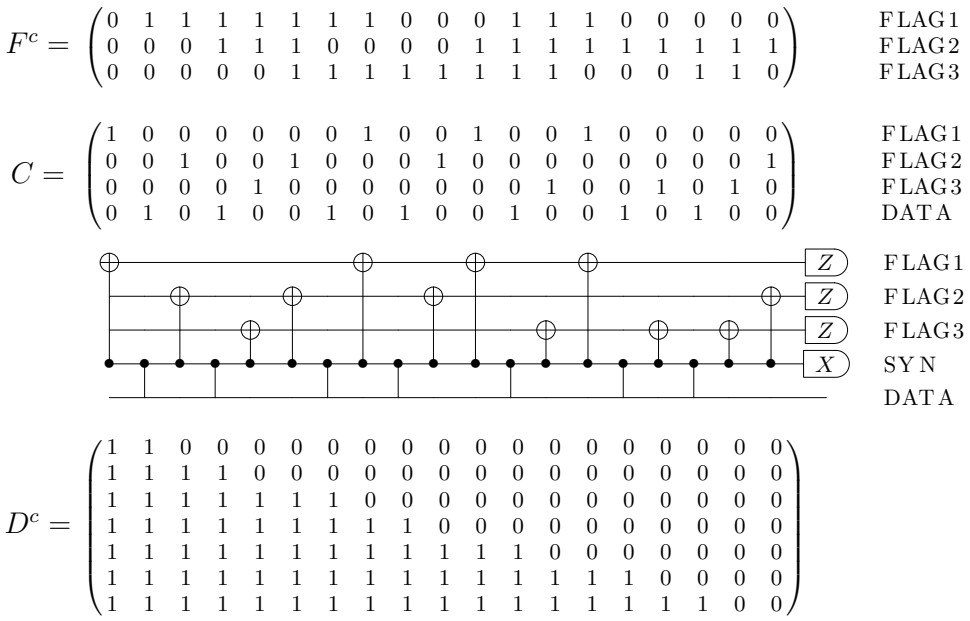

FIG. 2. An illustration of how a physical parity-check matrix F^c and its associated circuit matrix C correspond to a physical circuit and its associated data matrix D^c.

A. Flagged syndrome extraction for a distance-3 code

We start with an illustrative example, in which we show how to fault-tolerantly measure a weight-$(w+1)$ stabilizer of a distance-3 code using $3\lceil \log_2 w \rceil$ flag qubits and one syndrome qubit. Consider

$$F = \begin{pmatrix} H \\ H \\ H \end{pmatrix},$$

where

$$H = \begin{pmatrix} 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 \end{pmatrix}$$

is the parity-check matrix for the Hamming code on w (physical) bits. Imposing no physical constraints, we let $F^c = F$. The corresponding circuit is then uniquely identified by specifying one data CNOT per column of F. An example circuit of the last three flag qubits for $w = 8$ is shown in Fig. 3 (the other six flag qubits are excluded due to space constraints).

Clearly, if an error occurs on the syndrome qubit between data CNOTs i and $i+1$, the flag pattern produced is the same (replacing 1s with 0s and -1s with 1s) as the result of the multiplication Fe, where e is a binary vector with a single 1 as the ith entry. This allows us to prove an illustrative first lemma.

Lemma 1. Given

$$F^c = F = \begin{pmatrix} H \\ H \\ H \end{pmatrix},$$

we can fault-tolerantly identify the location of a single fault on the syndrome qubit, up to a stabilizer.

Proof. Suppose that the flag pattern is given by P. Consider two errors, e_1 and e_2, that produce the flag pattern P. We will show that the errors must propagate to the data identically (i.e., $De_1 = De_2$) up to a stabilizer.

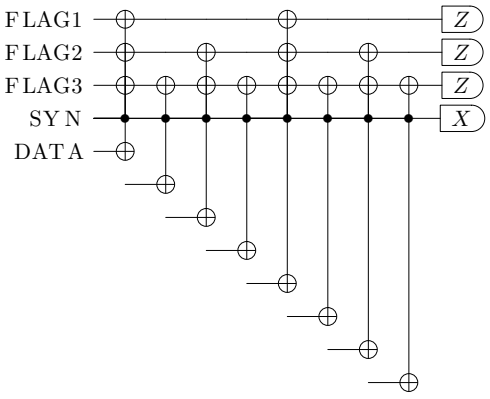

FIG. 3. The parity-check matrix for the $[7,4,3]$ Hamming code implemented with CNOTs, with spaces between data CNOTs corresponding to physical bits of the codeword.

For the purposes of this proof, call the trivial error (the no-error case) a syndrome error. Syndrome errors produce a flag pattern of weight either 0 (corresponding to an error before or after all CNOTs composing the flag gadget, or the trivial error) or at least 3, since columns of F have at least 3 nonzero entries (since columns of H have at least 1 nonzero entry).

Consider first the case in which e_1 is a flag error, while e_2 is a syndrome error. A flag error produces a flag pattern with exactly one nonzero entry, so the flag patterns cannot be the same and this case is impossible.

Next, suppose that e_1 and e_2 are both flag errors. Then, they do not propagate to the data.

Now suppose that e_1 and e_2 are both syndrome errors. If the flag pattern has zero nonzero entries, both errors propagate to a stabilizer. If the flag pattern is of weight at least 3, both errors must be between the same pair of data CNOTs, since the columns of F are distinct and hence only an error between data CNOTs $i, i+1$ produces column F_i. So $De_1 = De_2$.

This shows that any two errors with the same flag pattern propagate to the same error on the data, so we can assign a unique correction to any flag pattern. ■

B. Flagged syndrome extraction for a distance-d code

The picture is similar when we consider more than one error occurring during syndrome extraction.

Lemma 2. Consider the flag gadget based on the binary matrix

$$F = \begin{pmatrix} H \\ \hline \cdots \\ \hline H \end{pmatrix}$$

consisting of d repetitions of H, where H is a parity-check matrix for the distance-d BCH code [41,42] (for a summary of the BCH-code parity-check matrix, see Appendix A). We show that two errors, e_1 and e_2, such that $|e_1| + |e_2| < d$ and which produce the same (fixed) flag pattern P for this flag gadget satisfy $De_1 = De_2$ up to a stabilizer.

Proof. Since H is the parity-check matrix for a distance $d = 2t+1$ code, the column-wise sum $\sum_{k=1}^{n} H_k$ is nonzero, where H_k is the kth column of H and $1 \le n \le 2t$. This implies that $\left|\sum_{k=1}^{n} F_k\right| \ge 2t+1$ for all n such that $1 \le n \le 2t$. Consequently, any flag pattern P has exactly one associated set of up to t columns $\{F_k\}_{0\le k\le t}$, up to including the all-zeros column (corresponding to an error before or after all CNOTs), such that $\left|P \oplus \sum_k F_k\right| \le t$. To see this, observe by the triangle inequality that if there were two such sets of columns, their sum would have weight strictly less than $2t+1$. Each column F_k included implies that an odd number of syndrome errors have occurred between data CNOTs k and $k+1$. This implies that any two errors e_1 and e_2 with a different parity of syndrome errors between data CNOTs $i, i+1$ for any i (i.e., any two errors such that $De_1 \neq De_2$) have distinct flag patterns and we can again assign a unique correction to each flag pattern. ■

This shows that in the case in which any number of CNOTs can act simultaneously, our construction is capable of perfectly identifying De for any e composed of syndrome and flag errors. In Sec. V, we show that in the absence of the ability to perform multiple CNOTs simultaneously, our construction is still able to approximately infer De and that the approximation is fault tolerant.

V. IMPOSING PHYSICAL CONSTRAINTS

When implementing a quantum error-correcting code on hardware, there are many physical constraints to work around; e.g., connectivity constraints, constraints on how many qubits can be involved in a gate, and constraints on how many gates can act simultaneously.

We consider a physically motivated constraint that we believe to be one of the more difficult ones to overcome: two CNOTs with a common control or target qubit cannot be applied simultaneously. We show that this constraint does not affect our construction. First, in Sec. V A, we provide a method to modify a given parity-check matrix so that the corresponding circuit does not use more than one CNOT at once. Then, in Sec. V B, we show that this modification still allows for fault-tolerantly identifying syndrome faults by repeating a classical parity-check matrix.

A. Unfolding a parity-check matrix

In this section, we describe an "unfolding" procedure to construct C or, equivalently, F^c, matrices that satisfy Eq. (2) when starting with an F matrix of a certain form but that do not require performing multiple CNOTs simultaneously. We will describe a form for F such that this unfolding procedure is sufficient to satisfy Eq. (2) in Sec. V B. We assume that F is some number of parity-check matrices for a BCH code [41,42] stacked on top of each other (for a brief summary of BCH codes, see Appendix A).

Given an F matrix, we first append a zero column to the right and left sides of the matrix. Then we add columns between adjacent columns of the appended matrix such that two consecutive columns differ in exactly one element. More precisely, we define a new matrix F^c such that

$$\begin{cases} F^c_{i,0} = F_i, & \forall i, \\ F^c_{i,k} = F_i \oplus [F_i \oplus F_{i+1}][k], & \forall i, k, \end{cases} \tag{3}$$

where $F^c_{i,k}$ is a column of the unfolded matrix, with two indices, i and k, which together specify the column in question. The notation $[F_i \oplus F_{i+1}][k]$ denotes a column vector

of the same dimensions as F_i and the same elements up to the kth nonzero element of $F_i \oplus F_{i+1}$, with zeros after the kth nonzero element. For each index i, the second index, k, is bounded above by the number places where columns F_i and F_{i+1} differ. By indexing k from zero, we recover F by considering the columns $F_{i,0}$ for i ranging over the column indices of F. The matrix F^c then has the form

$$F^c = \big(F_1, \quad F_1 \oplus [F_1 \oplus F_2][1], \quad \ldots | \quad F_2, \quad F_2 \oplus [F_2 \oplus F_3][1], \quad \ldots | F_m, \quad F_m \oplus [F_m \oplus F_{m+1}][1], \quad \ldots\big). \tag{4}$$

Each column of the unfolded parity-check matrix consists of a stack of subcolumns, each subcolumn being either a column of H or the zero column, except for at most one subcolumn that describes the transition between columns of H. The matrix F^c and the location of the data CNOTs uniquely specify C by just considering the difference of two consecutive columns of F^c—for a small example, see Fig. 4.

B. General fault-tolerant flag gadgets

We now use the unfolding method presented in Sec. V A to construct general fault-tolerant flag gadgets. The proof is analogous to the proof in Sec. IV B and follows from the observation that the independence of columns of H guarantees that any two errors with the same flag pattern must be very similar.

Given an arbitrary t, define F as the stack of $2t+1$ copies of the parity-check matrix, H, where H defines a distance $2t+1$ error-correcting code (e.g., the BCH code) on top of each other, i.e., define

$$F = \begin{pmatrix} H_1 \\ \hline \vdots \\ \hline H_{2t+1} \end{pmatrix},$$

where H_k is simply one copy of H.

As in Sec. IV A, define D to correspond to the physical implementation in which a data CNOT comes between each pair of columns of F.

Note that in this definition of D, we do not distinguish between an error directly before a data CNOT and an error directly after a data CNOT. This is consistent with the fact that we cannot distinguish these two errors by any flag construction and it does not impact fault tolerance, since misidentifying an error directly before a data CNOT as an error directly after a data CNOT (or vice versa) leads to at most one error on the data (which is allowable since the original number of errors is at least one).

Lemma 3. Let e_1 and e_2 be two errors that produce the same flag pattern P. Denoting the vector of all syndrome errors by $e := e_{1,s} \oplus e_{2,s}$, we have $|De_{2i} \oplus De_{2i+1}| \leq 1$ for e_j the jth nonzero component of e.

Proof. We will show that syndrome errors occur in pairs, with at most one data CNOT between members of the pair.

Write P as $e_{1,f} \oplus \sum_{i \in I_1} F^c_{1,i}$, where I_1 is the set of indices corresponding to nonzero entries of $e_{1,s}$. This defines the set of columns $F^c_{1,i}$ corresponding to each syndrome error in e_1. Decomposing P in terms of e_2 defines the set of columns $F^c_{2,i}$ corresponding to each syndrome error in e_2.

Assuming that each H_k has r rows, define $F^k_{*,i}$ as the vector consisting of the $r(k-1)$th through rkth entries of $F_{*,i}$, and similarly for $e^k_{*,f}$.

Consider the set of all H_k such that:

(1) $F^k_{1,i}$ is a column of H_k, or is the zero vector
(2) $F^k_{2,i}$ is a column of H_k, or is the zero vector
(3) H_k is affected by zero flag errors (i.e., $e^k_{1,f}$ is the zero vector, as is $e^k_{2,f}$)

We first show that there exists at least one such H_k. Start with the set $N = \{H_k\}_{k \leq 2t+1}$. For each column $F_{1,i}$, at most one subcolumn of $F_{1,i}$ is both *not* a column of H and *not* the zero vector. After removing all submatrices that do not satisfy (1), we are left considering a set N' of size at least $|N| - |e_{1,s}|$, since $F^k_{*,i}$ is neither a column of H_k nor the zero vector for at most one k and i ranges from 1 to $|e_{1,s}|$. Similarly, after removing all submatrices that do not satisfy (2), we are left to consider at least $|N| - |e_{1,s}| - |e_{2,s}|$ submatrices. Finally, after removing all submatrices that do not satisfy (3), we are left to consider at least $|N| - |e_{1,s}| - |e_{2,s}| - |e_{1,f}| - |e_{2,f}|$ submatrices.

Noting that $|N| = 2t+1$ and that $|e_{1,s}| + |e_{2,s}| + |e_{1,f}| + |e_{2,f}| \leq 2t$, we see that we have at least $2t+1-2t = 1$ submatrix satisfying (1)–(3).

We can now let k be any index such that H_k satisfies the three conditions given. We use the columns of H_k to characterize the relation between $e_{1,s}$ and $e_{2,s}$.

Recall that $F^k_{1,i}$ and $F^k_{2,i}$ are either columns of H_k or the zero vector by assumption. Since e_1 and e_2 have the same syndrome and $e^k_{1,f} = e^k_{2,f} = 0$ by assumption, we must have $\bigoplus_{i \in I_1} F^k_{1,i} \oplus \bigoplus_{i \in I_2} F^k_{2,i} = 0$. Since H has distance $2t+1$, the sum of any *distinct* $2t$ of its columns is nonzero. Since the sum *is* zero, the columns must be

$$F = \begin{pmatrix} \cdots & 1 & 1 & 0 \\ & 1 & 0 & 1 \\ & 1 & 1 & 1 \end{pmatrix} \xmapsto{\text{appending } \vec{0}} \begin{pmatrix} \cdots & 1 & 1 & 0 & 0 \\ & 1 & 0 & 1 & 0 \\ & 1 & 1 & 1 & 0 \end{pmatrix} \xmapsto{\text{unfolding}} \begin{pmatrix} \cdots & 1 & 1 & 0 & 0 & 0 & 0 \\ & 1 & 0 & 0 & 1 & 0 & 0 \\ & 1 & 1 & 1 & 1 & 1 & 0 \end{pmatrix} = F^c$$

$$C = \begin{pmatrix} & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ \cdots & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 \\ \cdots & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \end{pmatrix}$$

FIG. 4. An example of the unfolding procedure, with columns of the original parity-check matrix in green and transition columns in purple. The first, third, sixth, and eighth columns correspond to the CNOTs connecting to the data, while the remaining columns are the differences between consecutive columns of F^c. Because they are weight 1, they also correspond to a single CNOT.

nondistinct. In particular, either $F^k_{*,i} = 0$ or $F^k_{*,i} = F^k_{*,j}$ with a unique j paired to each i. By Appendix B, we can assume that zero-vector columns correspond to errors before or after measurement, which will propagate to a stabilizer. Each pair of errors corresponding to a pair of identical columns must have at most one data CNOT between them, since columns of H are distinct and each column is expanded to cover exactly one data CNOT. ■

Lemma 4. Let S be an arbitrary flag pattern. Consider the set of all errors that produce the given flag pattern, $E := \{e = e_s + e_f : F^c e_s \oplus e_f = S, |e_s| + |e_f| \leq t\}$. Then, for any $e_1, e_2 \in E$ such that $|e_1| \leq |e_2|$, we have $\left|De_{1,s} \oplus De_{2,s}\right| \leq \left|e_{2,s}\right| \leq |e_2|$.

Proof. Let $e_1, e_2 \in E$ have syndrome error weights $\left|e_{1,s}\right| = k$ and $\left|e_{2,s}\right| = n$ with $k \leq n$.

We know that each component error (except for possibly one in the boundary, which does not affect De anyway) has a paired error with at most one data CNOT in between by Lemma 3. Consider all of the component errors of $e_{1,s}$ that have their pair in $e_{2,s}$. If there are l of these, $De_{1,s} \oplus De_{2,s}$ has weight at most $l + [(n-l)/2] + [(k-l)/2] = l - [(n+k-2l)/2] = [(n+k)/2] \leq (2n/2) = n$.

We obtain the term on the left-hand side by considering the error $e_{2,s}$ and applying the error $e_{1,s}$ as the correction. Then, we note the following:

(1) When we apply $e_{1,s}$ to the syndrome qubit, the l corrections that have their pair in $e_{2,s}$ each introduce at most one error on the data (corresponding to the data CNOT possibly in the middle of the pair).
(2) The $k - l$ errors that are not part of $e_{2,s}$ but that are included in the correction consist of pairs and so their correction introduces at most $(k-l)/2$ errors.
(3) The $n - l$ errors that are part of $e_{2,s}$ that we do not correct are pairs (separated by at most one data CNOT) and hence introduce at most $(n-l)/2$ errors, which stay on the data after correction. ■

Corollary 1. F^c satisfies the condition in Eq. (2).

Proof. Again letting S be arbitrary, consider the set of all errors that produce S, namely, $E := \{e = e_f + e_s : F^c e_s \oplus e_f = S, t_s + t_f \leq t\}$. Then, let the correction operator be given by $D^c c$, where c is defined as $arg\,min_{e \in E} |e|$. We wish to show that $\left|D^c e_s \oplus R(F^c e_s \oplus e_f)\right| \leq t_s + t_f$ for any $e \in E$. Rewriting, we wish to show that $|D^c c \oplus D^c e_s| \leq t_s + t_f$. This holds by Lemma 4, since by definition $|c| \leq |e|$. ■

C. Application to non-CSS syndrome measurements

Suppose that we wish to measure an operator $X^{\otimes n} Z^{\otimes m}$. Note that this applies to the measurement of Y-type operators if n and m are not disjoint. To measure this operator, usually some of the CNOTs (corresponding to the X terms in the syndrome) are replaced with controlled phase gates, while the syndrome qubit is still measured in the X basis. This means that the only type of error that can propagate from the syndrome qubit to the data is still an X-type error. As such, we can use exactly the same set of flags that we use to protect the syndrome for a CSS code.

In Fig. 5, this is illustrated for measuring the $XZZXI$ syndrome of the $[\![5, 1, 3]\!]$ perfect code.

D. Cost analysis to implement a single stabilizer

Having produced a fault-tolerant flag-qubit construction, we analyze the circumstances in which it outperforms other constructions to protect a single stabilizer measurement. We mainly focus on the number of flag qubits used, as for near-term small-scale devices the number of qubits is severely limited; and also for large-scale devices, reducing the number of qubits needed for fault tolerance allows for more efficient use of hardware.

In our construction, the number of flag qubits used is $(2t+1)n(w,t)$, where $n(w,t)$ is the number of rows in the classical parity-check matrix. For the BCH code on w bits with distance $2t+1$, the number of rows is given by $t\lceil \log_2(w) \rceil$ (see Appendix A) for a total cost of $(2t^2 + t)\lceil \log_2(w) \rceil$ flag qubits.

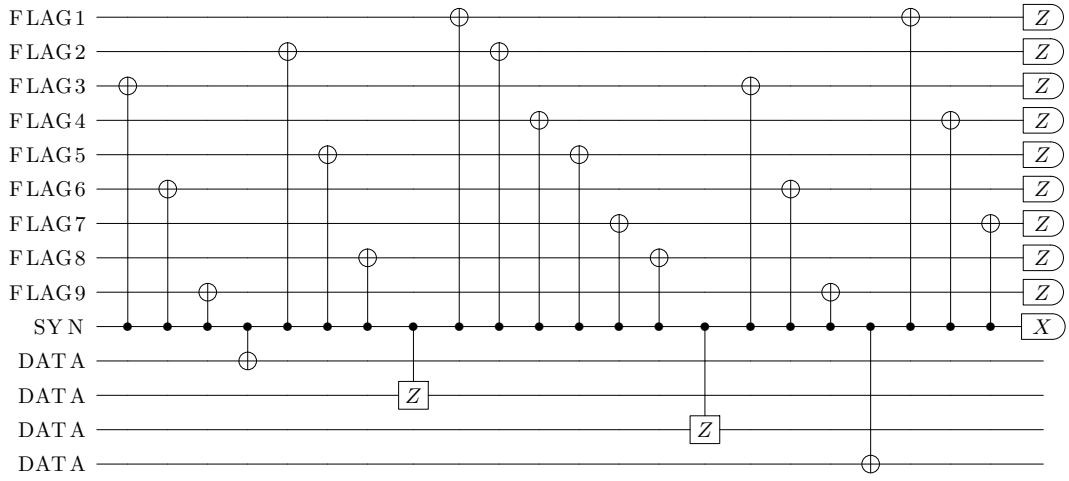

FIG. 5. An example of our flag construction for measuring the *XZZXI* syndrome of the five-qubit perfect code. Note that if we were to measure *XXXXI*, the flag circuit would be identical.

For $t = O(1)$, this is asymptotically logarithmic in the weight of the stabilizer, which is optimal. However, for small w, t, the constant factors can be large. For example, for $w = 15$ and $t = 2$, the number of flag qubits used is $(10)(4) = 40$, which is outperformed by the various schemes for fault tolerance that scale linearly in the weight of the syndrome, ignoring the cost of state preparation. Even reducing the number of repetitions to $t + 2$, as in Sec. IX, yields 32 flag qubits. It is only once $w \geq 70$ (or $w \geq 48$, assuming that we only need to repeat $t + 2$ times) that our construction outperforms schemes that scale linearly in the weight of the syndrome.

This suggests that the exponential saving provided by our constructions only shows up in codes with high-weight stabilizers, such as Bacon-Shor [43] codes on certain choices of the lattice. However, in Sec. VII, we present a slight modification to the syndrome-extraction procedure that allows our construction to protect multiple syndromes at once. The estimates using the analytical results are also large because we wish to prove the fault tolerance of the general construction. If we use our framework for numerical searches, we see that it can provide resource savings even for small codes (see Sec. IX).

Comparisons of the scaling of our construction versus other flag constructions can be found in Table I.

Note that our framework also captures previous flag-gadget constructions; the construction given by Reichardt *et al.* [6] can be seen as a repetition code and the construction given by Prabhu *et al.* [9] is based upon a punctured Hamming code. In our generalization, the increased overhead is mostly due to the qualitatively different fault-tolerance requirements imposed by increasing the distance. In Sec. IX, we also explore using our framework to numerically search for small flag gadgets and show that the overhead can be substantially reduced in some cases.

We now briefly address the cost in CNOTs to implement our construction. As a very pessimistic upper bound, each row of the parity-check matrix H will alternate between 0 and 1, requiring w CNOTs per row for a weight-w syndrome. With $(2t + 1)t\lceil \log_2 w \rceil$ total rows, this implies a maximum CNOT cost of $(2t + 1)tw\lceil \log_2 w \rceil \sim t^2 w \log_2 w$. Realistic parity-check matrices do not have rows that alternate between 0 and 1, making this upper bound very loose.

VI. DECODING PROCEDURES

Given a certain flag pattern, we must be able to infer a correction to apply to the data; i.e., we must decode the flag pattern. In this section, we outline two different decoding algorithms.

A. Brute force

Section V B implicitly defines a decoder. We construct a table of errors and how they propagate. We can consider the set of all errors $e_1, \ldots, e_n$ that have the same syndrome

TABLE I. A comparison of the resources required to use our construction versus other methods that apply to general weight stabilizers.

w	t	Our construction	Prabhu *et al.* [9]
25	1	16	6
50	1	19	7
75	1	22	8
100	1	22	8
25	2	51	N/a
50	2	61	N/a
75	2	71	N/a
100	2	71	N/a

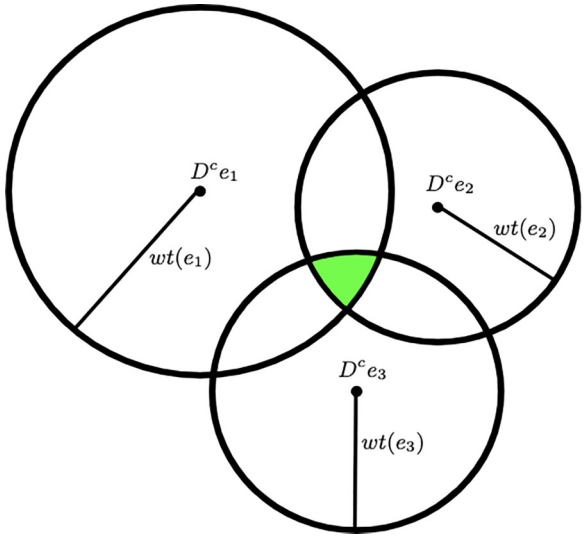

FIG. 6. An example of the decoding process for a set of errors e_1, e_2, and e_3 with a common syndrome. In green is the set of all corrections that do not increase the number of errors on the data regardless of which physical error produced the observed syndrome. Note that this figure is only for illustrative purposes. By using at least $2t+1$ repetitions with our construction, a picture with overlapping balls, neither strictly contained in the other, will not arise. Instead, the center of each small ball will be contained in each larger ball.

(i.e., $F^c e_i = F^c e_j$ for all i,j). Since we cannot distinguish between $e_1, \ldots, e_n$, we need to find some correction that will correct any e_i fault tolerantly.

We can phrase this condition in terms of Hamming balls drawn about the data error to which each e_i propagates. Succinctly, there exists a fault-tolerant recovery map R if and only if, for arbitrary flag pattern P, we have

$$\bigcap_{e_i \in \{e : F^c e = P\}} B_{|e_i|}(D^c e_i) \neq \emptyset,$$

where $B_r(d)$ is the Hamming ball with radius r around the point d in the space of all possible errors (or corrections) on the data. This is obvious if we observe that $B_{|e_i|}(D^c e_i)$ is the set of corrections that do not increase the number of errors on the data from the original number of physical errors. This condition is illustrated in Fig. 6.

The brute-force decoding algorithm simply computes the intersection of all Hamming balls with appropriate radii centered at the data errors arising from collections of faults with the same flag pattern and picks an arbitrary member of the intersection as the correction to apply.

B. Majority-vote decoder

Another natural procedure presents itself. Intuitively, we can use the repetition of the parity-check matrix H to correct the errors on the flag pattern (i.e., flag errors) and then use the new flag pattern obtained to apply a correction that does not increase the number of errors on the data from the original number of physical errors based by decoding the classical code hidden inside our flag qubits.

First, we define a useful concept.

Definition 6. For an error e and a matrix F composed of repetitions of a parity-check matrix H, we call repetition i *error free* if e has support entirely on columns of H or zero columns.

This definition captures the repetitions of H that give the same information that the classical parity-check matrix would give were it to act perfectly upon a codeword with errors.

Lemma 5. If F is composed of $(t+1)^2$ repetitions of a classical parity-check matrix H, the following correction procedure produces a fault-tolerant correction for any error-producing flag pattern P:

(1) Divide P into $(t+1)^2$ subcolumns, P_i, $1 \leq i \leq (t+1)^2$.
(2) Let P^* be the subcolumn that occurs most frequently.
(3) Taking P^* as the syndrome for the classical parity-check matrix H, decode P^* and apply the correction indicated.

Proof. Note that for any error e, the classically computed correction associated with a subpattern produced by an error-free repetition is a fault-tolerant correction for e.

The fact that such a correction is fault tolerant follows from the fact that in an error-free repetition, the correction produced by the classical decoder exactly identifies the parity of syndrome errors in the area covered by each column of H. So at worst, the correction produced neglects to correct some even number of errors in each region covered by a single column of H. As before, the area covered covers at most one data CNOT and hence neglecting to correct these errors still yields a fault-tolerant correction. So it is enough to show that the most common subpattern is produced by an error-free repetition.

If e is incident upon a column of repetition i that is not a column of H and e is incident upon a column of repetition j that is not a column of H, but for all repetitions k for $i < k < j$, all errors are incident upon a column of H or the zero column and no repetition k suffers a flag error, all subpatterns from repetitions $i+1$ to $j-1$ are identical. So a single error $e_i \in e$ can divide a region of repetitions into two regions with possibly different patterns, with some arbitrary subpattern in between. So it is enough to show that dividing a region of length $(t+1)^2 - t$ into t subregions produces at least one subregion of length $t+1$. Since $[(t+1)^2 - t]/t > t + 2 - 1 = t + 1$, this condition

is satisfied. So the most common subpattern has multiplicity at least $t+1$ and hence can only be produced by $t+1$ error-free repetitions in a row. ■

C. Intermediate decoder

Finally, we present an intermediate decoding algorithm. This algorithm still scales exponentially with t but not as badly as the brute-force version. It also only requires $2t+1$ repetitions of the parity-check matrix, as does the brute-force version.

We first decode each subpattern with a decoder for the BCH code (skipping any that cannot be decoded). Write the decoded error as a bit string of syndrome errors. The results of decoding any two error-free repetitions must differ by a bit string of the form $\sum_{i=1}^{k} 0^{f_1(i)}110^{f_2(i)}$, where k is upper bounded by the number of non-error-free repetitions between them and $0^{f_{0,1}(i)}$ denotes some number of zeros that depends on i.

The locations of the 11 substrings in each term of the sum in the difference vector correspond to the location of a transition error. Assuming that we suffer at most t errors, there are at least $t+1$ error-free repetitions, which differ from each other in the way described above. Recall that a minimum-weight error that is error free in repetition i and produces the flag pattern P is a fault-tolerant correction for any other error that is error free in repetition i and produces P. So the problem is reduced to finding a minimal number of strings of the form $0^{f_1(i)}110^{f_2(i)}$ and then assigning each one to a repetition, such that each consecutive pair of the remaining repetitions differ from each according to the sum of strings assigned to repetitions between them. After finding such a set of difference vectors, we can choose any of the remaining repetitions as the correction.

D. Comparison of decoding costs

The brute-force decoder described in Sec. VI A requires compiling a table of all errors and their syndromes. Clearly, the dominant term in the time cost for this decoder is the cost of considering every error. The number of distinct errors is given by

$$\sum_{k=0}^{t}\binom{n}{k},$$

where n is the total number of CNOTs used in the circuit (equivalently, the weight of C), which is proportional to the number of flag qubits f. This scales exponentially with the number of flag qubits. If l is the least number of CNOTs attached to a flag qubit, then this decoding algorithm requires $\Omega(\sum_{k=0}^{t}\binom{l(2t+1)+w}{k})$ operations. Using the bound $\sum_{k=0}^{t}\binom{n}{k}=\Omega(2^{H(t/n)n})$, we can lower bound this by $\Omega(2^{l(2t)}2^{H(l)})=\Omega(2^{2lt})$, where $H(x)$ is the binary entropy of $0\le x\le 1$.

However, the majority-vote decoding procedure in Sec. VI B only requires finding the most frequently occurring subpattern of length $t\log_2 w$, before decoding one classical syndrome.

In the case of the BCH code, the decoding algorithm requires $O(wt)$ operations [44]. Finding the most frequent subpattern takes $O(t^5\log_2(w))$ operations for a total complexity of $O(t^5\log_2 w+wt)$.

Finally the intermediate decoder in Sec. VI C has complexity $O(\binom{w}{t}\binom{2t+1}{t})$, where the first term is the number of ways to choose the left or right location of the transition columns and the second term is the number of ways to choose the repetition into which each transition column falls.

VII. PROTECTING SEVERAL STABILIZER MEASUREMENTS

In general, several syndromes need to be extracted consecutively and the entire sequence needs to be repeated to account for measurement errors (in the Shor error-correction picture). In this section, we show that the flag-qubit construction in Sec. V B can be trivially modified to identify error locations in this whole sequence of repeated measurements. Because our construction enjoys sublinear scaling, flagging the entire sequence of syndrome extraction requires fewer qubits than flagging each measurement separately.

A. Connecting and flagging a sequence of stabilizers

Consider a syndrome-extraction circuit in the form of Fig. 7(a). We can modify this circuit by adding CNOTs connecting subsequent pairs of syndrome qubits, without changing the measurement outcomes observed, as in Fig. 7(b). This modification means that X errors on syndrome qubit i will propagate to all syndrome qubits $j>i$. X errors that propagate from one syndrome qubit to another propagate to trivial errors (stabilizers) on the data and do not affect the measurement results since we measure in the X basis. Adding this modification means that a flag gadget starting on syndrome qubit i and ending on syndrome qubit $j>i$, as in Fig. 7(d), will flag if there is an odd number of errors in the range that it covers, regardless of which syndrome qubit they occur on (where we interpret any errors occurring after the connecting CNOT as unflagged measurement errors). By modifying the circuit one more time and connecting consecutive stabilizer measurements via an ancilla prepared in the $|0\rangle$ state, instead of directly, we can ensure that preparation errors on a later syndrome qubit do not propagate to measurement errors on an earlier syndrome qubit [Fig. 7(c)].

Suppose that we wish to measure L stabilizers, each with weight w_i. To ensure fault tolerance, each round of measurement must be repeated s times, where s depends on the properties of the code. Usually, s is taken to be

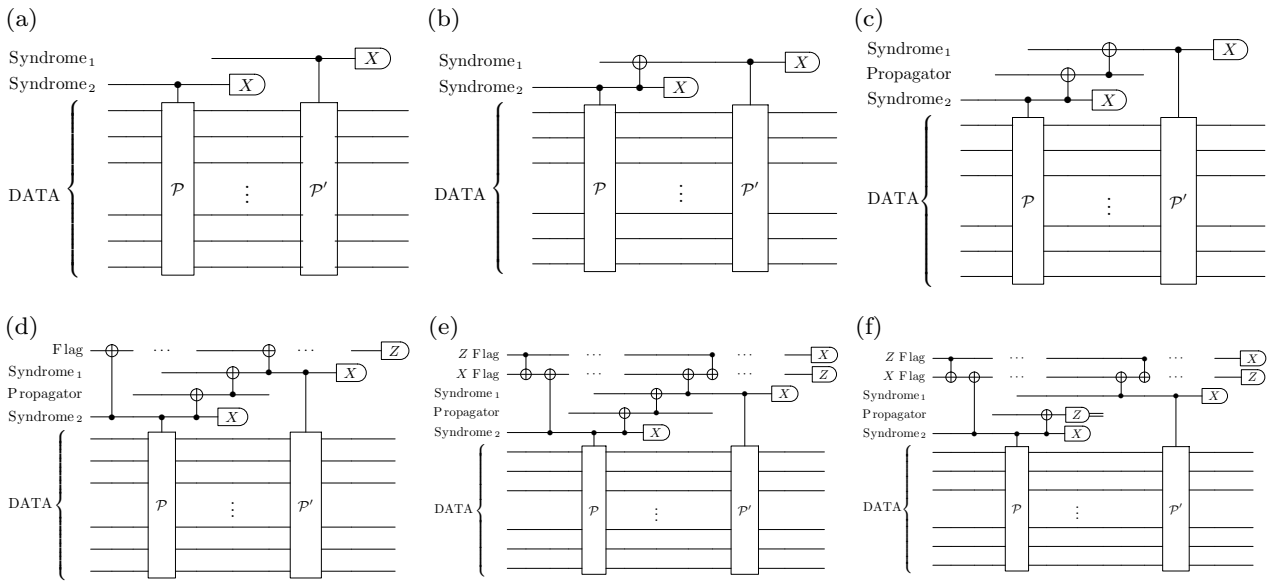

FIG. 7. The syndrome extraction of $\mathcal{P}$ and $\mathcal{P}'$ (where $\mathcal{P}$ and $\mathcal{P}'$ are multiqubit Pauli operators). (a) Conventional syndrome extraction. (b) Connected syndrome extraction. (c) Connected syndrome extraction with a propagator qubit. (d) Connected syndrome extraction with an example flag qubit. (e) Syndrome extraction with an added Z-type flag. (f) Syndrome extraction using a virtual connection.

$(t+1)^2$, so that if at most t errors occur, we are guaranteed to see the syndrome corresponding to a fault-tolerant correction appearing $t+1$ times in a row [29,40]. We define the total weight W as $\sum_{i=1}^{L} w_i$. For convenience, we assume that $w_i = w$ for all i, so that the total weight is just Lw. This is not essential for the construction, however, and Lw can be replaced by W if desired in the following discussion. Connecting all of these extractions together in the form outlined above leads to a circuit with sW distinct locations that need to be identified by the flag gadgets. Our construction in Sec. V B holds and we can identify up to t errors distributed between these sW locations using only $(2t+1)t\left\lceil \log_2(sW+1)\right\rceil$ flag qubits.

To ensure that one Z error on a flag qubit cannot cause multiple measurement errors in the connected setting, we employ precisely the same solution that we used to ensure that errors on the syndrome qubit do not cause high-weight data errors. We wish to track errors on each flag such that any Z error causes at most one unflagged measurement error, meaning that if we measure L stabilizers s times, there are $sL-1$ logically distinct locations on each flag qubit. If we construct a new flag gadget for each original flag qubit to flag the $L-1$ distinct locations, fault tolerance is restored, at the cost of $(2t+1)t\left\lceil \log_2(sL)\right\rceil$ additional Z-type flags per flag qubit. This is illustrated in Fig. 7(e) and its fault tolerance is treated more carefully in Appendix C.

B. Error propagation and time ordering

We now wish to make a few observations. First, $sL-1$ is an upper bound on the number of logically distinct locations per flag qubit. This is because a Z error occurs before the measurement of stabilizer i propagates trivially from a flag qubit to syndrome qubit i if the flag qubit is connected to syndrome qubit i by an even number of CNOTs. Additionally, the Z-type flags can benefit from the same numerical search demonstrated for small flag gadgets in Sec. IX. Finally, since we only measure flag qubits after all rounds of syndrome extraction, we allow low-weight errors on the syndrome qubits to propagate to high-weight errors on the data. This means that some of the syndrome bits measured will be incorrect, since we do not measure the flags until all syndromes have been extracted. However, since the syndrome-extraction circuit is a Clifford circuit, after measuring the flags we can track the propagation of errors classically and correct any syndrome bits necessary before applying the correction implied by the (updated) syndromes. This also holds for the measurement errors caused by Z errors on the flag qubits, caught by the Z-type flags. Additionally, since we do not measure flags until after all syndromes have been extracted, we cannot stop early if we see $t+1$ identical syndromes in a row—instead, we have to continue measuring syndromes for a full s repetitions.

This method of connecting syndrome qubits naively increases the time required for a round of syndrome

extraction, since the connections induce a time ordering that disallows extracting syndrome bits in parallel. However, the time ordering is not necessary. Instead of using the propagator ancilla to connect syndrome qubits, we can measure the ancilla in the Z basis and then adjust the flag measurements according to the measurement result (simulating the propagation of the possible X error through the CNOTs classically). Using this virtual connection removes any time ordering and allows for a flag circuit based on our construction to protect multiple syndrome extractions acting in parallel [see Fig. 7(f)]. Note that removing the time ordering means that there are exactly $sL-1$ locations per flag for each Z-type flag gadget to identify.

VIII. RESOURCE ANALYSIS

Consider a code defined by L (independent) stabilizer generators, each with weight w. Assuming that qubits cannot be reset (for a discussion on when this constraint is relaxed, see Sec. X), the total number of qubits required to do s rounds of syndrome extraction using Shor error correction or similar methods is linear in Lsw. In contrast, the number of qubits required by our construction for fault-tolerant extraction is given by $O(t^2\log_2(w))$ in the case of separate syndromes, or $(2t+1)t\left\lceil\log_2(Lsw+1)\right\rceil(2t+1)t\left\lceil\log_2(Ls)\right\rceil+2Ls-1$ in the case of connected syndromes (Sec. VII), where $2t+1$ is the distance of the code. As discussed in Sec. V D, the construction clearly provides an advantage when the weight of the stabilizer w is large. In this section, we show that by connecting multiple syndromes, similar advantage is gained for codes with small weight w, such as qLDPC codes.

It is important to emphasize that one main advantage of our construction is that we only need to prepare simple single-qubit states, in contrast to Shor, Steane, or Knill error correction, in which access to highly entangled fault-tolerantly prepared ancilla states is assumed. In particular, in these schemes, the required resources to produce such an entangled state can significantly increase with the distance of the code $2t+1$. Throughout all comparisons in this section, we have neglected the cost of preparing the ancilla states. Including the cost of fault-tolerantly preparing the ancilla states would increase the resource savings offered by our construction. In fact, preparation of the code states used in Steane or Knill error correction is often performed using Shor error correction [40], so our comparisons to Shor error correction can be seen as providing generous lower bounds on comparisons to Steane or Knill error correction.

A. Applicability to qLDPC codes

Any family of codes with parameters that result in a small ratio $t^4\log_2 Lsw\log_2 Ls/Lsw$ can benefit from our construction. One important family to consider is qLDPC codes [28], where the weight of each stabilizer is a constant independent of the number of qubits n, i.e., $w=O(1)$. Note that this constraint also fixes the number of independent stabilizers to be $L=\Theta(n)$, since each qubit must be included in at least one stabilizer for the code to be able to correct any single error. For this family of codes, we make the following observation.

Remark. For any code with $Lw=\Omega(t^{2+\varepsilon})$ for $\varepsilon>0$, our construction requires exponentially close to two qubits per stabilizer measurement.

To see this, note that for Shor-style syndrome extraction, we have $s=\Theta(t^2)$ and therefore the ratio scales as $\Theta\left(\log_2 Lt^2\log_2 L/\mathrm{poly}(L)\right)$, which is bounded above by $\Theta\left((\log_2 n)^2/\mathrm{poly}(n)\right)$ for any error-correcting qLDPC code (we suppress constant factors in all big-Θ expressions). Code families in which this condition is satisfied include qLDPC codes, for which the distance grows more slowly than $\sqrt{n}$, such as hyperbolic surface codes [45,46].

A regime in which our scheme does not provide an advantage for qLDPC codes is when they both satisfy the single-shot property ($s=O(1)$) [47–49] and have linear distance ($t=\Theta(n)$). Although we focus on qLDPC codes in this section, if w grows with n, our observation still holds, i.e., $Lw=\Omega(t^{2+\varepsilon})$ implies that asymptotically the scheme requires close to two qubits per stabilizer measurement.

B. Small-size codes

We now examine how our analytical construction performs on small-size codes. Although our choice of code to design flag qubits achieves the logarithmic scaling in the total stabilizer weight that we aimed for, the magnitude of the prefactors for this specific choice makes it impractical to connect syndrome qubits for small codes. However, only for distance-3 codes, there exists another construction that scales logarithmically in stabilizer weight and can be applied to arbitrary weight stabilizers [9]. Applying our framework to this construction, we can demonstrate the effectiveness of our method to connect flagged syndrome qubits.

Similarly to Sec. VIII, we consider a code defined by L stabilizer generators, each with weight w, which are each measured s times. The construction by Prabhu *et al.* [9] uses $\left\lceil\log_2 w\right\rceil$ flag qubits to fault-tolerantly measure a single stabilizer of a distance-3 code. Applying our construction for connecting syndromes yields a total ancilla cost of $\left\lceil\log_2 Lsw\right\rceil\left\lceil\log_2 Ls\right\rceil+2Ls-1$. Note that we apply our Z-type flag construction using the same flag construction as the (X-type) stabilizer flags to achieve this scaling. With the smaller prefactors, we can clearly see the advantage of connecting syndrome qubits in Table II. In Table III, we compare our construction to Shor-style

TABLE II. A comparison of the number of extra qubits needed to do syndrome extraction fault tolerant to distance 3 with and without flag qubits for regimes involving few qubits. The first row is the quantum Hamming code [50], while the second is the quantum Golay code [50] (treated as a distance-3 code). Shor EC means Shor-style error correction.

n	s	w	L	Shor EC Lsw	Connected syndrome extraction $\lceil\log_2(Lsw)\rceil\lceil\log_2(Ls)\rceil + 2Ls - 1$	Prabhu *et al.* [9] $Ls\lceil\log_2(w)+1\rceil$
31	4	16	10	1240	139	200
23	4	8	22	2024	245	352

syndrome extraction for distances where other flag constructions are not applicable. The numbers provided give a lower bound on w and L to achieve an advantage when $w \approx L$. Similar advantages can be observed if either of w or L is much larger than the other. For instance, taking four copies of the $[\![5,1,3]\!]$ code concatenated with itself, considered as a $[\![100,4,5]\!]$ code, yields 96 stabilizers with average weight $20/3$ and gives an approximate 630-qubit advantage over Shor's scheme.

It also is important to note that it is not always necessary to measure all of the stabilizers $(t+1)^2$ times in Shor-style error correction [29,30,53]. Given that our construction is agnostic to the form of the measured stabilizers, we can directly apply it to an error-correction procedure that uses fewer repetitions, such as a recent scheme for short syndrome-measurement sequences [31]. In our resource estimates in this section, we have considered using $(t+1)^2$ rounds of syndrome extraction.

IX. COMPUTER SIMULATIONS

So far, we have focused on analytical asymptotics but with the framework provided by Sec. III, we can do computer-aided searches for fault-tolerant flag-gadget constructions. In this section, we search for stabilizer weights where it is possible to use fewer than $2t+1$ repetitions of the BCH-code parity-check matrix as well as very small examples for distance-5 codes. The code we have developed to perform these simulations is publicly available on LoboGit.

The construction given in Sec. V B only gives an upper bound on the number of repetitions required; it does not always take $2t+1$ repetitions in order to achieve fault tolerance for up to t errors. In particular, for measuring a weight-15 syndrome, fault tolerance is achieved using only $t+1$ or $t+2$ repetitions of the BCH code, as opposed to $2t+1$ (see Table IV). This shows that certain parity-check matrices have properties that reduce the negative effects of columns produced by transitioning from one column of the check matrix to another.

Similarly, numerical simulations show that three repetitions of the two-error-correcting BCH code on 13 bits are sufficient to deduce fault-tolerant corrections for extracting a weight-13 stabilizer of a distance-5 code. In Appendix D, we argue that we can double the weight of the stabilizer by making a small modification to the correction procedure but no change to the flag qubit pattern. This is not guaranteed to hold when using three repetitions of the parity-check matrix instead of $2t+1=5$ but simulations show that in this case, replacing the weight-13 stabilizer with a weight-26 stabilizer and using three repetitions of the BCH code on 13 bits does indeed allow for fault-tolerant corrections. This is significant since the parity-check matrix for the BCH code on 13 bits has eight rows. This means that we use 24 flag qubits and one syndrome qubit for a total cost of 25 qubits to extract a weight-26 stabilizer, which is less costly than Shor error correction (which would use 26 qubits). Since in this example, $t=2$, previous flag-qubit constructions do not apply (assuming slow reset).

Numerical simulations also show that it is possible to fault-tolerantly extract an arbitrary weight-5 stabilizer for a distance-5 code using only two flag qubits, for a total cost of three ancilla qubits. The circuit to do so is shown in Fig. 8, while the correction rules are shown in Table VI.

TABLE III. A comparison of the number of extra qubits needed to do syndrome extraction fault-tolerantly to distances 5 and 7 with and without flag qubits. The first row is taken from the $[\![105,61,5]\!]$ quantum BCH code, while the second is from the $[\![119,23,7]\!]$ quantum BCH code [51]. In both examples, w is taken to be $\lfloor n/2\rfloor$, the average weight of the stabilizers produced via the CSS construction. The third row is the $[\![512,501,3]\!]$ quantum Hamming code [50] concatenated with the $[\![5,1,3]\!]$ perfect code [52] to produce a $[\![2560,501,9]\!]$ code. In this example, the average stabilizer weight w is slightly higher than 10 and the resources are computed as such.

t	s	w	L	Shor EC Lsw	Connected syndrome extraction $(2t+1)^2t^2\lceil\log_2(Lsw+1)\rceil \times \lceil\log_2(Ls)\rceil + 2Ls - 1$
2	9	52	44	20 592	14 291
3	16	59	96	90 624	85 538
4	25	10	2059	518 400	496 933

TABLE IV. The fault-tolerance results for $w = 15$, $d = 2t + 1$, and r repetitions of the parity-check matrix for a distance-d BCH code. "N" indicates non-fault-tolerant parameters, while "Y" indicates fault-tolerant parameters. These results are obtained using the numerical methods described in Sec. IX.

t	r			
	2	3	4	5
1	Y	Y	Y	Y
2	N	Y	Y	Y
3	N	N	N	Y

The circuit and correction rules have both been obtained by brute force. This method yields a two-qubit advantage over Shor syndrome extraction and a three-qubit advantage over the alternative flag-gadget construction given for stabilizer codes of any distance [8]. Note that, in contrast to the definition of e_s given in Sec. III, we must consider X errors on the syndrome qubit that occur before the first CNOT. This has been accounted for in the simulations.

These examples are summarized in Table V.

In general, this method of syndrome extraction yields performance improvements for any code with relatively high-weight syndromes (or high total weight, as in Sec. VII). One example of such a family of codes is produced when choosing the set of stabilizers measured according to some classical code [29,30]. Generally, the weight of the stabilizers measured is much greater than the weight of the stabilizer generators—e.g., a distance-d (for d sufficiently large) rotated surface code equipped with the $[16, *, d]$ BCH code measures stabilizers of weight approximately 28. Other examples can be obtained from code concatenation, which leads to high-weight stabilizers being measured after only a few concatenations.

X. QUBIT RESET

Although we have focused on the regime in which qubit reset is slow or impractical, our construction can also provide a reduction in the number of qubits necessary for fault-tolerant error correction when qubit reset is practical.

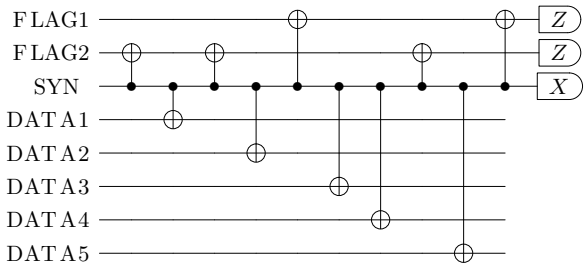

FIG. 8. A flag gadget for fault-tolerantly extracting a weight-5 stabilizer.

TABLE V. The total number of ancilla qubits (including the syndrome qubit) sufficient to measure a weight-w stabilizer of a distance-5 code fault tolerantly and the number of repetitions of the BCH parity-check matrix used. Note that the example for a weight-5 stabilizer does not use the parity-check-matrix construction.

w	5	15	26
Qubits	3	25	25
Repetitions	None	3	3

In general, as qubit measurement and reset gets faster, our construction provides fewer reductions.

Suppose that qubits can be reset in parallel in time τ and single-qubit measurements can be performed in time μ (where a CNOT takes unit time). Note that if qubits cannot be reset (in parallel), the method of connected syndrome extraction presented in Sec. VII is more useful than if qubits can be reset (in parallel). In Table II, we implicitly assume that τ is arbitrarily large.

However, it is of course possible to reset qubits. We perform resource estimation for a round of error correction for Shor syndrome extraction and flagged (connected) syndrome extraction. The maximum number of qubits required for flagged syndrome extraction is given simply by $(2t+1)t\lceil \log_2(sW+1)\rceil + 2Ls - 1$ (or $\lceil \log_2(Lsw)\rceil \lceil \log_2(Ls)\rceil + 2Ls - 1$ for distance 3) as in Sec. VII. To estimate the resources required for Shor syndrome extraction, we use Algorithm 1, which provides a lower bound on the number of qubits necessary when doing a round of error correction as fast as possible.

To provide a concrete example, we consider the recent experiments on superconducting hardware [54], where the measurement and reset times are approximately 500 ns and 160 ns, respectively. With a CNOT time of approximately 13 ns [55], this corresponds to $\tau = 38.5$ and $\mu = 12.3$. Resource estimation shows that for these values of τ and μ, our construction yields a qubit advantage over Shor EC for the $[\![9, 1, 3]\!]$ Shor code but not for the $[\![7, 1, 3]\!]$ color code. This aligns with the fact that the stabilizers that need to be protected in the Shor code (the weight-6 X-type stabilizers) are higher weight than the stabilizers for the color code (which are weight-4), meaning that Shor syndrome

TABLE VI. One choice of fault-tolerant correction rules for the circuit presented in Fig. 8. Note that there are other choices for the fault-tolerant correction rules.

FLAG1, FLAG2 measurements	Correction on DATA
$+1, +1$	I
$+1, -1$	X_1
$-1, +1$	I
$-1, -1$	$X_1X_2X_3$

ALGORITHM 1. An outline of our strategy for estimating resources required for Shor EC.

input a set of generators G and a stack of available qubits p
  used $\leftarrow \emptyset$
  busy $\leftarrow \emptyset$
  for each generator $g \in G$ **do**
    move any qubits which have become non-busy from busy to top of p
    selected $\leftarrow$ first $|g|$ qubits from p
    used $\leftarrow$ used $\cup$ selected
    busy $\leftarrow$ selected and times selected qubits will become available again
  end for
  return |used|

extraction requires fewer ancilla qubits to be used simultaneously when extracting the syndromes for the color code. This again shows that our construction is especially useful when measuring high-weight stabilizers.

Other recent experiments on ion-trap hardware [56] give measurement times corresponding to $\mu = 13$. Although reset times are not given, we assume a comparable time scale and set $\tau = 13$ for resource estimation. In this regime, where CNOTs are relatively slow, we begin to see qubit advantages for our construction on a distance-3 code when stabilizers are weight approximately 13.

It may be possible to achieve qubit advantages using our construction even in the case that τ and μ are relatively small and when measuring low-weight stabilizers by using fewer than $2t+1$ repetitions of the parity-check matrix. The numerical results in Sec. IX show that it is not always necessary to use $2t+1$ repetitions of the parity-check matrix and it is often sufficient to use $t+1$ repetitions.

XI. CONCLUSIONS

In this work, we have developed a framework to design flag gadgets based on classical codes. This framework allows for enough freedom to achieve logarithmic cost scaling with the appropriate choice of code (e.g., the BCH code), to perform fault-tolerant syndrome measurement of any general quantum code.

To maximize the gain from this exponential saving, we have proposed methods to fault-tolerantly measure multiple stabilizers using a single gadget. We have proposed several small examples of the constructions using a computer-assisted search that can be appropriate for near-term experiments on small quantum computers. We have discussed the overhead when slow qubit reset is available and proposed several decoding strategies.

This work leaves several questions open. In Sec. V B, we have assumed that columns introduced by the unfolding procedure in Sec. V A were completely arbitrary, and hence used $2t+1$ repetitions of the parity-check matrix to guarantee that they would not cause non-fault-tolerant corrections. However, it may be possible to characterize these columns in a way that allows for fewer repetitions (or, in the extreme case, to remove these columns altogether). Incorporating other physically relevant constraints, such as geometric locality of interactions, is another direction to pursue.

In addition, our construction uses a two-step procedure, in which first the propagated errors from the syndrome onto data are corrected and then the syndrome bits are used to do error correction according to the quantum code. However, one can integrate the information from the flag qubits and the syndrome bits in order to directly infer the correction to apply to the data, possibly using fewer flag qubits than the two-step version, and one can add structure between multiple syndrome qubits (as in, e.g., Ref. [11]).

Finally, throughout the analytical portion of this work, we have used the parity-check matrix for the BCH code because of the favorable scaling that it provides. However, our framework also allows for the systematic design of flag gadgets when the number of ancilla qubits or the number of measurements is only one of many goals. By considering different classical codes, one can easily incorporate constraints based on CNOT count, connectivity, or other physically relevant parameters. Additionally, in Sec. IX, we search over small flag gadgets in a brute-force way and show that even smaller flag gadgets exist than those produced by considering the BCH code.

The code that we have developed to perform these simulations is publicly available on LoboGit [57].

ACKNOWLEDGMENTS

This material is based upon work supported by the U.S. Department of Energy, Office of Science, National Quantum Information Science Research Centers, Quantum Systems Accelerator. Additional support by the National Science Foundation (NSF) CAREER Award No. CCF-2237356 and NSF Grant No. CCF-1954960 is acknowledged.

APPENDIX A: BCH CODES

Given m such that $2^m \geq w$, the BCH code correcting up to $t \leq w/2$ errors out of w locations uses at most mt parity checks. The parity-check matrix is constructed using a *primitive element* α of $GF(2^m)$ (i.e., an element α of the finite field with 2^m elements such that every nonzero element of $GF(2^m)$ can be written as α^i for some i).

The (redundant) parity-check matrix H' is defined as

$$H' := \begin{pmatrix} 1 & \alpha & \alpha^2 & \dots & \alpha^{w-1} \\ 1 & \alpha^2 & (\alpha^2)^2 & \dots & (\alpha^3)^{w-1} \\ & & & \vdots & \\ 1 & \alpha^{2t} & (\alpha^{2t})^2 & \dots & (\alpha^{2t})^{w-1} \end{pmatrix}.$$

It is easy enough to see that $\sum(\alpha^i)^2 = 0$ if and only if $\sum \alpha^i = 0$ since $GF(2^m)$ is a field of characteristic two—consequently, every other row of H' is redundant. We can remove every other row so that $H_{ij} = (\alpha^{2j+1})^{i+1}$, for i and j starting at zero.

Representing elements of $GF(2^m)$ as bit strings of length m produces a parity-check matrix with tm rows. Since $2^m \geq w$, this corresponds to $t\lceil \log_2 w \rceil$ rows (or parity checks).

We now show that H' yields enough information to correct up to t errors. Recall that this is equivalent to any set of up to $2t$ columns of H' being linearly independent. Suppose for the sake of contradiction that $H'v = 0$ for some v such that $|v| \leq 2t$, where we use the full (redundant) matrix H'. Then,

$$\begin{pmatrix} \alpha^{j_1} & \alpha^{j_2} & \dots & \alpha^{j_{|v|}} \\ (\alpha^{j_1})^2 & (\alpha^{j_2})^2 & \dots & (\alpha^{j_{|v|}})^2 \\ & & \vdots & \\ (\alpha^{j_1})^{2t} & (\alpha^{j_2})^{2t} & \dots & (\alpha^{j_{|v|}})^{2t} \end{pmatrix} \begin{pmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{pmatrix} = 0$$

for all j_i such that $v_{j_i} = 1$. Since $2t \leq w$, we can truncate this equation to the first $2t$ rows so that the matrix on the left-hand side is square. Factoring out a power of α^{j_i} from the j_i column shows that this equation reduces to the determinant of a Vandermonde matrix being equal to zero, which is a contradiction. Since H has the same column-independence properties as H', this concludes the proof that H is a parity-check matrix for a t-error-correcting code. Note that it is possible that the true distance of the code given by $\ker H$ is greater than $2t + 1$.

APPENDIX B: ORDERING COLUMNS OF PARITY-CHECK MATRIX

In Sec. V A, we have required that the parity-check matrix in question admits an ordering of its columns such that, when unfolded, there are no zero columns between two nonzero columns. Here, we provide an explicit ordering satisfying this constraint.

Lemma 6. For any arbitrary binary matrix H that does not contain the zero vector as a column, there exists a matrix H' obtained by reordering columns of H such that $\forall_{k,i} : H'_i \oplus [H'_i \oplus H'_{i-1}][k] \neq 0$.

Proof. Obtain H' from H by sorting the columns of H in descending order from left to right, where we interpret each column as a binary integer with the highest significance bit at the bottom.

Suppose for the sake of contradiction that there exist some i and k such that $H'_i \oplus [H'_i \oplus H'_{i-1}][k] = 0$. Let the index of the kth nonzero element of $H'_i \oplus H'_{i-1}$ be called j.

Since $H'_i \oplus [H'_i \oplus H'_{i-1}][k] = 0$, we have that H'_i is zero after index j. By the ordering assumption, this implies that H'_{i-1} is zero after index j as well.

Similarly, since the first j elements of $H'_i \oplus [H'_i \oplus H'_{i-1}]$ are equal to the first j elements of $H'_i \oplus H'_i \oplus [H'_{i-1}] = H'_{i-1}$, we have that the first j elements of H'_{i-1} are zero. So H'_{i-1} is the zero vector. This contradicts our assumption on H.

Therefore, sorting the columns in descending order yields an ordering that does not produce the zero vector as an intermediate column. ■

APPENDIX C: FAULT TOLERANCE OF FLAGGED CONNECTED SYNDROMES

We prove that the flagged connected syndrome procedure given in Sec. VII is fault tolerant. First, we more precisely describe the construction proposed. As in Sec. II, we identify certain (sets of) locations of the circuit with bits of a classical codeword. Here, bit i of classical codeword j corresponds to all locations after the first CNOT on flag qubit j from which a Z error would propagate to syndrome qubit i and before the last such CNOT. We can see that each classical codeword will be of length at most Ls. Having identified locations in the circuit with a classical codeword, we now construct Z-type flag gadgets corresponding to the parity-check matrix of the BCH code on at most Ls bits, one gadget for each classical codeword, as in Sec. II. These flag gadgets have the same form as X-type flag gadgets, except that a CNOT with control i and target j now has control j and target i and flag qubits are prepared in a $|+\rangle$ state and measured in the X basis. This is illustrated in Fig. 7(e).

Lemma 7. The above construction is fault tolerant.

Proof. Consider first the performance of the X-type flag gadgets, without including the Z-type gadgets. If no Z errors occur on the X-type flag qubits, syndrome extraction is fault tolerant. This is because if $k \leq t$ errors are suffered on the syndromes, we identify a correction that leaves at most k errors on the data. This is equivalent to perfectly identifying the error on the data that has propagated from the syndrome and also suffering k unknown errors on the data. Crucially, the effect that the errors left on the data until the flags are measured have on the later stabilizer measurements can be tracked and accounted for. So flagging connected syndromes and only applying the corrections implied by the flags at the end does not impact fault tolerance.

Similarly, if a Z error on a flag qubit propagates to at most one syndrome qubit, syndrome extraction is fault tolerant, since flag qubits are measured in the Z basis (i.e., Z errors on the flag qubits only propagate to one (measurement) error in this case). So it is enough to show that adding Z-type flag gadgets to each X-type flag qubit ensures this propagation of Z errors and is itself fault tolerant.

As in Sec. V B, our construction allows us to identify the location of up $k \leq t$ flipped bits on the classical codeword (i.e., regions with odd-parity Z errors) up to some imprecision. This imprecision means that we may identify an error on bit i as an error on bit $i+1$. However, a correction exists that leads to the Z error propagating to at most one syndrome qubit in both cases (namely, assume that there was a Z error at the beginning of location $i+1$ and correct accordingly). This shows that the Z-type flag gadgets allow us to ignore the effect of Z-type errors on the X-type flag qubits.

Finally, we must show that errors on the Z-type flags do not propagate to high-weight errors. This is clear, since each Z-type flag is connected to exactly one X-type flag qubit, which is measured in the Z basis. Therefore, only X errors affect the measurement result. Since X errors do not affect the measurement result of the Z-type flags (since they are measured in the X basis), one error on the Z-type flags causes at most one measurement error. ■

APPENDIX D: REDUCING EFFECTIVE STABILIZER WEIGHT BY $\frac{1}{2}$

In Ref. [9], the authors make the observation that placing two data CNOTs in each region uniquely identified by a flag pattern is sufficient to ensure fault tolerance. To show this, the correction rule is changed slightly so that instead of applying a correction anywhere in the flagged region, the correction must be applied between the pair of data CNOTs. We can apply this optimization to our construction as well, when we repeat the parity-check matrix at least $2t+1$ times. It might also be possible to apply this optimization when repeating fewer than $2t+1$ times but proving this requires stronger characterizations of the transition columns.

In the case in which we use $(t+1)^2$ repetitions (as in Sec. VI) and apply the correction associated with the most frequent flag subpattern, the modification is obvious. If the BCH decoder gives $\{e_i\}$ as the correction, where e_i corresponds to applying a correction in a certain region, it is enough to simply assume that the correction falls between the pair of data CNOTs in that region.

The procedure is similar for the case in which we use fewer repetitions (Sec. V B). For a given flag pattern P, the algorithm outlined produces a correction of minimal weight that has the same flag pattern. The only difference when we add a data CNOT as a pair to each previous data CNOT is that the set of minimal-weight corrections may include non-fault-tolerant corrections. However, the correction corresponding to the case in which each component of the correction is between the new pair of data CNOTs is fault tolerant. So we just need some rules for identifying this correction. It is clear that if every component correction comes after an odd number of data CNOTs, the total correction will consist of corrections falling between pairs of data CNOTs and will hence be fault tolerant. Such a correction will always exist in the set of minimal-weight corrections with syndrome P, since shifting a component correction past at most one data CNOT within the same region of flags does not change the weight or the syndrome.

It is important to note that it may not be possible to replace each data CNOT (qubit) with two data CNOTs (qubits) in general. It is only possible in our construction because of the fact that errors with the same flag pattern differ in a very specific way—namely, that if two errors e_1, e_2 have the same flag pattern, each physical error on the syndrome can be paired with another physical error on the syndrome from either e_1 or e_2 with at most one data CNOT in between (or two data CNOTs in between when replacing each data CNOT by two data CNOTs), as proven in Lemma 3.

[1] D. P. DiVincenzo and P. W. Shor, Fault-tolerant error correction with efficient quantum codes, Phys. Rev. Lett. **77**, 3260 (1996).

[2] D. P. DiVincenzo and P. W. Shor, Fault-tolerant error correction with efficient quantum codes, Phys. Rev. Lett. **77**, 3260 (1996).

[3] A. M. Steane, Active stabilization, quantum computation, and quantum state synthesis, Phys. Rev. Lett. **78**, 2252 (1997).

[4] S. Huang and K. R. Brown, Between Shor and Steane: A unifying construction for measuring error syndromes, Phys. Rev. Lett. **127**, 090505 (2021).

[5] S. Huang and K. R. Brown, Constructions for measuring error syndromes in Calderbank-Shor-Steane codes between Shor and Steane methods, Phys. Rev. A **104**, 022429 (2021).

[6] R. Chao and B. W. Reichardt, Quantum error correction with only two extra qubits, Phys. Rev. Lett. **121**, 050502 (2018).

[7] R. Chao and B. W. Reichardt, Fault-tolerant quantum computation with few qubits, npj Quantum Inf. **4**, 42 (2018).

[8] R. Chao and B. W. Reichardt, Flag fault-tolerant error correction for any stabilizer code, PRX Quantum **1**, 010302 (2020).

[9] P. Prabhu and B. W. Reichardt, Fault-tolerant syndrome extraction and cat state preparation with fewer qubits, Quantum **7**, 1154 (2023).

[10] C. Chamberland and M. E. Beverland, Flag fault-tolerant error correction with arbitrary distance codes, Quantum **2**, 53 (2018).

[11] B. W. Reichardt, Fault-tolerant quantum error correction for Steane's seven-qubit color code with few or no extra qubits, Quantum Sci. Technol. **6**, 015007 (2020).

[12] C. Chamberland, G. Zhu, T. J. Yoder, J. B. Hertzberg, and A. W. Cross, Topological and subsystem codes on low-degree graphs with flag qubits, Phys. Rev. X **10**, 011022 (2020).

[13] M. Gutiérrez, M. Müller, and A. Bermúdez, Transversality and lattice surgery: Exploring realistic routes toward

coupled logical qubits with trapped-ion quantum processors, Phys. Rev. A **99**, 022330 (2019).
[14] T. Tansuwannont, C. Chamberland, and D. Leung, Flag fault-tolerant error correction, measurement, and quantum computation for cyclic Calderbank-Shor-Steane codes, Phys. Rev. A **101**, 012342 (2020).
[15] C. Chamberland, A. Kubica, T. J. Yoder, and G. Zhu, Triangular color codes on trivalent graphs with flag qubits, New J. Phys. **22**, 023019 (2020).
[16] T. Tansuwannont and D. Leung, Achieving fault tolerance on capped color codes with few ancillas, PRX Quantum **3**, 030322 (2022).
[17] T. Tansuwannont and D. Leung, Fault-tolerant quantum error correction using error weight parities, Phys. Rev. A **104**, 042410 (2021b).
[18] Y. Shi, C. Chamberland, and A. Cross, Fault-tolerant preparation of approximate GKP states, New J. Phys. **21**, 093007 (2019).
[19] C. Chamberland and K. Noh, Very low overhead fault-tolerant magic state preparation using redundant ancilla encoding and flag qubits, npj Quantum Inf. **6**, 91 (2020).
[20] C. Chamberland and A. W. Cross, Fault-tolerant magic state preparation with flag qubits, Quantum **3**, 143 (2019).
[21] A. Rodriguez-Blanco, A. Bermudez, M. Müller, and F. Shahandeh, Efficient and robust certification of genuine multipartite entanglement in noisy quantum error correction circuits, PRX Quantum **2**, 020304 (2021).
[22] D. M. Debroy and K. R. Brown, Extended flag gadgets for low-overhead circuit verification, Phys. Rev. A **102**, 052409 (2020).
[23] C. Ryan-Anderson, J. G. Bohnet, K. Lee, D. Gresh, A. Hankin, J. P. Gaebler, D. Francois, A. Chernoguzov, D. Lucchetti, N. C. Brown, *et al.*, Realization of real-time fault-tolerant quantum error correction, Phys. Rev. X **11**, 041058 (2021).
[24] Quantum Information and Computation (Rinton Press, 2018), Vol. 18, http://dx.doi.org/10.26421/qic18.11-12.
[25] J. Hilder, D. Pijn, O. Onishchenko, A. Stahl, M. Orth, B. Lekitsch, A. Rodriguez-Blanco, M. Müller, F. Schmidt-Kaler, and U. G. Poschinger, Fault-tolerant parity readout on a shuttling-based trapped-ion quantum computer, Phys. Rev. X **12**, 011032 (2022).
[26] M. H. Abobeih, Y. Wang, J. Randall, S. J. H. Loenen, C. E. Bradley, M. Markham, D. J. Twitchen, B. M. Terhal, and T. H. Taminiau, Fault-tolerant operation of a logical qubit in a diamond quantum processor, Nature **606**, 884 (2022).
[27] L. Postler, S. Heußen, I. Pogorelov, M. Rispler, T. Feldker, M. Meth, C. D. Marciniak, R. Stricker, M. Ringbauer, R. Blatt, *et al*, Demonstration of fault-tolerant universal quantum gate operations, Nature **605**, 675 (2022).
[28] N. P. Breuckmann and J. N. Eberhardt, Quantum low-density parity-check codes, PRX Quantum **2**, 040101 (2021).
[29] N. Delfosse, B. W. Reichardt, and K. M. Svore, Beyond single-shot fault-tolerant quantum error correction, IEEE Trans. Inf. Theory **68**, 287 (2022).
[30] A. Ashikhmin, C.-Y. Lai, and T. A. Brun, in *2014 IEEE International Symposium on Information Theory* (IEEE, Honolulu, HI, 2014), pp. 546–550.
[31] T. Tansuwannont and K. R. Brown, Adaptive syndrome measurements for Shor-style error correction, ArXiv:2208.05601.
[32] L. Lao and C. G. Almudever, Fault-tolerant quantum error correction on near-term quantum processors using flag and bridge qubits, Phys. Rev. A **101**, 032333 (2020).
[33] B. Pato, T. Tansuwannont, S. Huang, and K. R. Brown, Optimization tools for distance-preserving flag fault-tolerant error correction, PRX Quantum **5**, 020336 (2024).
[34] P.-H. Liou and C.-Y. Lai, Parallel syndrome extraction with shared flag qubits for Calderbank-Shor-Steane codes of distance three, Phys. Rev. A **107** (2023).
[35] I. Cong, H. Levine, A. Keesling, D. Bluvstein, S. T. Wang, and M. D. Lukin, Hardware-efficient, fault-tolerant quantum computation with Rydberg atoms, ArXiv:2105.13501.
[36] A. Bermudez, X. Xu, M. Gutiérrez, S. C. Benjamin, and M. Müller, Fault-tolerant protection of near-term trapped-ion topological qubits under realistic noise sources, Phys. Rev. A **100**, 062307 (2019).
[37] A. Jayashankar, M. D. H. Long, H. K. Ng, and P. Mandayam, Achieving fault tolerance against amplitude-damping noise, Phys. Rev. Res. **4**, 023034 (2022).
[38] P. Parrado-Rodríguez, C. Ryan-Anderson, A. Bermudez, and M. Müller, Crosstalk suppression for fault-tolerant quantum error correction with trapped ions, Quantum **5**, 487 (2021).
[39] P. Aliferis, D. Gottesman, and J. Preskill, Quantum accuracy threshold for concatenated distance-3 codes, ArXiv:quant-ph/0504218.
[40] P. W. Shor, Fault-tolerant quantum computation, ArXiv:quant-ph/9605011.
[41] R. C. Bose and D. K. Ray-Chaudhuri, On a class of error correcting binary group codes, Inf. Control **3**, 68 (1960).
[42] A. Hocquenghem, Codes correcteurs d'erreurs, Chiffres (Paris) **2**, 147 (1959).
[43] S. Bravyi, Subsystem codes with spatially local generators, Phys. Rev. A **83**, 012320 (2011).
[44] D. Schipani, M. Elia, and J. Rosenthal, in *2011 IEEE International Symposium on Information Theory Proceedings* (IEEE, St. Petersburg, Russia, 2011), pp. 835–839.
[45] N. P. Breuckmann and B. M. Terhal, Constructions and noise threshold of hyperbolic surface codes, IEEE Trans. Inf. Theory **62**, 3731 (2016).
[46] N. P. Breuckmann, C. Vuillot, E. Campbell, A. Krishna, and B. M. Terhal, Hyperbolic and semi-hyperbolic surface codes for quantum storage, Quantum Sci. Technol. **2**, 035007 (2017).
[47] H. Bombín, Single-shot fault-tolerant quantum error correction, Phys. Rev. X **5**, 031043 (2015).
[48] E. T. Campbell, A theory of single-shot error correction for adversarial noise, Quantum Sci. Technol. **4**, 025006 (2019).
[49] O. Fawzi, A. Grospellier, and A. Leverrier, in *2018 IEEE 59th Annual Symposium on Foundations of Computer Science (FOCS)* (IEEE, 2018), pp. 743–754.
[50] A. M. Steane, Simple quantum error-correcting codes, Phys. Rev. A **54**, 4741 (1996).
[51] M. Grassl and T. Beth, Quantum BCH codes, ArXiv:quant-ph/9910060.
[52] R. Laflamme, C. Miquel, J. P. Paz, and W. H. Zurek, Perfect quantum error correction code, ArXiv:quantph/9602019.

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

[54] R. Acharya, I. Aleiner, R. Allen, T. I. Andersen, M. Ansmann, F. Arute, K. Arya, A. Asfaw, J. Atalaya, R. Babbush, *et al.*, Suppressing quantum errors by scaling a surface code logical qubit, Nature **614**, 676 (2023).

[55] B. Foxen, C. Neill, A. Dunsworth, P. Roushan, B. Chiaro, A. Megrant, J. Kelly, Z. Chen, K. Satzinger, R. Barends, *et al.*, Demonstrating a continuous set of two-qubit gates for near-term quantum algorithms, Phys. Rev. Lett. **125**, 120504 (2020).

[56] L. Egan, D. M. Debroy, C. Noel, A. Risinger, D. Zhu, D. Biswas, M. Newman, M. Li, K. R. Brown, M. Cetina, *et al.*, Fault-tolerant control of an error-corrected qubit, Nature **598**, 281 (2021).

[57] B. Anker, Generalized flag qubit LoboGit repository, https://lobogit.unm.edu/banker/generalized-flag-qubit/.