


## Realization of Real-Time Fault-Tolerant Quantum Error Correction

C. Ryan-Anderson, J. G. Bohnet, K. Lee,<sup>\*</sup> D. Gresh, A. Hankin, J. P. Gaebler, D. Francois, A. Chernoguzov<sup>✉</sup>, D. Lucchetti, N. C. Brown<sup>✉</sup>, T. M. Gatterman, S. K. Halit<sup>✉</sup>, K. Gilmore<sup>✉</sup>, J. A. Gerber<sup>✉</sup>, B. Neyenhuis<sup>✉</sup>, D. Hayes, and R. P. Stutz<sup>✉</sup>

*Quantinuum, 303 South Technology Court, Broomfield, Colorado 80021, USA*

 (Received 10 August 2021; revised 24 November 2021; accepted 7 December 2021; published 23 December 2021; corrected 11 January 2022)

Correcting errors in real time is essential for reliable large-scale quantum computations. Realizing this high-level function requires a system capable of several low-level primitives, including single-qubit and two-qubit operations, midcircuit measurements of subsets of qubits, real-time processing of measurement outcomes, and the ability to condition subsequent gate operations on those measurements. In this work, we use a 10-qubit quantum charge-coupled device trapped-ion quantum computer to encode a single logical qubit using the  $[[7, 1, 3]]$  color code, first proposed by Steane [Phys. Rev. Lett. **77**, 793 (1996)]. The logical qubit is initialized into the eigenstates of three mutually unbiased bases using an encoding circuit, and we measure an average logical state preparation and measurement (SPAM) error of  $1.7(2) \times 10^{-3}$ , compared to the average physical SPAM error  $2.4(4) \times 10^{-3}$  of our qubits. We then perform multiple syndrome measurements on the encoded qubit, using a real-time decoder to determine any necessary corrections that are done either as software updates to the Pauli frame or as physically applied gates. Moreover, these procedures are done repeatedly while maintaining coherence, demonstrating a dynamically protected logical qubit memory. Additionally, we demonstrate non-Clifford qubit operations by encoding a  $\bar{T}|+\rangle_L$  magic state with an error rate below the threshold required for magic state distillation. Finally, we present system-level simulations that allow us to identify key hardware upgrades that may enable the system to reach the pseudothreshold.

DOI: [10.1103/PhysRevX.11.041058](https://doi.org/10.1103/PhysRevX.11.041058)

Subject Areas: Quantum Physics, Quantum Information

### I. INTRODUCTION

Large-scale quantum computers promise to solve classically intractable problems in areas such as quantum simulation, prime factorization, and others [1–7]. However, these complex quantum computations demand levels of precision that are currently unachievable due to imperfect control and noise in gate operations between physical qubits. In fact, it is unlikely that analog physical qubit control will ever reach the precision demanded by large-scale computations. Quantum error correction (QEC) [8–10] was the key ingredient to digitize quantum operations, making extremely low error rates possible in principle. QEC works by redundantly encoding quantum information into a protected subspace within a larger Hilbert space of many physical qubits. Using a polynomially scaling number of physical qubits, the probability of a

computation being corrupted can be suppressed exponentially, making arbitrarily precise quantum computation feasible.

Achieving efficient error suppression by a QEC code introduces several requirements for a quantum processor. Not only do the error rates of the underlying physical operations (initialization, unitary gates, and measurement) need to be below a certain threshold [11–13], but the quantum processor must interact with a classical computer in real time to diagnose and correct errors. These interactions between the quantum and classical processors need to be repeated several times in every step of a computation, defining new requirements in addition to the classic DiVincenzo criteria [14]. These new criteria include the ability to measure a subset of qubits with little impact on other qubits and the ability to classically process measurements and determine corrections faster than the system decoheres. Additionally, the implementation should be fault tolerant (FT) to at least the dominant physical layer errors in order to prevent physical errors from cascading and causing logical errors [15–17].

Several required elements of FT QEC of a single logical qubit have been demonstrated on a variety of quantum computing architectures: classical repetition codes [18–26], error detection codes [27–33], the 5-qubit code [34–36], the

<sup>\*</sup>Present address: Google Quantum AI, Santa Barbara, California, USA.

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

[[7,1,3]] color code [37,38], the Bacon-Shor-13 code [39], the 9-qubit Shor code [40,41], bosonic codes [42–46], as well as primitives utilizing cluster states [47,48]. However, a full demonstration of all necessary components for a FT implementation of a QEC code capable of repeatedly correcting all single-qubit errors has not yet been realized.

In this article, we report on realizing a FT implementation of the smallest instance of the color code [10,49–51]. Using a trapped-ion quantum charge-coupled device (QCCD) quantum computer [52,53], we encode, control, and repeatedly error correct a single logical qubit using ten physical qubits. On the physical layer, trapped-ion qubits use high-fidelity single- and two-qubit gates and midcircuit measurements and resets to execute the quantum circuits that fault tolerantly initialize the logical qubit, manipulate it via logical single-qubit Clifford gates, perform error syndrome measurements, and apply corrections. Importantly, the low cross talk during midcircuit measurements [20,54] enables ancilla measurements of the error syndromes without decohering the data qubits that encode the logical information. The error syndrome measurements are sent to a classical computer where real-time decoding and tracking of errors and corrections occur. Since all of this can be done with high fidelity and quickly compared to the dephasing rate of the physical qubits, the logical qubit can be *repeatedly* error corrected, a crucial feature of scalable quantum computing. We perform these operations with the six eigenstates of the logical Pauli operators and initialize a magic state for non-Clifford operations. These results demonstrate a universal set for quantum computation, with the notable exception of an entangling gate between two logical qubits, which requires more qubits than our system currently supports.

### A. Background

QEC codes are identified by three parameters  $[[n, k, d]]$ , where  $n$  is the number of physical qubits,  $k$  is the number of logical qubits the code admits, and  $d$  is the code distance, which is related to the minimum number of arbitrary single-qubit errors the code will correct,  $t = \lfloor d/2 \rfloor$ . The [[7, 1, 3]] stabilizer code, commonly referred to as the Steane code [10] and depicted in Fig. 1, is the smallest example of the topological color code [49]. We refer to it as a color code henceforth. The distance three code uses seven physical qubits to encode a single logical qubit and protects against all instances of a single physical qubit incurring an error ( $t = 1$ ). Advantages of topological stabilizer codes include requiring only local interactions, high error thresholds, and minimal error detection overhead [49,55–57]. The [[7, 1, 3]] color code has the added advantage of all single-qubit and two-qubit Clifford gates being transversal and naturally FT [58]. While the term “color code” generally refers to any member of the family of color codes, we only investigate one version here, so for simplicity, we drop the [[7, 1, 3]] notation and refer to the code we study as the

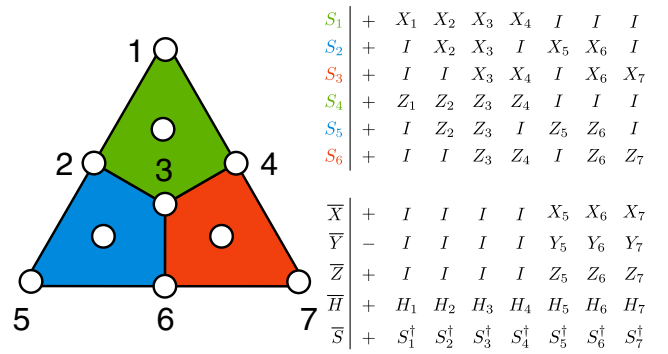


FIG. 1. The [[7, 1, 3]] color code. Physical qubits are indicated by white circles. The seven data qubits are on the vertices of the polygons, and three ancilla qubits for syndrome measurements at the center of each polygon. For each polygon, the four qubits at the vertices define both X-type and Z-type stabilizer measurements used in each error correction cycle. The stabilizer generators and logical operators are tensor products of single-qubit Paulis with support indicated by the physical qubit subscript index. For example, implementing the logical  $\bar{Z}$  operation is done with physical layer Z operations on qubits 5, 6, and 7. Likewise, measuring  $\bar{Z}$  is done by measuring  $Z_5 Z_6 Z_7$  where the logical qubit measurement outcome is the product of the three individual physical qubit measurement outcomes. Although not shown, Y-type stabilizers are generated by the X- and Z-type stabilizers listed here.

color code. We also note that we place over bars and  $L$  subscripts on logical qubit operators and states to distinguish them from physical qubit operators and states.

The color code, like all stabilizer codes, detects errors by measuring a commuting set of operators known as stabilizers (Fig. 1). The stabilizer measurements form an error syndrome which is processed by a classical decoding algorithm to determine a correction. This process must be done during the quantum computation to enable general, nontrivial logical computation, thus highlighting the need for systems to determine corrections during real time and not after the quantum computation has concluded. In this paper, we define a QEC cycle as the process of measuring syndromes, where syndromes are measured multiple times to account for measurement errors as well as decoding and updating corrections. Note that in general the determination of corrections may be delayed until an operation, such as a logical non-Clifford gate, requires a correction to be determined before the computation can proceed. In our experiments, however, we demonstrate the more demanding situation where the determination of corrections is not delayed but determined at the end of each QEC cycle.

Our quantum computer uses a Honeywell 2D surface electrode ion trap (Fig. 2) similar to the one described in Ref. [53]. The trap is loaded with ten  $^{171}\text{Yb}^+$  qubit ions and ten  $^{138}\text{Ba}^+$  sympathetic cooling ions. The color code is implemented using all ten available qubits (Fig. 1), where seven data qubits encode the logical state and three ancilla qubits perform syndrome measurements. The trap can

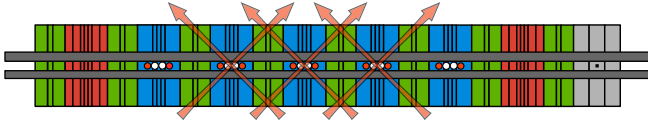


FIG. 2. The ion trap loaded with ten  $^{171}\text{Yb}^+$  qubit ions (red circles) and ten  $^{138}\text{Ba}^+$  coolant ions (white circles). The trap has different functional regions, or zones, with functions determined by the electrode geometry and laser beam configuration. Ion transport is used to arrange ions into zones for gates and measurements. The electrodes in red and blue denote regions that support transport operations, including linear transport, crystal split and combine, and physical swap. The green regions support linear transport and storage of ions between gating operations. The three regions where qubit initialization, gating, and measurement occur are marked by the crossing laser beams. In the gray region ions are loaded from an effusive atomic oven behind the trap using photoionization.

execute parallel quantum operations across three different zones, and quantum circuits are executed as a series of interleaved initialization, gating, measurement, and ion-transport operations. With just three ion-transport primitive operations, qubits can be arbitrarily rearranged in the middle of a circuit: qubit ions paired with coolant ions use (1) linear transport of  $\{\text{Yb}, \text{Ba}\}$  and  $\{\text{Yb}, \text{Ba}, \text{Ba}, \text{Yb}\}$  crystals, (2) split or combine operations to go between  $\{\text{Yb}, \text{Ba}\}$  crystals and  $\{\text{Yb}, \text{Ba}, \text{Ba}, \text{Yb}\}$  crystals, and (3) physical (not quantum gate-based) swap operations to switch the ordering of the qubit ions in a crystal by transporting them around each other in two dimensions [59]. These operations allow any two qubits to be paired for entangling gates and allow qubits to be isolated for single-qubit gates, initialization, and measurement. Dynamically rearranging the ions during the circuits ensures the one-dimensional geometry trap does not restrict the geometry of codes used (the color code has a two-dimensional geometry).

The system is programmed according to the quantum circuit model. Logical qubit operations are compactly described at the physical qubit level as quantum circuits and are expressed in an extended version of OpenQASM 2.0 [60]. At the time of developing the QEC experiments discussed in this paper, the OpenQASM language did not fully support all of the conditional logical operations needed for QEC. We, therefore, extended the language to include classical assignment, classical operations, and expanded comparison of registers. These OpenQASM extensions were key enablers for the highly dynamical QEC protocols we chose to implement [50].

## II. EXPERIMENTS

In this section, we discuss our experiments demonstrating the QEC operations necessary for universal QEC computation restricted to a single logical qubit.

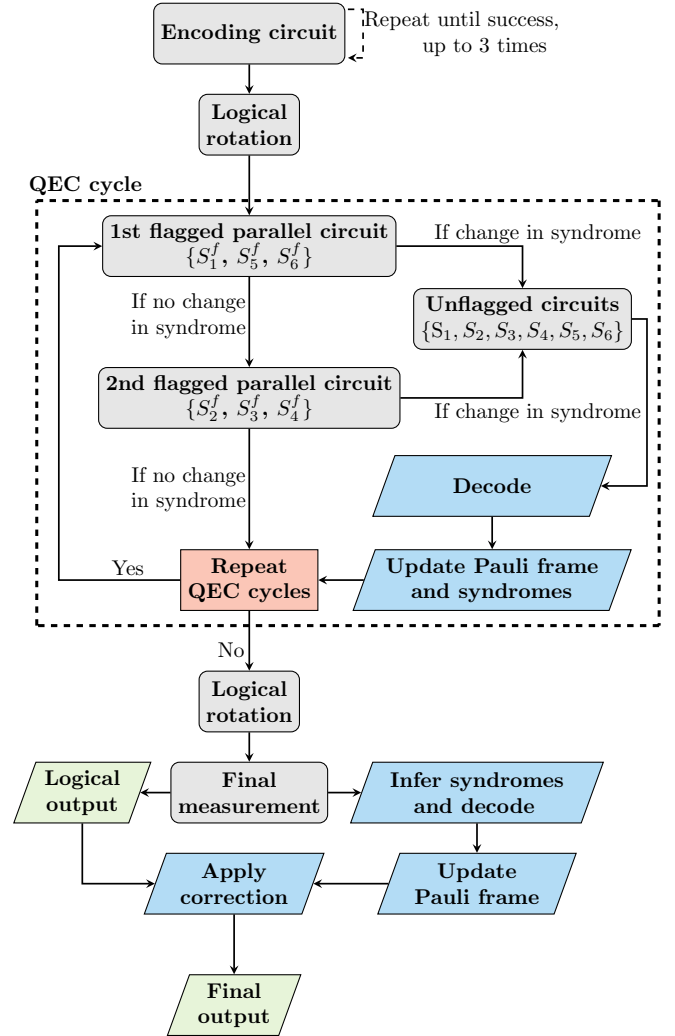


FIG. 3. A schematic of the QEC experiment. Here the steps taken in a QEC experiment are outlined including repeat-until-success initialization of a  $|0\rangle_L$  state, the logical rotation of the state, conditional branching taken to perform syndrome extraction fault tolerantly, decoding, the destructive logical measurement, and the determination of a logical measurement output. An extended version of this figure, which includes circuit diagrams, can be found in Fig. 10.

### A. QEC cycles

The main result of this paper is a full demonstration of the ability to repeat QEC cycles, which includes the determination of corrections during computation (see Fig. 14). This demonstration consists of initializing a logical Pauli basis state using a FT encoding circuit, running multiple adaptive FT QEC cycles, and then measuring in the appropriate logical basis, all while tracking corrections, running a decoding algorithm, and updating the corrections after each QEC cycle in real time. A schematic of the overall procedure can be seen in Fig. 3 with additional details found in Fig. 10. We now detail the various steps of this QEC protocol.

### 1. Logical state preparation

To prepare different logical basis states,  $\{|0\rangle_L, |1\rangle_L, |+\rangle_L, |-\rangle_L, |+i\rangle_L, |-i\rangle_L\}$ , we utilize a FT encoding circuit. The encoding circuit first prepares  $|0\rangle_L$  and fault tolerantly verifies successful preparation by measuring  $\bar{Z}$  [61] via three CNOT gates with an ancilla at the end as shown in Fig. 10. If the measured ancilla is found in  $|0\rangle$ , the circuit succeeded and moves to the next step. If the ancilla is found in  $|1\rangle$ , all qubits are reinitialized, and the circuit runs again. This procedure repeats until successful, up to three iterations if necessary. This verification succeeds with a total probability of 99.984(9)% with the first attempt succeeding 97.9(2)% of the time. Regardless of successful preparation, we proceed by rotating  $|0\rangle_L$  to another logical Pauli basis state by the appropriate application of the  $\bar{X}$ ,  $\bar{H}$ , and  $\bar{S}$  operators as prescribed in Fig. 1 (see Fig. 16).

### 2. Adaptive syndrome extraction protocol

Next, to protect the logical qubit while idling during a computation, we execute multiple QEC cycles and utilize ancilla qubits to measure syndromes in a nondestructive manner. The syndrome extraction protocol operates by detecting changes in stabilizer measurement outcomes that ideally give a +1 eigenvalue. Because syndrome extraction requires the use of noisy gates, syndrome measurements must be repeated multiple times within a single QEC cycle to be FT to syndrome measurement errors. For each FT QEC cycle, we implement the flagged three-parallel syndrome extraction protocol described in Ref. [50] (see Figs. 3, 10, and 14).

In order for the color code to be FT to all single-qubit errors, we must account for a special case of a single-qubit error called a “hook” error [55], which spreads to higher weight errors causing logical errors; see Fig. 11(a). To identify these hook errors from other errors of similar syndrome signatures [see Fig. 11(b)], the decoder requires two sets of syndrome measurements. First, the flagged syndromes  $S_i^f$  are measured using flagged parallel circuits that measure stabilizers in a way that flags (identifies) malignant hook errors. Then, if necessary, the second set of unflagged syndromes  $S_i$  is measured using standard unflagged syndrome circuits to distinguish hook errors from nonhook errors. Note that both sets of stabilizers follow from the definitions given in Fig. 1, but we use the  $f$  superscript to indicate syndromes measured using flagged circuits instead of those measured with unflagged circuits.

The syndrome extraction protocol is adaptive and relies on a comparison of previous unflagged syndrome measurements to determine the next step in the protocol. If previous unflagged syndrome measurements have not been made yet (e.g., the first QEC cycle), the current syndromes are compared to the trivial case of all +1 syndromes. The syndrome extraction protocol begins with the flagged parallel circuits to measure syndromes, the first set being

$\{S_1^f, S_5^f, S_6^f\}$ . The change in the current flagged syndrome eigenvalue measurements  $S_i^f$  compared to the last unflagged syndrome eigenvalue measurements made in a previous QEC cycle  $S_{i,\text{previous}}$  is determined. That is,  $\Delta S_i^f = S_i^f \times S_{i,\text{previous}}$  is calculated. If there are no changes in stabilizers’ measurements (i.e.,  $\Delta S_1^f, \Delta S_5^f, \Delta S_6^f = +1, +1, +1$ , which indicates no new errors have been detected), the second set of flagged stabilizers  $\{S_2^f, S_3^f, S_4^f\}$  is also measured. Then, if there are no changes in the second set of flagged stabilizers’ measurements, the syndrome extraction protocol is deemed complete. However, if either of the flagged circuits do indicate a change in syndromes, an additional and final round of syndrome extraction is triggered using standard circuits without flags  $\{S_1, S_2, S_3, \dots\}$ . Finally, at the end of every QEC cycle, syndrome changes are sent to a decoder to infer a correction, and the baseline stabilizer values  $S_{i,\text{previous}}$  are updated in software for the next QEC cycle.

An additional description of the steps in adaptive syndrome extraction protocol can be found in the flowchart Fig. 3 as well as in the Appendix F in Fig. 17 and also in Fig. 10, which is a more detailed flowchart including the circuitry used.

### 3. Decoder

Our decoder algorithm consists of two steps using two different lookup tables (see Tables I and II), analogous to the decoders described in Refs. [62,63]. Lookup tables are simple decoders that map syndromes to corrections. We decode  $X$  and  $Z$  errors separately and correct at the logical level instead of at the physical level (that is, the corrections are whether to apply  $\bar{I}$ ,  $\bar{X}$ ,  $\bar{Y}$ , or  $\bar{Z}$ ). Physical errors can always be decomposed into two commuting operators, one logical operator and another operator that does not apply a logical operator but at most changes the syndromes. (Such errors that at most modify syndromes are products of elements in the stabilizer and destabilizer groups. See Ref. [64], Chap. 2 in Ref. [65], and Chap. 4 in Ref. [66].) Therefore, the physical errors can be corrected by the application of a logical correction and a syndrome update, which is a record of the last measured set of syndromes  $S_{i,\text{previous}}$ . Given this record of syndromes measured from the previous QEC cycle and the measurement of the current set of syndromes, the changes in syndromes are calculated (i.e.,  $\Delta S_i = S_i \times S_{i,\text{previous}}$  and  $\Delta S_i^f = S_i^f \times S_{i,\text{previous}}$ ). It is these changes in syndromes that are sent to the decoder to come up with a logical correction. By concerning ourselves with only the current changes in syndromes and not directly modifying the syndromes via the correction, we are able to mod out the part of the error that flips syndrome outcomes and correct only the logical component. An advantage of choosing to only consider corrections of logical errors is

TABLE I. First lookup table decoder. This logical-level decoder only considers the changes in unflagged syndrome measurements  $\Delta S_i$ . The changes in syndromes corresponding to  $X$ -type and  $Z$ -type stabilizer generator are decoded separately. A slash mark separates the indices of the set of  $X$ -type and  $Z$ -type stabilizer measurements, respectively, and the corresponding logical corrections. The numbering of the syndromes  $\Delta S_i$  is consistent with the numbering of the stabilizer generators in Fig. 1. The circuitry for the measurement of these syndromes can be found in Fig. 10. If a syndrome outcome is not indicated in the table, then the logical correction is  $\bar{I}$ . Note that this decoder is equivalent to the one described in Fig. 21.

$\Delta S_{1/4}$	$\Delta S_{2/5}$	$\Delta S_{3/6}$	Correction
+1	-1	+1	$\bar{Z}/\bar{X}$
+1	+1	-1	$\bar{Z}/\bar{X}$
+1	-1	-1	$\bar{Z}/\bar{X}$

that it reduces the size of the decoder to a few if statements in QASM (see Figs. 21 and 22).

For completeness, we now step through the two stages of decoding during a QEC cycle. During a particular QEC cycle, the decoder is called if and only if the final round of syndrome extraction is triggered (i.e., the unflagged circuits; see Fig. 10).

The first stage in the decoding algorithm utilizes the unflagged syndrome measurements  $S_i$  and ignores the flagged syndrome measurements  $S_i^f$  since this decoder is only concerned with the nonhook errors. Instead of sending the directly measured unflagged syndromes  $S_i$ , the changes in syndromes are sent to the decoder. Because of symmetry, the same lookup table decoder (see Table I) is used to decode the  $X$ -type syndromes ( $\Delta S_1$ ,  $\Delta S_2$ , and  $\Delta S_3$ ) and the  $Z$ -type syndromes ( $\Delta S_4$ ,  $\Delta S_5$ , and  $\Delta S_6$ ). Combining these two  $X$ -type and  $Z$ -type syndrome corrections, a logical correction is then determined for this first stage.

The second stage of the decoding algorithm utilizes the flagging information to identify hook errors and to provide an additional correction to update the one provided in the first stage. Single-qubit hook errors produce the same syndrome measurements in the unflagged circuits as other less damaging single-qubit errors (see Fig. 11). Thus, to distinguish between these two types of single-qubit errors, we require additional syndrome information from the flag circuits. To do this, we send both the changes in unflagged syndrome measurements  $\Delta S_i$  and the changes in the flagged syndrome measurements  $\Delta S_i^f$  in the lookup table decoder (see Table II). In particular, we once again decode the  $X$ -type syndromes ( $\Delta S_1^f$ ,  $\Delta S_2^f$ , and  $\Delta S_3^f$  versus  $\Delta S_1$ ,  $\Delta S_2$ , and  $\Delta S_3$ ) and the  $Z$ -type syndromes ( $\Delta S_4^f$ ,  $\Delta S_5^f$ , and  $\Delta S_6^f$  versus  $\Delta S_4$ ,  $\Delta S_5$ , and  $\Delta S_6$ ) separately. The corrections determined by the first lookup table and second lookup table then determine the final logical correction for the current QEC cycle given an overall logical  $\bar{I}$ ,  $\bar{X}$ ,  $\bar{Y}$ , or  $\bar{Z}$ ,

TABLE II. Second lookup table decoder. This logical-level decoder considers the changes in the flagged syndrome measurements  $\Delta S_i^f$  and unflagged syndrome measurements  $\Delta S_i$  to give an additional correction to the first lookup table (see Table I). The changes in syndromes corresponding to  $X$ -type and  $Z$ -type stabilizer generator are decoded separately. A slash mark separates the indices of the set of  $X$ -type and  $Z$ -type stabilizer measurements, respectively, and the corresponding logical corrections. The numbering of the syndromes  $\Delta S_i$  and  $\Delta S_i^f$  is consistent with the numbering of the stabilizer generators in Fig. 1. The circuitry for the measurement of these syndromes can be found in Fig. 10. If a syndrome outcome is not indicated in the table, then the logical correction is  $\bar{I}$ . Note that this decoder is equivalent to the one described in Fig. 22.

$\Delta S_{1/4}^f$	$\Delta S_{2/5}^f$	$\Delta S_{3/6}^f$	$\Delta S_{1/4}$	$\Delta S_{2/5}$	$\Delta S_{3/6}$	Correction
-1	+1	+1	+1	-1	+1	$\bar{Z}/\bar{X}$
-1	+1	+1	+1	+1	-1	$\bar{Z}/\bar{X}$
+1	-1	-1	+1	+1	-1	$\bar{Z}/\bar{X}$

which is then used to update the software-tracked correction determined from previous QEC cycles. Note that the combination of the conditional rounds of syndrome extraction and the second lookup table makes the QEC cycle FT to single Pauli errors including hook errors as well as measurement errors.

After the unflagged syndrome extraction is triggered and the decoding process has completed, the record of the last syndrome measurement  $S_{i,\text{previous}}$  is then updated using the syndromes measured by the unflagged syndrome measurement circuits. In this FT protocol the unflagged syndromes are treated as ideal measurements since they are only measured in the event that a new error has been detected and the code is only guaranteed to correct a single error. In this way, the unflagged syndrome establishes the “ground truth” of the last measured syndromes to be compared against in future QEC cycles.

#### 4. Pauli frame update

Applying corrections physically via noisy gates can potentially induce more errors. Fortunately, many errors can be corrected without implementing physical gate operations and instead done with an essentially perfect software correction. To this end, the correction for the logical qubit is stored in a binary array, known as a Pauli frame [67], during the computation. The Pauli frame is represented by two bits, corresponding to the possible corrections  $\{\bar{I}, \bar{X}, \bar{Y}, \bar{Z}\}$ . At the end of each QEC cycle, the new correction is combined with the previous Pauli frame according to Pauli matrix multiplication rules, and the binary array is updated (see the bottom of Fig. 17).

Finally, after completing a variable number of QEC cycles, the data qubits are directly measured (see the bottom of Fig. 14). The final destructive measurement

projects the state into the logical  $\bar{X}$ ,  $\bar{Y}$ , or  $\bar{Z}$  basis based on the logical single-qubit gate applied before measurement, and from this measurement, two pieces of critical information are extracted. First, the raw logical output is calculated by multiplying the  $\pm 1$  outcomes of the non-identity components of the logical operator being measured (qubits 5, 6, and 7; see Fig. 1). Second, while the process of measuring the data qubits is destructive to the quantum state, the syndromes that commute with the measured operator can still be constructed from the resulting classical information, allowing for a final correction. For example, when measuring the fidelity of  $|+\rangle_L$ , we measure all data qubits in the  $X$  basis and the raw logical output is given by results of qubits 5, 6, and 7, while the syndrome is equal to the measurement result tuple of  $\{S_1, S_2, S_3\}$ . However, while for  $\bar{X}$  and  $\bar{Z}$  measurements it is sufficient to infer the syndromes using stabilizers of the same Pauli type in Fig. 1, this is not the case for the  $Y$  basis. Fortunately, in the special case of a  $\bar{Y}$  measurement, we can infer the syndromes of the three  $Y$  stabilizers that are the products of the  $X$  and  $Z$  stabilizers listed in Fig. 1 that have identical nontrivial support. After the syndromes are inferred from the final measurement, the decoding algorithm is used to determine a correction, simply whether to flip the raw logical output or not.

### 5. Results

The experimental results for the full QEC protocol can be seen in Fig. 4. Each basis state and QEC cycle iteration was run multiple times with a varying number of trials. To better estimate the error bars due to experimental noise fluctuations, we used a jackknife resampling method [68] to estimate the average and the standard deviation of the different trials for a given state. To estimate the logical state preparation and measurement (SPAM) and logical error rate per QEC cycle, we fit the data to an exponential decay curve and extract the fitting parameters as shown in Table III. In the protocol, logical SPAM is equivalent to doing zero QEC cycles. We measured the average logical SPAM error to be  $1.7(2) \times 10^{-3}$ , compared to the  $2.4(4) \times 10^{-3}$  SPAM error of our physical qubits. The data in Fig. 4 show that while the initialization circuit produces high-fidelity states, repeated QEC cycles introduce significant logical errors of an average of approximately 1.75(4)% per QEC cycle as determined by exponential decay fits to the data (see Table III). We observe that the logical  $\bar{Z}$  basis is more robust than the other two bases, suggesting that logical  $\bar{Z}$  errors are more common. As seen in Fig. 1, the  $\bar{Z}$  operator is composed of only  $Z_i$  operators on physical qubits, suggesting that qubit dephasing is a significant source of error.

The experimental run time of the logical SPAM portion of experiments, including the encoding circuit, two rounds of single-qubit rotations, and the final measurement, is  $< 60$  ms, and each QEC cycle takes less than  $< 200$  ms. Transport accounts for  $\sim 10$  ms of logical SPAM operations

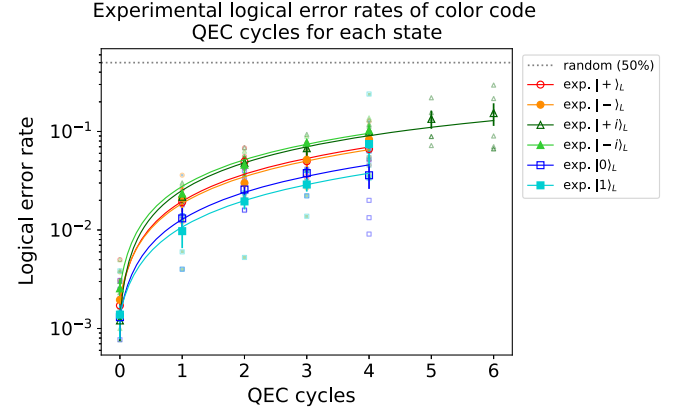


FIG. 4. Comparing the observed logical fidelities of the six logical Pauli basis states over many QEC cycles. Averages and standard deviations were determined by jackknife resampling between individual experiments [68]. The large points with error bars are experimental averages and standard deviations. The smaller and lighter points are individual trials for a given QEC cycle experiment. Note that for 0 QEC cycles, numerous experimental trials have an error rate of 0 and, thus, are not displayed due to the log scale. The lines are fits to the experimental averages, where the fits are exponential decay curves  $p_L(c) = 0.5 + (p_{\text{SPAM}} - 0.5)(1 - 2p_{\text{cycle}})^c$  (see Appendix B for derivation). Here  $p_L(c)$  is the logical error rate of a cycle  $c$  and logical basis  $L$ ,  $p_{\text{SPAM}}$  the logical SPAM error, and  $p_{\text{cycle}}$  is the logical QEC cycle error. Note that  $p_{\text{SPAM}}$  is determined directly from the 0 QEC cycle results and fixed when determining the fits. Thus, from the fits, we obtain estimates of  $p_{\text{cycle}}$ . The values of  $p_{\text{SPAM}}$  and  $p_{\text{cycle}}$  are reported in Table III.

and  $< 70$  ms of each QEC cycle. The remaining time is dominated by the cooling operations, which occur prior to each two-qubit gate, similar to the experiments detailed in Ref. [53].

To put the logical qubit error rates and clock speed into context by comparing to the physical layer; the dominant error at the physical level is the two-qubit gate error of  $\sim 3 \times 10^{-3}$  (Table VI), and the physical layer clock speed

TABLE III. Observed logical SPAM error  $p_{\text{SPAM}}$  from 0 QEC cycle experimental results. Here, logical error per QEC cycle  $p_{\text{cycle}}$  fit parameters for the six Pauli basis states are presented, which are obtained via exponential decay fit and used in the plotted curves in Fig. 4. The average  $p_{\text{SPAM}}$  is  $1.7(2) \times 10^{-3}$  and the average  $p_{\text{cycle}}$  is  $1.75(4) \times 10^{-2}$ .

State	$p_{\text{SPAM}}$	$p_{\text{cycle}}$
$ +\rangle_L$	$1.7(7) \times 10^{-3}$	$1.8(1) \times 10^{-2}$
$ -\rangle_L$	$2.0(7) \times 10^{-3}$	$1.7(2) \times 10^{-2}$
$ +i\rangle_L$	$1.2(5) \times 10^{-3}$	$2.4(1) \times 10^{-2}$
$ -i\rangle_L$	$2.6(6) \times 10^{-3}$	$2.5(1) \times 10^{-2}$
$ 0\rangle_L$	$1.3(5) \times 10^{-3}$	$1.16(7) \times 10^{-2}$
$ 1\rangle_L$	$1.4(6) \times 10^{-3}$	$0.94(7) \times 10^{-2}$

(one round of gating between random pairs of qubits) is 10.5 ms. We note that a complete understanding of the overhead associated with QEC will have to wait until logical qubit entangling operations are characterized.

### B. Active versus software corrections

Pauli frame updates cannot always be used for a QEC computation since Pauli operators have nontrivial transformations under conjugation by non-Clifford gates. Thus, to implement corrections stored in software, we either physically apply the correction to the qubits before the gate or adapt the non-Clifford gate to include the correction (see Fig. 15). To demonstrate the ability to physically apply corrections as needed, we use the logical  $S$  gate as a stand-in for the non-Clifford  $\bar{T}$  gate. We first initialized the state  $|+\rangle_L$ , ran one cycle of QEC to generate potential corrections, physically applied a correction (either  $\bar{X}$ ,  $\bar{Y}$ , or  $\bar{Z}$ ) according to the Pauli frame, physically applied logical  $\bar{S}$ , performed an additional QEC cycle, and measured in the  $\bar{Y}$  basis (note  $\bar{S}|+\rangle_L = |+i\rangle_L$ ; see Fig. 12), achieving a logical fidelity of 93(2)%.

We repeated this experiment without applying a physical correction and instead rotated the Pauli frame according to Pauli transformations of the physically applied  $\bar{S}$  gate. This software-correction version of the experiment achieved a logical fidelity of 92(1)%. These two error rates are not significantly different from each other, thus demonstrating the key ability to take software-tracked corrections and apply them in real time as necessary.

Note, for comparison, that the fidelity of two QEC cycles of the  $|+\rangle_L$  and  $|+i\rangle_L$  logical basis states experiments without the logical  $S$  gate (as given in Fig. 4) was found to be 95.1(7)% and 95.2(8)%, respectively, while the fidelity of logical  $S$  gate experiments, which also included a total of two QEC cycles, was 93(2)% for the experiments with physically applied corrections and 92(1)% for the experiments with software-tracked corrections. This indicates that the physical application of the logical  $S$  gate may have resulted in roughly 2%–3% additional error. We leave further investigation of the noise due to logical single-qubit transversal gates to future studies.

### C. Preparing a magic state

Universal quantum computing requires the ability to implement non-Clifford gates, which, unlike Clifford gates, cannot be constructed by simple transversal operations in the color code. Non-Clifford gates can, however, be implemented using state preparation primitives that produce so-called magic states [69]. One choice of magic state that is sufficient to complete a universal gate set is  $\bar{T}|+\rangle_L = (|0\rangle_L + e^{i\pi/4}|1\rangle_L)/\sqrt{2}$ , where  $\bar{T} = \text{diag}(1, e^{i\pi/4})$ . Note that we cannot use the FT encoding circuit used for the logical Pauli states because the verification step requires the measurement of a logical operator which would collapse the  $T$

state, and thus, we use a non-fault-tolerant encoding circuit for the color code [70], shown in Fig. 13 (see Fig. 14). Once prepared,  $\bar{T}|+\rangle_L$  can be used to apply  $\bar{T}$  gates via gate teleportation [71] in a system capable of logical two-qubit gates. The fidelity of this operation is estimated by measuring the operator  $\bar{T}|+\rangle_L\langle +|_L\bar{T}^\dagger = \frac{1}{2}[\bar{I} + \frac{1}{\sqrt{2}}(\bar{X} + \bar{Y})]$ , and we report an error of 2.2(6)%. Theoretical analysis shows this error level to be sufficiently low to serve as logical magical state inputs to distillation protocols [69,72], indicating we are able to produce high-quality, distillable states to implement FT non-Clifford gates. However, a further study is needed to analyze the performance of a full logical state distillation protocol given our logical input state fidelity and an experimental system, undergoing our particular noise, that is large enough to enact such a logical protocol [73]. We leave these studies to future numerical analysis as well as the experimental validation that will eventually be required.

## III. SIMULATIONS AND ANALYSIS

To help understand the noise in our system, we compare the experimental results of the QEC protocol to numerical simulations as seen in Fig. 5. The simulations are state-vector simulations [66,74] utilizing a realistic error model

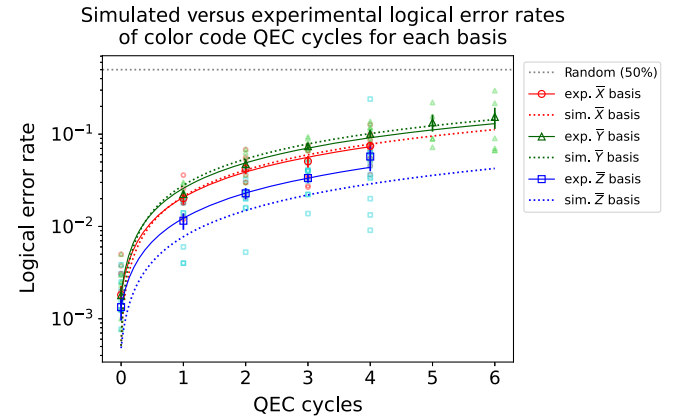


FIG. 5. Comparing the observed and simulated logical fidelities of the logical Pauli bases. Averages and standard deviations were determined by combining the data of both basis states of a given basis and using jackknife resampling between individual experiments [68]. The large points with error bars are experimental averages and standard deviations. The smaller and lighter points are individual trials for a given QEC cycle experiment. Note that for 0 QEC cycles, numerous experimental trials have an error rate of 0 and, thus, are not displayed due to the log scale. The lines are fits to the simulated and experimental averages, where the fits are exponential decay curves  $p_L(c) = 0.5 + (p_{\text{SPAM}} - 0.5)(1 - 2p_{\text{cycle}})^c$  (see Appendix B for derivation). Here  $p_L(c)$  is the logical error rate of a cycle  $c$  and logical basis  $L$ ,  $p_{\text{SPAM}}$  the logical SPAM error, and  $p_{\text{cycle}}$  is the logical QEC cycle error. Note that  $p_{\text{SPAM}}$  is determined directly from the 0 QEC cycle experiments and fixed when determining the fits. Thus, from the fits, we obtain estimates of  $p_{\text{cycle}}$ . The values of  $p_{\text{SPAM}}$  and  $p_{\text{cycle}}$  are reported in Table IV.

and experimentally measured error parameters (details in Table VI). The error model includes simple depolarizing gate noise, leakage errors, and modeling of coherent dephasing noise during transport and cooling operations. The simulation uses the same instructions that are generated by the compiler and sent to the quantum computer, including the same gate decomposition, gate duration, and transport operations.

Our analysis is presented in three sections. Section III A describes how we use the measured data to complete our microscopic system modeling and how we construct a simple error channel at the logical level. Section III B describes how the microscopic system model is used to determine how individual error sources are manifested at the logical level. Section III C describes our use of the microscopic model to estimate the error levels required to reach the pseudothreshold for this code.

### A. Microscopic simulations and the logical error channel

The complex physical layer error model can be distilled to a much simpler logical error model described by an asymmetric depolarizing channel,

$$\tilde{\mathcal{E}}(\rho) = (1 - p_L)\rho + p_x\bar{X}\rho\bar{X} + p_y\bar{Y}\rho\bar{Y} + p_z\bar{Z}\rho\bar{Z}, \quad (1)$$

where the Pauli error probabilities  $p_x$ ,  $p_y$ , and  $p_z$  are fitting parameters for both the experimental and simulation data. Note that  $p_L = p_x + p_y + p_z$  ensures the map is trace preserving and  $p_L$  is referred to as the logical error rate.

For individual basis states, only the Pauli operators that do not commute with the state cause errors. For example,

the  $\bar{Z}$  basis is immune to  $\bar{Z}$ -type noise but susceptible to  $\bar{X}$ - and  $\bar{Y}$ -type noise. Therefore, the error probabilities associated with the different bases are given by the three equations:

$$\begin{aligned} \bar{X} \text{ basis: } p_{yz} &= p_y + p_z, \\ \bar{Y} \text{ basis: } p_{xz} &= p_x + p_z, \\ \bar{Z} \text{ basis: } p_{xy} &= p_x + p_y. \end{aligned} \quad (2)$$

By fitting the experimental and simulated data of individual basis states to exponential decay curves, the probabilities  $p_{yz}$ ,  $p_{xz}$ , and  $p_{xy}$  are solved for and are reported in Table IV. The system of equations in Eq. (2) can then be inverted to solve for the depolarizing parameters  $p_x$ ,  $p_y$ , and  $p_z$  and are reported in Table V.

For most of the error parameters, we have experimental estimates (see Appendix D). However, accounting for different sources of dephasing noise is not straightforward. To account for dephasing, we vary the dephasing rate used in the simulation and compare it with the experimental results. The depolarizing parameters determine the logical error rate  $p_L$ , which we use to determine the best-fit dephasing rate empirically.

The simulation and experimental results qualitatively agree, suggesting that the most important sources of noise are understood. However, further investigation is needed to fully understand the impact of additional known errors and unknown error sources. In particular, it is important that we further characterize both coherent and incoherent phase noise sources independent of QEC. We also note the bias in the logical error model toward the  $\bar{Z}$  component and the

TABLE IV. Experimental and simulated logical SPAM error obtained from the data in Fig. 5. The average SPAM error rate is  $1.6(4) \times 10^{-3}$  for experiment and  $4.9(1) \times 10^{-4}$  for simulation, and the average QEC cycle error rate is  $1.80(6) \times 10^{-2}$  for experiment and  $1.85(7) \times 10^{-2}$  for simulation.

Basis	SPAM		QEC cycle	
	Simulation	Experiment	Simulation	Experiment
$\bar{X}$ : $p_{yz}$	$5.0(1) \times 10^{-4}$	$1.8(5) \times 10^{-3}$	$2.1(1) \times 10^{-2}$	$1.89(6) \times 10^{-2}$
$\bar{Y}$ : $p_{xz}$	$5.0(1) \times 10^{-4}$	$1.6(4) \times 10^{-3}$	$2.75(4) \times 10^{-2}$	$2.4(1) \times 10^{-2}$
$\bar{Z}$ : $p_{xy}$	$4.8(1) \times 10^{-4}$	$1.3(4) \times 10^{-3}$	$7.3(1) \times 10^{-3}$	$1.10(4) \times 10^{-2}$

TABLE V. Experimental and simulated logical SPAM and QEC cycle error channel rates for the channel in Eq. (1) given basis error rates from Table IV.

Error	SPAM		QEC cycle	
	Simulation	Experiment	Simulation	Experiment
$p_x$	$2.4(1) \times 10^{-4}$	$7(4) \times 10^{-4}$	$7.1(7) \times 10^{-3}$	$8.1(6) \times 10^{-3}$
$p_y$	$2.4(1) \times 10^{-4}$	$7(4) \times 10^{-4}$	$2(7) \times 10^{-4}$	$2.9(6) \times 10^{-3}$
$p_z$	$2.6(1) \times 10^{-4}$	$1.1(4) \times 10^{-3}$	$2.04(7) \times 10^{-2}$	$1.61(6) \times 10^{-2}$
$p_L$	$7.4(1) \times 10^{-4}$	$2.5(4) \times 10^{-3}$	$2.77(7) \times 10^{-2}$	$2.71(6) \times 10^{-2}$



interesting asymmetry between the  $\bar{X}$  and  $\bar{Y}$  components. While the  $\bar{Z}$  bias may be explained by the asymmetry in the microscopic noise model from the presence of dephasing, there is no such asymmetry between  $X$  and  $Y$  in the physical layer error model. Therefore, we suspect that the circuit structure tends to convert  $Z$  noise asymmetrically; however, we leave a detailed analysis to future work.

Since the dephasing errors manifest from different sources (spatial and temporal magnetic field fluctuations, relative phase drifts between different laser beams, etc.), there are likely both coherent and incoherent sources of qubit dephasing in our system. To gain an understanding of the impact of coherent dephasing on the QEC protocol, we ran both coherent (state-vector) and incoherent (stabilizer) simulations. The coherent dephasing is modeled as the channel,

$$R_Z(\theta)\rho R_Z(\theta)^\dagger = \exp(-iZ\theta/2)\rho \exp(+iZ\theta/2), \quad (3)$$

which is applied between ideal gates where  $\theta =$  dephasing rate  $\times$  duration. The duration corresponds to the time it takes for transport operations or the qubit idling time while operations are being applied to other qubits. For the incoherent simulation, dephasing is modeled as the Pauli twirled version of the coherent channel, where  $Z$  is applied stochastically with a probability of

$$p_{\text{dephase}} = \sin(\theta/2)^2. \quad (4)$$

The dephasing rate needed to account for the logical QEC cycle error rate was 0.26 Hz for the coherent simulation and 0.43 Hz for the incoherent simulation, indicating that coherent buildup in the distance three color code may affect its performance and should be studied in future work. However, we expect such single-qubit coherent noise to be less of an issue as the distance of the code is increased as suggested by Ref. [75]. Note that all simulation numbers reported were generated using the coherent dephasing model; however, given identical error parameters except for the dephasing rates mentioned above, the results of the two simulations were nearly identical.

## B. Logical level error budget

How different error sources such as leakage and dephasing translate from the physical level to the logical level can be unclear. To understand how different microscopic error sources contribute to QEC cycle logical errors, we construct an error budget for three different physical layer error categories: unitary gates (the single- and two-qubit gates), measurement and initialization, and dephasing noise. Here, measurement and initialization includes both

SPAM as well as midcircuit measurement and reset (MCMR). Note that the three error categories contain multiple noise mechanisms; for example, the unitary gate errors contain not just depolarizing noise but also spontaneous emission, which includes leakage as well (see Appendix D for more details).

Ideally, errors at the physical level occurring with probability  $\mathcal{O}(p)$  should be suppressed by QEC to  $\mathcal{O}(p^2)$ . Therefore, we analyze the impact of errors using a second-order model given by

$$\begin{aligned} p_L(s, \{p_i\}) &= s \sum_i a_i p_i + s^2 \sum_{i,j} b_{ij} p_i p_j \\ &= s \sum_i A_i + s^2 \sum_{i,j} B_{i,j}, \end{aligned} \quad (5)$$

where  $p_i$  are the relative probabilities of SPAM, two-qubit gate, and dephasing errors at the physical level, and  $A_i \equiv a_i p_i$  and  $B_{ij} \equiv b_{ij} p_i p_j$  are coefficients we solve for by scaling  $s$  and using a numerical fitting routine. As discussed in the previous section, the  $p_i$  parameters are characterized independently, but the  $a_i$  and  $b_{ij}$  coefficients are complicated functions of both the circuit and error structures that are currently not well understood. The  $A_i$  and  $B_{ij}$  coefficients are solved for using the microscopic simulator. We first set two of the three error probabilities  $p_i$  and  $p_j$  to zero and then calculate  $p_L(s, p_i = 0, p_j = 0)$  in simulations. By varying  $s$  in these simulations, we can fit the simulation data to a parametrized curve with both linear and quadratic components to determine  $A_k$  and  $B_{kk}$  for one particular error category indexed by  $k$ . This is repeated for the two other error categories, giving all three elements of  $A$  and the three diagonal elements of  $B$ . The off-diagonal elements of  $B$  are solved for by only setting one  $p_i$  to zero and setting  $s = 1$ ; for example,  $B_{1,2} = \frac{1}{2} p_L(s=1, p_3=0) / (A_1 + A_2 + B_{1,1} + B_{2,2})$ . The estimated individual and correlated contributions to the logical error are shown in Fig. 6.

The error budget breakdown indicates that the noise due to unitary gates and dephasing account for the majority of the logical error, accounting for approximately 45% and 49%, respectively, while measurement and initialization account for the remaining 6%. However, studying only the individual contributions obtained by the calculation in Fig. 6, we miss some important information contained in  $A$  and  $B$ . As can be seen in Fig. 6, the contribution to the logical error that is linear in the physical errors  $A$  is dominated by the gate errors. Ideally, the logical error probability should be quadratic in the gate error, and the large linear dependence is likely due to the leakage error associated with the gate operations. This point will be further illustrated below as we study the pseudothreshold of the code.

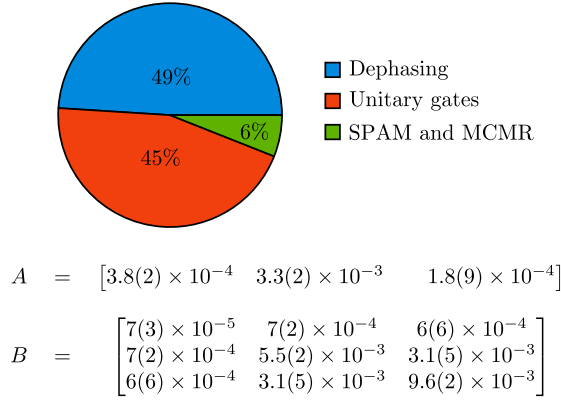


FIG. 6. The logical qubit error budget. The pie chart shows the percent contribution to the logical QEC cycle error rate from the physical noise due to SPAM and MCMR, unitary gates, and qubit dephasing.  $A$  and  $B$  are the vector and matrix in Eq. (5) and the elements are ordered as {SPAM and MCMR, gates, dephasing}. The individual contributions in the pie chart indexed by  $i$  are calculated as  $(A_i + \sum_j B_{ij}) / (\sum_i A_i + \sum_{ij} B_{ij})$ .

### C. Pseudthreshold estimates

While the error budget is useful in studying the current impact different physical error sources have on the QEC protocol, we now investigate a simulation tool better suited to predicting the protocol performance assuming improved physical error rates.

For QEC to be helpful in computations, logical level error rates must be below the physical level error rates. This crossover point is known as the “pseudthreshold.” In general, pseudthreshold estimates are difficult because they require a detailed understanding of the system’s underlying physics. However, in cases where the dominant physical noise mechanism is known, one method is to scale that noise mechanism in simulations and solve for the pseudthreshold, defining it as the point where the logical error rate is equal to the dominant physical noise source. The largest error rate in the simulation model is the two-qubit depolarizing error rate of  $p2 = 3.1 \times 10^{-3}$  (see Appendix D for details). Note that the two-qubit gate contributes not only depolarizing noise, but also leakage noise via spontaneous emission at a probability of  $5.5 \times 10^{-4}$ ; however, to simplify discussions, we consider only the two-qubit depolarizing rate. Also, while dephasing has a large impact on the logical error budget, the physical level dephasing error rate per physical qubit operation is an order of magnitude lower than  $p2$ . That is, with a dephasing rate of 0.26 Hz and each duration between gates, the stochastic error probability averaged over the entire circuit is approximately  $2.2 \times 10^{-4}$  for each qubit. Thus, in the simulations, on the physical per operation level, the dephasing error channel has a lower error rate than the two-qubit depolarizing rate.

While our simulations lack the level of detail needed for precise estimates, we use a crude model to estimate system

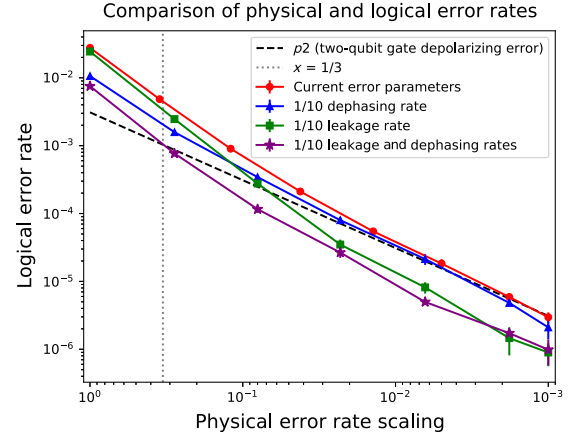


FIG. 7. Comparison of the logical error rate to different error models as the physical error rates are scaled. The dashed line is equal to the physical layer two-qubit depolarizing error rate,  $p2$ , times the scaling factor defining the  $x$  axis, which we use to define our pseudthreshold line. The four solid lines represent the logical qubit error rate for different simulated error model scenarios. They are plotted as a function of an overall scaling of the error model parameters. The red line (circles) uses our current error model parameters estimate, which are detailed in Table VI; the blue line (triangles) was generated assuming our current error parameters except with the dephasing rate being reduced by  $10\times$ ; the green line (squares) was generated assuming our current error parameters except with the leakage rate being reduced by  $10\times$ ; the purple line (stars) was generated by starting with our current error parameters and suppressing both the dephasing and the leakage rates by  $10\times$ . The vertical dotted line ( $x = 1/3$ ) is approximately where the purple line crosses the pseudthreshold. We find that reducing the leakage will be key to achieving a QEC cycle logical error rate below pseudthreshold (when the logical error rate is less than the leading physical error rate). Note that some error bars are smaller than the markers.

improvements needed to pass the conjectured pseudthreshold. To this end, we first simulated the logical error rate of a single QEC cycle and scaled all physical error rates used (see Table VI) by a common scaling factor. This is a coarse view of how much our dominant bare physical error (the two-qubit gate) needs to improve to reach the simulated pseudthreshold. As seen in Fig. 7, simply improving all our errors by a common factor is not enough to reduce our logical error rate below the two-qubit depolarizing error rate  $p2$ . To understand this, we also simulated three other models with different relative error budget breakdowns, first reducing the dephasing rate to  $1/10$  the original value, then reducing the leakage rate to  $1/10$  the original value, and then both reducing the dephasing and leakage rate to  $1/10$  their original value. By scaling these additional error models by an overall scaling factor, we find that suppressing the dephasing rate is likely not sufficient to reach the pseudthreshold, but that substantially reducing the relative leakage rate *and* the dephasing rate should put the pseudthreshold within reach if the two-qubit gate error can be reduced by approximately a factor of 3.

The relative importance of taming leakage errors is not unexpected. Most QEC protocols were designed to correct errors within the qubit manifold, an assumption that leakage errors violate. When leakage errors occur (e.g., during spontaneous laser scattering events), leaked qubits can spread errors as they interact with other qubits and the circuits for the color code are, in fact, only FT to qubit errors and not leakage errors. While leakage is not a leading source of error in our system, if left untreated it will saturate the system [76], eventually corrupting all the qubits and the logical information. As shown in Ref. [77], leakage errors can be reduced by roughly 3 orders of magnitude through physical mitigation techniques. Alternatively, circuit-level techniques can be incorporated in the QEC protocol [51,78,79].

Additionally, we believe there are routes to improving dephasing rates, including dynamical decoupling, additional shielding, and spatial phase tracking in the control software to account for spatial inhomogeneities in the magnetic field.

#### IV. CONCLUSION

In this work, we demonstrated the primitives needed for quantum error correction restricted to a single logical qubit, including high-fidelity state preparation and readout of logical basis states and a magic state, logical single-qubit gates, and repeatable error correction cycles. By establishing the necessary hardware capabilities along with detailed simulations of the processes, we can now begin developing a QCCD system architecture that is optimized for computations at the logical level.

The experimental data and simulated results highlight the need for substantial error rate improvements to get well below the pseudothreshold. The largest contributor to the physical level error budget is the two-qubit gate and, as noted in Refs. [53,80], we believe the dominant error mechanisms can be suppressed with upgraded electrode voltage sources and laser systems. Using the simulator developed here, we identify leakage and dephasing as crucial sources of noise. Interestingly, while leakage is not currently a dominant physical error, scaling the physical error rates in simulation indicates that leakage will dominate the logical error rate. While leakage errors are particularly detrimental to code performance, they can be converted to Pauli errors with the addition of a leakage repumping routine [77] and should not present a fundamental roadblock. Perhaps surprisingly, even though dephasing errors are relatively small compared to two-qubit gate errors at the physical level, our logical qubit error budget indicates dephasing errors have a large impact on the code performance. Fortunately, trapped-ion hyperfine qubits have been shown to exhibit much longer coherence times [81] compared to our current dephasing rates, and we believe this particular error can be significantly improved by a combination of improved shielding, dynamical

decoupling, and accurate mapping of magnetic field inhomogeneities and spatial phase tracking in the control software. Additionally, dephasing errors can be suppressed further through speed improvements, which are currently bottlenecked by laser cooling and transport operations, both of which can be substantially improved through advanced techniques [82,83], and we note that smaller trap geometries [84] or two-dimensional topologies [85] could also improve transport times. At the logical level, some interesting noise mitigation techniques include Pauli or Clifford twirling [86–89], randomized compiling [90,91], circuit-level gadgets to suppress coherent errors [92–94] and leakage errors [78,79], subsystem codes, codes designed for biased noise [95], and larger distance codes [75].

These experiments and emulation tools pave the way for codesigning QCCD systems and QEC software to implement large-scale fault-tolerant quantum computers. The next milestones on the road to a fully quantum error-corrected computer include logical operations between multiple qubits [33,96,97] and operation below the pseudothreshold.

#### ACKNOWLEDGMENTS

We would like to thank Charlie Baldwin and Michael Foss-Feig for helpful comments on the manuscript. We would also like to thank Daniel Lidar and Craig Gidney for useful discussions. Most importantly, we thank the entire group at HQS for their superb work and many contributions. In particular, we thank Joe Chambers and Tom Skripka for the real-time engine implementation capable of making fast control decisions based on midcircuit measurement results, Raanan Tobey and Matt Bohn for specialized laser system work enabling high-fidelity gate operations, and Bryan Spann for his work to construct parallelized laser beam delivery systems.

#### APPENDIX A: EXPERIMENTAL METHODS

The physical qubits are encoded in the  $^{171}\text{Yb}^+$   $S_{1/2}$  hyperfine clock states  $|F=0, m_f=0\rangle$  and  $|F=1, m_f=0\rangle$  with  $F$  and  $m_f$ , respectively, being the total angular momentum and the  $z$ -projection quantum numbers. Gating operations are implemented with stimulated Raman transitions [53,98] with single-qubit gates being performed with two-ion  $\{\text{Yb}, \text{Ba}\}$  crystals using two copropagating circularly polarized beams at 368.0 nm, and two-qubit operations being performed with four-ion  $\{\text{Yb}, \text{Ba}, \text{Ba}, \text{Yb}\}$  crystals using two beams with a wave-vector difference  $\Delta k$  coupling to the axial motion. The single  $^{171}\text{Yb}^+$  axial center-of-mass mode frequency is 1.0 MHz, and we use the first higher-order mode at 1.74 MHz for entangling operations. Before any two-qubit gate operations, motional excitations are cooled out of the system using a combination of Doppler and resolved sideband cooling on the  $^{138}\text{Ba}^+$  ions.

We characterize physical qubit operations using error rates extracted from parallel randomized benchmarking

[99]. Physical qubits are initialized and measured using standard optical pumping and state-dependent fluorescence techniques [100] with an average SPAM error of  $2.4(9) \times 10^{-3}$ , where the uncertainty indicates the typical variations across the zones and over time. During qubit measurement and reset, laser scatter and ion fluorescence absorbed by idle qubits cause errors ranging from  $<1.0 \times 10^{-4}$  to  $3 \times 10^{-3}$  depending on the measurement zone and the other qubits' physical locations. Single-qubit and two-qubit gate errors are, respectively, measured to be  $7(1) \times 10^{-5}$  and  $3.0(1) \times 10^{-3}$ . The memory error in a depth-one circuit is benchmarked at  $<5(3) \times 10^{-4}$ . Our estimates for errors in transport operations, such as physical swap, are negligible compared to other errors.

We compile OpenQASM to a hardware-specific pulse language that combines electrode and laser control. The sequences are executed with arbitrary waveform generators driving electrode voltages and direct digital synthesizers components controlling laser pulses, both sequenced by a distributed network of controllers which combine programmable logic components with ARM microprocessors. The microprocessor allows for sequencing the error corrections as optional laser pulses embedded in a fixed ion-transport schedule. To execute conditional gates, qubit measurements are analyzed and broadcast by a single microprocessor. Typically, the conditional information is distributed in parallel with unconditional circuit operations, and in worst-case conditions adds  $<250 \mu\text{s}$  of distribution latency.

## APPENDIX B: FIT EQUATION

The decay equation used to fit to logical error rates and presented in the captions of Figs. 4 and 5 is derived from coupled recursive relation equations. As an example and without loss of generality, if the  $|0\rangle_L$  state is chosen and initialization and measurement is done with fidelity  $1 - p_{\text{SPAM}}$ , and each QEC cycle preserves  $|0\rangle_L$  with probability  $1 - p_{\text{cycle}}$  and flips the measurement outcome to  $|1\rangle_L$  with probability  $p_{\text{cycle}}$ , the measurement dynamics are described by the equations

$$\begin{aligned} p_0(c+1) &= (1 - p_{\text{cycle}})p_0(c) + p_{\text{cycle}}p_1(c), \\ p_1(c+1) &= (1 - p_{\text{cycle}})p_1(c) + p_{\text{cycle}}p_0(c), \end{aligned} \quad (\text{B1})$$

with the initial conditions being

$$\begin{aligned} p_0(0) &= 1 - p_{\text{SPAM}}, \\ p_1(0) &= p_{\text{SPAM}}. \end{aligned} \quad (\text{B2})$$

Here, the variables  $p_0(c)$  and  $p_1(c)$  give the respective probabilities of measurement outcomes  $|0\rangle_L$  and  $|1\rangle_L$  as a function of the number of QEC cycles  $c$ . The solutions to these equations are  $p_0(c) = 0.5 - (p_{\text{SPAM}} - 0.5)(1 - 2p_{\text{cycle}})^c$  and  $p_1(c) = 0.5 + (p_{\text{SPAM}} - 0.5)(1 - 2p_{\text{cycle}})^c$

as shown in the main text for an arbitrary state initialization. Note that since  $p_1(c)$  indicates a logical error has occurred, since the state started as  $|0\rangle_L$  in the derivation,  $p_1(c)$  is equivalent to  $p_L(c)$ .

It is interesting to note that the decay equation can also be derived from the continuous decay curve,

$$p_L(c) = \frac{1}{2}[1 - a \exp(-bc)], \quad (\text{B3})$$

where  $a$  and  $b$  are fit parameters. One can then solve for  $p_{\text{SPAM}}$  and  $p_{\text{cycle}}$  by noting that

$$\begin{aligned} p_{\text{SPAM}} &= p_L(0) = \frac{1}{2}(1 - a), \\ p_{\text{cycle}} &= p_L(1)|_{a=1} = \frac{1}{2}[1 - \exp(-b)]. \end{aligned} \quad (\text{B4})$$

One then finds that

$$\begin{aligned} a &= 1 - 2p_{\text{SPAM}}, \\ b &= -\ln(1 - 2p_{\text{cycle}}). \end{aligned} \quad (\text{B5})$$

And after substituting Eqs. (B5) into Eq. (B3), one recovers

$$p_L(c) = 0.5 + (p_{\text{SPAM}} - 0.5)(1 - 2p_{\text{cycle}})^c, \quad (\text{B6})$$

the same logical decay equation recovered in the discrete case and mentioned in the captions of Figs. 4 and 5.

## APPENDIX C: ADDITIONAL ANALYSIS OF EXPERIMENTAL PERFORMANCE

In this Appendix, we provide additional details on the performance of the QEC cycles experiments that the reader might find informative.

As discussed in Sec. II A, the Pauli frame corrections determined in the QEC cycles experiments were stored and tracked in software. At the end of the computation (see Sec. II A 4), the correction was combined with the measurement of the logical operator and compared to the expected result (corresponding logical state intended to be initialized). By the end of each QEC computation, numerous values are stored in classical registers. This includes the expected logical outcome, the uncorrected logical measurement outcome, the corrected logical measurement outcome, and the final Pauli frame correction. (Note that the syndrome history is not stored; therefore, we do not have the data necessary to determine corrections after the quantum computation.) Since both the uncorrected and corrected logical outcomes are stored by the quantum program, we can then compare the error rates as seen in Fig. 8. While this is not a comparison of the performance of a physical algorithm to a logical one, we see from the figure that the corrections as determined by real-time decoding resulted in a net reduction in logical error rate compared to running the algorithm without decoding.

Corrected versus uncorrected experimental logical error rates of color code QEC cycles for each basis

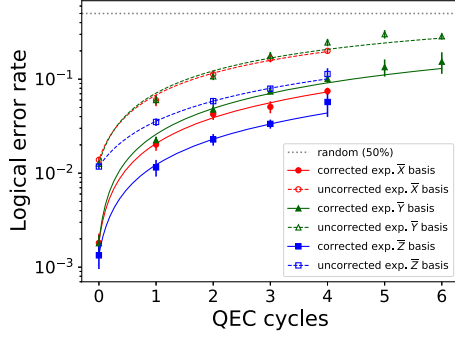


FIG. 8. Comparing the observed logical error rates for both uncorrected and corrected logical bases results over many QEC cycles. Averages (points) and standard deviations (error bars) were determined by jackknife resampling between individual experiments [68]. The lines are fits to the experimental averages, where the fits are exponential decay curves  $p_L(c) = 0.5 + (p_{\text{SPAM}} - 0.5)(1 - 2p_{\text{cycle}})^c$  (see Appendix B for derivation). Here  $p_L(c)$  is the logical error rate of a cycle  $c$  and logical basis  $L$ ,  $p_{\text{SPAM}}$  the logical SPAM error, and  $p_{\text{cycle}}$  the logical QEC cycle error. Note that  $p_{\text{SPAM}}$  is determined directly from the 0 QEC cycle results and fixed when determining the fits.

In addition to the above classical information that is stored in classical register, the branching taken during each QEC cycle was also tracked. That is, the branching is the conditional circuit path taken in multiple stages of syndrome extraction in a QEC cycle as discussed in Sec. II A 2 and depicted in Fig. 10. Using the labeling of the stages of syndrome extraction as given in Fig. 10, we can define three paths taken through a QEC cycle. The first path “A” is when no changes in flagged syndrome measurements are detected. This is the only path where the final round of syndrome extraction is not triggered and can be labeled as  $\{S_1^f, S_5^f, S_6^f\} \rightarrow \{S_2^f, S_3^f, S_4^f\}$ . The second path “B” is when a change in flagged syndrome measurements is detected by the first triple of measured stabilizer and is labeled  $\{S_1^f, S_5^f, S_6^f\} \rightarrow \{S_1, S_2, S_3, S_4, S_5, S_6\}$ . The final path “C” occurs when the second set of flagged syndrome measurements detected a change. This path includes the maximum number of syndrome extraction and can be labeled  $\{S_1^f, S_5^f, S_6^f\} \rightarrow \{S_2^f, S_3^f, S_4^f\} \rightarrow \{S_1, S_2, S_3, S_4, S_5, S_6\}$ .

As shown in Fig. 9, the probability per QEC cycle of taking the path A is roughly 77% while the probability of taking path B is approximately 12% and the probability of taking C is about 10%. This indicates for each cycle the majority of the time no change in physical error is detected and with approximately equal probability a change in error is found by either the first set of stabilizer measure or the second. Figure 9 also indicates a slight decrease in the probability of path A with increasing QEC cycles, suggesting a small increase in the probability of error with increased circuit depth. The error dependence on the circuit depth is not yet well understood and requires further study.

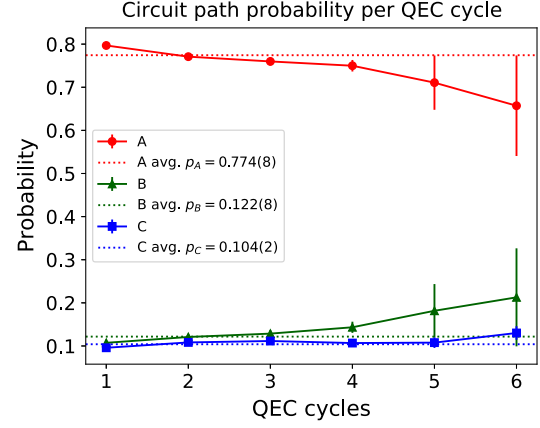


FIG. 9. Comparing the circuit path taken through the QEC cycles. Data averages and standard deviations were determined by jackknife resampling between individual experiments [68]. Horizontal dotted lines correspond to the probability of taking the indicated path after combining the data regardless of number of QEC cycles. Labels A, B, and C are defined in Appendix C.

## APPENDIX D: SIMULATIONS AND MODELING

The simulator used in this work is a modified version of the software PECOS [74] that receives instructions directly from the compiler, which is composed of the native quantum gate set, classical operations, and transport operations. That is, the instructions that were used to encode the

TABLE VI. Simulation error parameters. Except for the dephasing rate, the parameters used come from experimental measurement or well-defined microscopic noise analysis. Values based on microscopic noise analysis include further break-down of the errors such as initialization error consisting of 1/3 bit-flip errors and 2/3 leakage, and spontaneous emission consisting of 1/2 leakage error, 1/4 X error, and 1/4 Z error. The dephasing rates used in the simulations were determined separately for the incoherent and coherent simulations by adjusting the dephasing rate until the logical error rate for a QEC cycle matched the value found from the experiment.

Operation	Channel	Probability
Initialization	Bit flip	$1.66 \times 10^{-6}$
	Leakage	$3.33 \times 10^{-6}$
X/Y Single-qubit gate	Depolarizing	$7 \times 10^{-5}$
	Spontaneous emission	$1.25 \times 10^{-5}$
Two-qubit gate	Depolarizing	$3.1 \times 10^{-3}$
	Spontaneous emission	$5.5 \times 10^{-4}$
Measurement	Bit flip	$2.4 \times 10^{-3}$
Cross talk	Initialization	$2.3 \times 10^{-5}$
	Measurement	$2.3 \times 10^{-4}$
Operation	Channel	Rate (Hz)
Dephasing	Coherent	0.26
	Incoherent	0.43

QEC experiments in this paper and were executed by the simulator were the same instructions that the QCCD device was instructed to perform (before being translated to hardware-specific pulses). The simulator models errors both coherently using a state-vector back end [101] as well as incoherently using a stabilizer simulation [66]. All simulation results presented in this paper are coherent simulations; however, both the coherent and incoherent simulations produce nearly identical results given a dephasing rate of 0.26 and 0.43 Hz, respectively.

The error model includes simple depolarizing gate noise, leakage errors, and dephasing noise during transport and cooling operations. Errors on physical qubits are modeled as stochastic processes (excluding dephasing). Most errors are applied with probabilities determined by independent experiments as summarized in Table VI, except for dephasing errors. The coherent simulation modeled dephasing as  $RZ(\theta)$  following ideal gates, where  $\theta$  is the dephasing rate, 0.26 Hz times the time between gates due to qubit idling or transport operations. For the incoherent simulations, dephasing errors are applied as stochastic  $Z$  errors between gate operations with a probability of  $p = \sin(\theta/2)^2$ . That is, the probability obtained from Pauli twirling the coherent channel. Note that here the same  $\theta$  is used as the coherent model. As discussed in the main text, both dephasing rates (0.26 and 0.43 Hz) were empirically found.

Physical qubit initialization to  $|0\rangle$  occurs at the beginning of each circuit and after each measurement. The fidelity of this procedure is limited by off-resonant coupling, which is numerically simulated. The residual population remains in the  $F = 1$  manifold,  $2/3$  of which is distributed in the

leakage states  $|F = 1, m_f = \pm 1\rangle$ . The single-qubit and two-qubit gate errors are modeled as being dominated by a depolarization process whose amplitude is measured via randomized benchmarking experiments. The spontaneous emission that occurs during stimulated Raman transitions is estimated through atomic physics calculations closely following Ref. [102]. While the spontaneous emission is estimated to be small compared to the total error, the leakage induced by the process is more detrimental than errors that keep the ion in the qubit subspace. We, therefore, explicitly include this error in addition to the depolarizing error. Note that the spontaneous emission process is modeled as causing leakage with probability  $1/2$ , and causing  $X$  and  $Y$  errors each with probability  $1/4$ . Single-qubit rotations about the  $z$  axis are done entirely in software, and therefore contribute negligible errors. We model measurement errors as bit flips and note that leaked qubits return measurement results that are indistinguishable from  $|1\rangle$  due to the measurement process described in Ref. [100].

## APPENDIX E: FLOW CHARTS

Here we present in Figs. 10–13 an overview of the circuits and the control flow including a detailed schematic of the QEC cycle, the active correction experiment, and the non-FT encoding circuit used in the magic state preparation. Additionally, we provide examples of two different error configurations in an effort to better explain the two-stage decoder used.

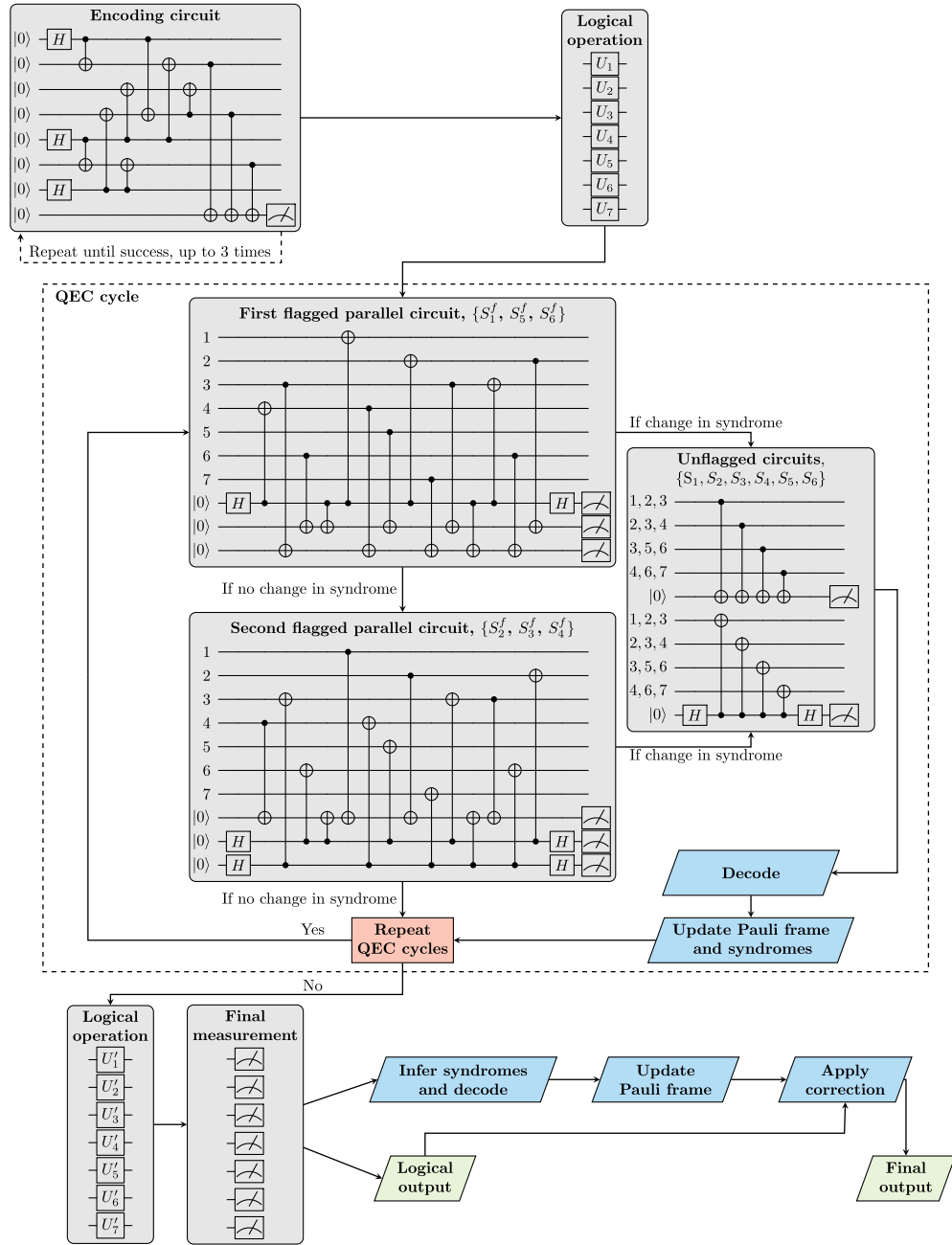


FIG. 10. A detailed schematic of the QEC demonstration. The encoding circuit first prepares  $|0\rangle_L$  and uses an ancilla to verify this preparation. If the ancilla measured is found in  $|0\rangle$ , the circuit succeeded and moves to the next step. If the ancilla is found to be in  $|1\rangle$ , then all qubits are reinitialized and the circuit ran again until successful, up to 3 times. After a maximum of three initialization attempts, we proceed by applying single-qubit unitaries to make a logical rotation to prepare the desired logical state. Adaptive QEC cycles are then performed. The first set of three syndromes is measured using the flag circuit  $\{S_1^f, S_2^f, S_3^f\}$ . If these syndromes do not indicate an error, then the second set of three syndromes is measured  $\{S_2^f, S_3^f, S_4^f\}$ , and if no errors are indicated, the syndrome extraction protocol is complete. If either of the flagged circuits does indicate an error, the protocol moves to measure all six stabilizers using the unflagged circuit  $\{S_1, S_2, S_3, \dots\}$ , thus completing the syndrome extraction. The syndrome information is then fed to the decoder to infer a correction, and the Pauli frame is updated. Finally, the state is rotated to the appropriate basis for measurement using single-qubit unitaries. Upon measurement, syndrome information is inferred and sent to the decoder for a final correction.

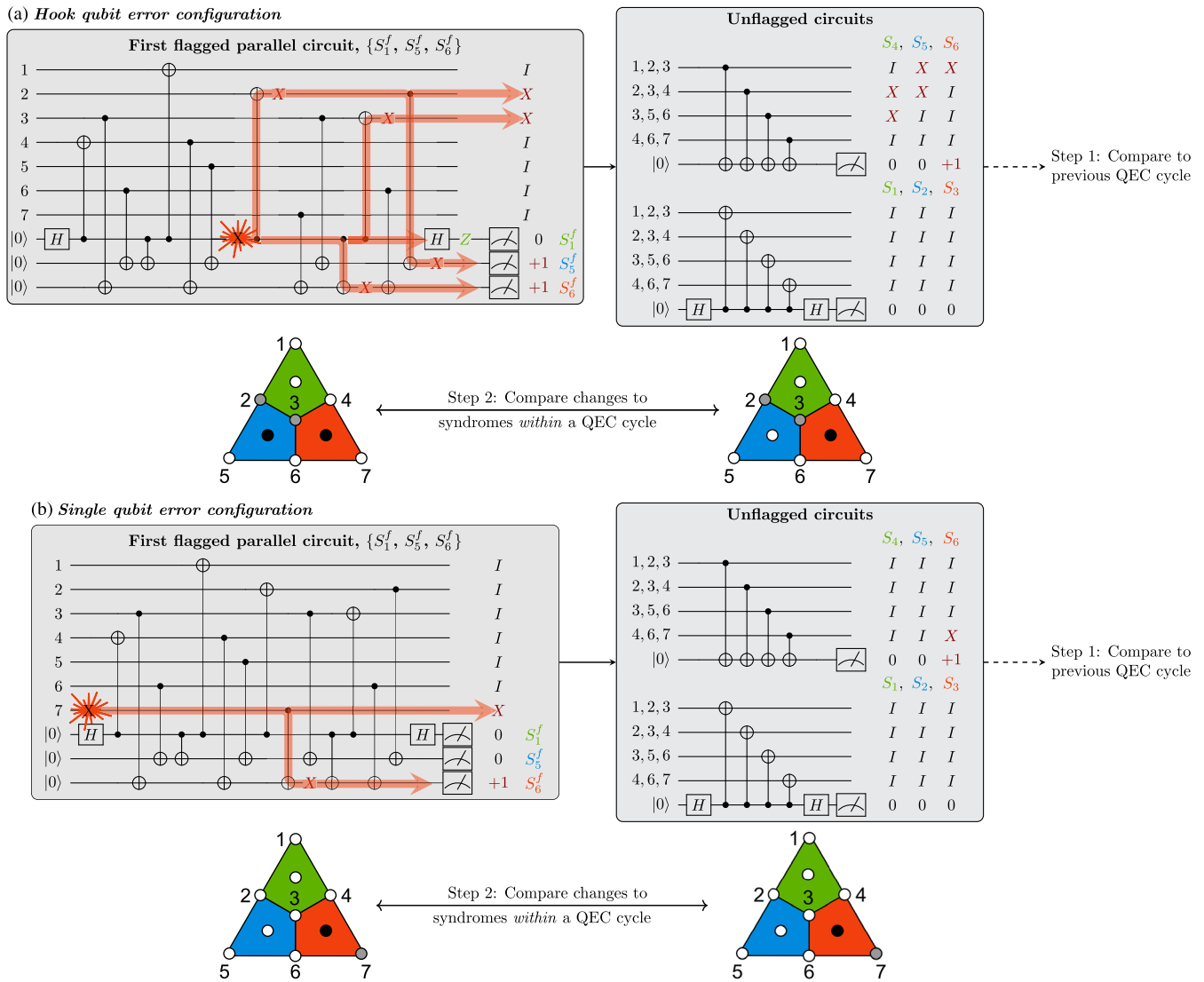


FIG. 11. An outline of the decoder procedure and two different error configurations. (a) A single-qubit hook error occurs on an ancilla and spreads to two data qubits. This triggers the additional round of syndrome extraction using the unflagged circuits. We denote the errors on the data qubits (gray circles) and the changed syndrome measurements (black circles) pictorially on the color code figure. (b) A single-qubit error occurs on a data qubit and causes a change in the syndrome measurements, again triggering an additional round of syndrome extraction. In both (a) and (b) the final round of syndrome measurements are identical, meaning the first step of the decoder cannot distinguish between these two error configurations. The second step of the decoder compares the syndromes measured within the QEC cycle to distinguish between cases such as this. We see that the two sets of measured syndromes are different for the hook error but the same for the single-qubit data error.



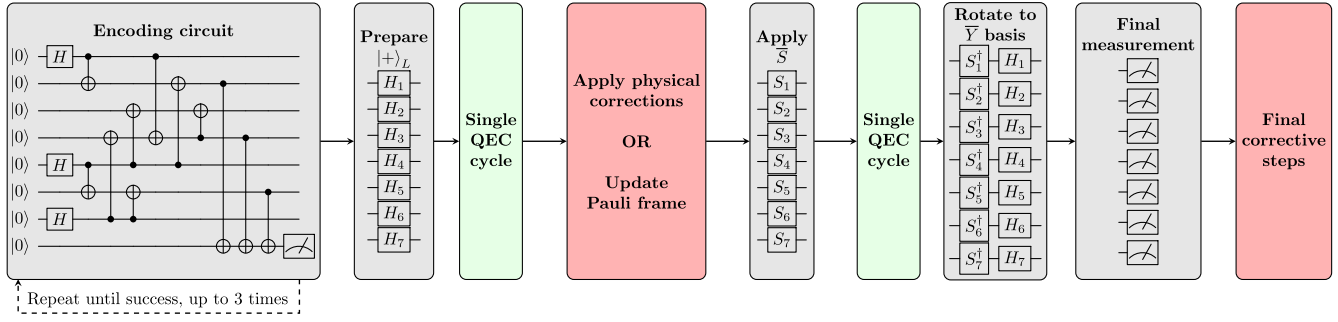


FIG. 12. A schematic of the  $\bar{S}$  gate experiments. In this set of experiments, we compare physically applied corrections to implementing the corrections in software using the Pauli frame. The main elements are the same as those used in Fig. 10. First, using the FT encoding circuit, we prepare  $|0\rangle_L$  and rotate to  $|+\rangle_L$  with an  $\bar{H}$  gate. We then perform one QEC cycle to generate corrections. Next, we either physically apply corrections to the qubits or update the Pauli frame in software, followed by physically applying the  $\bar{S}$  gate. We then apply one last QEC cycle to generate further corrections, rotate to the  $\bar{Y}$  basis, and measure the data qubits. The final corrective steps infer the syndromes, decode, and apply corrections to the output data as outlined in the main text and Fig. 10.

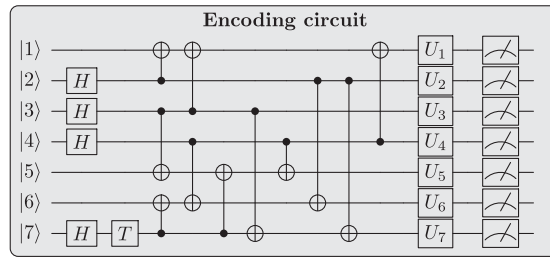


FIG. 13. A non-fault-tolerant encoding circuit used to prepare the magic state  $\bar{T}|+\rangle_L$ .

## APPENDIX F: PSEUDOCODE

In this Appendix, we present pseudocode in Figs. 14–22 in the style of PYTHON to more clearly describe the specifics of the hybrid quantum and classical programs ran as experiments in this paper.

```

1  def qec_cycles_exp(init_state: str, meas_basis: str, num_cycles: int, shots: int) -> Tuple
2      ↪ [int, int]:
3      """
4      Pseudocode representing the repeated QEC cycle and the magic-state experiments.
5      Args:
6      init_state: The logical state to initialize.
7      meas_basis: The logical measurement basis to measure in the end.
8      num_cycles: The number of QEC cycles.
9      shots: The number of experiments to attempt.
10
11     Returns:
12     Number of shots and the number outcomes consistent with state preparation.
13     """
14     runs = 0 # Number of total shots
15     success = 0 # Number of shots that had expected outcomes
16
17     for i in range(shots):
18         # Initialize the logical state
19         if init_state == '|T>':
20             # Encode T|+> using Steane code encoding circuit from Ref. [70]
21             state = prep_tstate()
22         else:
23             # Use FT circuit from Ref. [61] to prepare logical |0> (see Fig. 10) given 3
24             ↪ verification attempts.Continue regardless of final success
25             for _ in range(3):
26                 state, prep_error = prep_zero()
27                 if prep_error == 0:
28                     break # break out of for loop if successful
29
30             # Rotate that state from logical |0> to another logical state as appropriate
31             state = logical_rotate_init_basis(state, init_state)
32
33             # Run the QEC cycles
34             last_syndromes_x = [0, 0, 0]
35             last_syndromes_z = [0, 0, 0]
36             pf = [0, 0] # The Pauli frame: [Apply logical X?, Apply logical Z?]
37
38             for j in range(num_cycles):
39                 state, last_syndromes_x, last_syndromes_z, pf = qec_cycle(state, last_syndromes_x,
40                 ↪ last_syndromes_z, pf)
41
42             # Rotate the logical measurement basis
43             state = rotate_meas_basis(state, meas_basis)
44
45             # Do a destructive logical measurement
46             meas_output = logical_meas(state, meas_basis, last_syndromes_x, last_syndromes_z, pf)
47
48             # Determine if measurement result was as expected. If so, upade success count.
49             expected_outcome = expected_result(meas_output, init_state, meas_basis)
50             success += expected_outcome # 1 or 0 (expected or not)
51             runs += 1
52
53     return runs, success

```

FIG. 14. The main program for the repeated QEC cycle and the magic state experiment.

```

1 def active_corr_exp(active_correction: bool, shots: int) -> Tuple[int, int]:
2     """
3     Pseudocode representing the active vs software experiments.
4
5     Args:
6     active_correction: Whether to physically apply the logical corrections before the
7         ↪ logical S gate or not.
8     shots: The number of experiments to attempt.
9
10    Returns:
11    A tuple of the number of shots and the number of shots with measurement outcomes
12        ↪ consistent with state preparation.
13    """
14
15    runs = 0 # Number of total shots
16    success = 0 # Number of shots that had expected outcomes
17
18    for i in range(shots):
19
20        # Initialize the |+> logical state
21        # Use FT circuit from Ref. [61] to prepare logical |0> (see Fig. 10) given 3
22            ↪ verification attempts. Continue regardless of final success
23
24        for _ in range(3):
25            state, prep_error = prep_zero()
26            if prep_error == 0:
27                break # if verification is successful, break out of for loop and continue
28
29        # Rotate that state from logical |0> to logical |+>
30        state = logical_rotate_init_basis(state, init_state='|+>')
31
32        # Run the QEC cycles
33        last_syndromes_x = [0, 0, 0]
34        last_syndromes_z = [0, 0, 0]
35        pf = [0, 0] # The Pauli frame: [Apply logical X?, Apply logical Z?]
36
37        # Do a single QEC cycle
38        state, last_syndromes_x, last_syndromes_z, pf = qec_cycle(state, last_syndromes_x,
39            ↪ last_syndromes_z, pf)
40
41        if active_correction: # Physically apply any logical X or Z corrections
42            # We apply both X and Z; however, you don't have to apply Z since S and Z commute
43            state = apply_pauli_frame(state, pf)
44            pf = [0, 0] # Reset Pauli frame
45        else:
46            # Push Pauli frame through the S gate: P S = S (S† P S) = S P'
47            # Z -> Z, X -> Y
48            pf = sgate2pf(pf) # equivalent to XORing: pf[1] = pf[1] ^ pf[0]
49
50        # Physically apply the logical S gate
51        state = logical_sgate(state)
52
53        # Rotate the logical measurement basis to Y since S: |+> -> |+i>
54        state = rotate_meas_basis(state, meas_basis='Y')
55        # Do a destructive logical measurement
56        meas_output = logical_meas(state, meas_basis, last_syndromes_x, last_syndromes_z, pf)
57
58        runs += 1
59
60        # Determine if measurement result was as expected. If so, upade success count.
61        expected_outcome = expected_result(meas_output, init_state, meas_basis)
62        success += expected_outcome
63
64    return runs, success

```

FIG. 15. The main program for the active versus software correction experiment with the logical S.

```

1 def logical_rotate_init_basis(state: QuantumState, init_state: str) -> QuantumState:
2     """
3     Pseudocode representing rotating to appropriate logical basis.
4
5     Args:
6     state: The logical state to rotate.
7     init_state: The logical state to rotate to.
8
9     Returns:
10    A rotated logical state.
11    """
12
13    if init_state == '|0>':
14        # |0> -> |0>
15        pass # do nothing
16
17    elif init_state == '|1>':
18        # X |0> -> |1>
19        state = logical_x(state) # X on qubits 5, 6, and 7 (|0> -> |1>)
20
21    elif init_state == '|+>':
22        # H |0> -> |+>
23        state = logical_h(state) # H on all data qubits
24
25    elif init_state == '|->':
26        # H X |0> -> |->
27        state = logical_x(state) # X on qubits 5, 6, and 7 (|0> -> |1>)
28        state = logical_h(state) # H on all data qubits (|1> -> |->)
29
30    elif init_state == '|+i>':
31        # S H |0> -> |+i>
32        state = logical_h(state) # H on all data qubits (|0> -> |+>)
33        state = logical_s(state) # S† on all data qubits (|+> -> |+i>)
34
35    elif init_state == '|-i>':
36        # S H X |0> -> |-i>
37        state = logical_x(state) # X on qubits 5, 6, and 7 (|0> -> |1>)
38        state = logical_h(state) # H on all data qubits (|1> -> |->)
39        state = logical_s(state) # S† on all data qubits (|-> -> |-i>)
40
41    return state

```

FIG. 16. Rotate logical  $|0\rangle$  to the appropriate state.

```

1 def qec_cycle(state: QuantumState, last_syndrome_x: List[int, int, int], last_syndrome_z:
    ↪ List[int, int, int], pf: List[int, int]) -> Tuple[QuantumState, List[int, int, int],
    ↪ List[int, int, int], List[int, int]]:
2     """
3     Pseudocode representing a single QEC cycle.
4
5     Args:
6     init_state: The logical state to initialize.
7     meas_basis: The logical measurement basis to measure in the end.
8     num_cycles: The number of QEC cycles.
9     shots: The number of experiments to attempt.
10
11     Returns:
12     A logical state, the last X-type syndromes measured (not including flagged syndromes),
    ↪ the last Z-type syndromes measured (not including flagged syndromes), and the
    ↪ updated Pauli-frame, which tracks whether to apply a logical X and/or Z
    ↪ correction.
13     """
14     flag_diff_x = [0, 0, 0]
15     flag_diff_z = [0, 0, 0]
16
17     # Measure the first set of stabilizers in parallel.
18     fx0, fz1, fz2 = meas_flagging_syndromes_xzz(state)
19     # XOR with previously measured syndrome to get the difference
20     flag_diff_x[0] = fx0 ^ last_syndrome_x[0]
21     flag_diff_z[1] = fz1 ^ last_syndrome_z[1]
22     flag_diff_z[2] = fz2 ^ last_syndrome_z[2]
23
24     # If no change detected, go on to the next flagging syndromes
25     if flag_diff_x == [0, 0, 0] and flag_diff_z == [0, 0, 0]:
26         # Measure the second set of stabilizers in parallel.
27         fz0, fx1, fx2 = meas_flagging_syndromes_zxx(state)
28         # XOR with previously measured syndrome to get the difference
29         flag_diff_z[0] = fz0 ^ last_syndrome_z[0]
30         flag_diff_x[1] = fx1 ^ last_syndrome_x[1]
31         flag_diff_x[2] = fx2 ^ last_syndrome_x[2]
32
33     # If any change in syndromes was detected, re-measure the set of six stabilizer
    ↪ generators without flagging
34     if flag_diff_x != [0, 0, 0] or flag_diff_z != [0, 0, 0]:
35         sx0, sx1, sx2, sz0, sz1, sz2 = meas_six_syndromes(state)
36         syndromes_x = [sx0, sx1, sx2]
37         syndromes_z = [sz0, sz1, sz2]
38         # Get the change in syndromes
39         syndrome_diff_x = bitwise_xor(syndromes_x, last_syndrome_x)
40         syndrome_diff_z = bitwise_xor(syndromes_z, last_syndrome_z)
41
42         # Determine a Pauli-frame update to track correction in software
43         pf_new_x = decoder_2d(syndrome_diff_x)
44         pf_new_z = decoder_2d(syndrome_diff_z)
45         # Now modify the 2D decoder corrections based on flagging
46         pf_flag_new_x = decoder_flag_update(syndrome_diff_x, flag_diff_x)
47         pf_flag_new_z = decoder_flag_update(syndrome_diff_z, flag_diff_z)
48         # Update the Pauli frame with the new corrections vis XORing
49         pf[0] = pf[0] ^ pf_new_x ^ pf_flag_new_x
50         pf[1] = pf[1] ^ pf_new_z ^ pf_flag_new_z
51
52         # Set the last syndromes to the current ones
53         last_syndrome_x = last_syndrome_x
54         last_syndrome_z = last_syndrome_z
55
56     return state, last_syndrome_x, last_syndrome_z, pf

```

FIG. 17. A QEC cycle.

```
1 def rotate_meas_basis(state: QuantumState, meas_basis: str) -> QuantumState:
2     """
3     Rotate logical measurement basis.
4
5     Args:
6         state: The logical state.
7         meas_basis: The logical measurement basis to measure in the end.
8
9     Returns:
10        A logical state.
11    """
12    if meas_basis == 'Z':
13        pass # do nothing. we measure in Z by default
14
15    elif meas_basis == 'X':
16        state = logical_h(state) # H on all data qubits
17
18    elif meas_basis == 'Y':
19        state = logical_sdg(state) # S on all data qubits
20        state = logical_h(state) # H on all data qubits
21
22    return state
```

FIG. 18. Rotate measurement to the correct logical basis.

```

1 def logical_meas(state: QuantumState, meas_basis: str, last_syndrome_x: List[int, int, int
  ↪ ], last_syndrome_z: List[int, int, int], pf: List[int, int]) -> int:
2     """
3     Destructive logical measurement.
4
5     Args:
6     state: The logical state to measure.
7     meas_basis: The logical measurement basis being measured in.
8     last_syndrome_x: The last X-type syndromes measured (not including flagged ones)
9     last_syndrome_z: The last Z-type syndromes measured (not including flagged ones)
10
11    Returns:
12    Logical measurement outcome.
13    """
14    # Note: The logical measurement basis might have rotated outside of this function.
15
16    # Measure each data qubit with a single-qubit Z-basis measurement.
17    # Measurement labeling corresponds to the labeling of data qubits in Fig. 10
18    m1, m2, m3, m4, m5, m6, m7 = meas_z_data(state) # get measurement results
19
20    # Due to symmetry we can treat the X, Y, and Z basis in a similar manner:
21
22    # Get logical measurement output of logical X, Y, or Z (as appropriate)
23    meas_output = m5 ^ m6 ^ m7 # XOR measurements of logical operator on qubits 5, 6, 7
24
25    # Get syndromes by XORing measurement bits together
26    # Note: Due to the meas. basis, these results may be for X, Y, or Z type stabilizers
27    s1 = m1 ^ m2 ^ m3 ^ m4
28    s2 = m2 ^ m3 ^ m5 ^ m6
29    s3 = m3 ^ m4 ^ m6 ^ m7
30    syndromes = [s1, s2, s3]
31
32    # Get change of syndrome depending on measurement basis
33    if meas_basis == 'X':
34        syndrome_diff = bitwise_xor(syndromes, last_syndrome_x)
35    elif meas_basis == 'Y':
36        syndrome_diff = bitwise_xor(syndromes, last_syndrome_x)
37        syndrome_diff = bitwise_xor(syndromes, last_syndrome_z)
38    elif meas_basis == 'Z':
39        syndrome_diff = bitwise_xor(syndromes, last_syndrome_z)
40    else:
41        raise Exception() # No other basis expected
42
43    # Get correction from the syndrome diff as determined from measurement
44    final_correction = decoder_2d(syndrome_diff)
45
46    # Update logical measurement outcome based on the measured syndromes
47    meas_output = meas_output ^ final_correction # XOR together => flip outcome or not
48
49    # Apply correction from Pauli frame to measurement outcome
50    if meas_basis == 'X':
51        meas_output = meas_output ^ pf[1] # Apply logical Z
52    elif meas_basis == 'Y':
53        meas_output = meas_output ^ pf[0] # Apply logical X
54        meas_output = meas_output ^ pf[1] # Apply logical Z
55    elif meas_basis == 'Z':
56        meas_output = meas_output ^ pf[0] # Apply logical X
57    else:
58        raise Exception() # No other basis expected
59
60    return meas_output

```

FIG. 19. A measurement in the logical Z basis.

```

1 def expected_result(meas_output: int, init_state: str, meas_basis: str) -> int:
2     """
3     Determining whether a measurement result is consistant with expectation.
4
5     Args:
6     meas_output: Logical measurement output after correction. 0 => +1, 1 => -1
7     init_state: The logical state intended to be initialized.
8     meas_basis: The logical measurement basis measured in.
9
10    Returns:
11    A logical state.
12    """
13    # Measurement basis and state pairs that should have 0/+1 outcomes
14    zero_outcomes = (('X', '|+>'), ('Y', '|+i>'), ('Z', '|0>'),
15                    ('X', '|T>'), ('Y', '|T>'))
16    # Note, the measurements of the |T> state is not a measurement of fidelity but instead
17    # ↳ understood as the probability of being in the +1 eigenstate of the measured basis.
18
19    # Measurement basis and state pairs that should have 1/-1 outcomes
20    one_outcomes = (('X', '|->'), ('Y', '|-i>'), ('Z', '|1>'))
21
22    if (meas_basis, init_state) in zero_outcomes:
23        expected_val = 0
24
25    elif (meas_basis, init_state) in one_outcomes:
26        expected_val = 0
27
28    else:
29        # No other measurement basis and state pairs are expected to be evaluated.
30        raise Exception('Unexpect_init_state_and_measurement_basis_combination!')
31
32    if expected_val == meas_out:
33        success = 1
34    else:
35        success = 0
36
37    return success

```

FIG. 20. Determining if the result is expected.



```

1  def decoder_2d(syndrome_diff: List[int, int, int]) -> int:
2      """
3      A 2D decoder that determines logical corrections. Due to symmetry this, can be used
4          ↪ to decode stabilizer measurements of any Pauli type.
5
6      Args:
7          syndrome_diff: The change in syndromes compared to the last time they were
8              ↪ measured.
9
10     Returns:
11     A bit representing whether a logical error has occurred.
12     """
13     # Syndromes that indicate tha a single Pauli fault has flipped the logical operator
14     ↪ representative on data qubits 5, 6, and 7.
15     bad_syndromes = {[0, 1, 0], [0, 1, 1], [0, 0, 1]}
16
17     if syndrome_diff in bad_syndromes:
18         logical_error = 1
19     else:
20         logical_error = 0
21
22     return logical_error

```

FIG. 21. The basic 2D decoder that infers if a logical error has occurred.

```

1  def decoder_flag_update(syndrome_diff: List[int, int, int], flag_diff: List[int, int,
2      ↪ int]) -> int:
3      """
4      A lookup decoder used to modify corrections due hook errors indicated by changes in
5          ↪ flag and syndrome. Due to symmetry this, can be used to decode stabilizer
6          ↪ measurements of any Pauli type.
7
8      Args:
9          syndrome_diff: The change in syndromes compared to the last time they were measured
10             ↪ .
11          flag_diff: The change in flags compared to the last time syndromes were measured.
12
13     Returns:
14     A bit representing whether the 2D decoder's correction should be changed due to the
15     ↪ relationship between flags and syndromes.
16     """
17     # The following indicate hook faults have occurred:
18
19     # flag -> syndrome: 0 -> 1
20     if flag_diff == [1, 0, 0] and syndrome_diff == [0, 1, 0]:
21         change_correction = 1
22
23     # flag -> syndrome: 0 -> 2
24     elif flag_diff == [1, 0, 0] and syndrome_diff == [0, 0, 1]:
25         change_correction = 1
26
27     # flag -> syndrome: 1,2 -> 2
28     elif flag_diff == [0, 1, 1] and syndrome_diff == [0, 0, 1]:
29         change_correction = 1
30
31     else:
32         change_correction = 0
33
34     return change_correction

```

FIG. 22. An additional decoder that updates the correction based on flag information.

- [1] R. P. Feynman, *Quantum Mechanical Computers*, *Found. Phys.* **16**, 507 (1986).
- [2] P. W. Shor, *Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer*, *SIAM Rev.* **41**, 303 (1999).
- [3] P. W. Shor, *Algorithms for Quantum Computation: Discrete Logarithms and Factoring*, in *Proceedings of the 35th Annual Symposium on Foundations of Computer Science* (IEEE, Piscataway, NJ, 1994), pp. 124–134.
- [4] D. S. Abrams and S. Lloyd, *Simulation of Many-Body Fermi Systems on a Universal Quantum Computer*, *Phys. Rev. Lett.* **79**, 2586 (1997).
- [5] A. Aspuru-Guzik, A. D. Dutoi, P. J. Love, and M. Head-Gordon, *Simulated Quantum Computation of Molecular Energies*, *Science* **309**, 1704 (2005).
- [6] V. von Burg, G. H. Low, T. Häner, D. S. Steiger, M. Reiher, M. Roetteler, and M. Troyer, *Quantum Computing Enhanced Computational Catalysis*, *Phys. Rev. Research* **3**, 033055 (2021).
- [7] R. Orus, S. Mugel, and E. Lizaso, *Quantum Computing for Finance: Overview and Prospects*, *Rev. Phys.* **4**, 100028 (2019).
- [8] P. W. Shor, *Scheme for Reducing Decoherence in Quantum Computer Memory*, *Phys. Rev. A* **52**, R2493 (1995).
- [9] A. R. Calderbank and P. W. Shor, *Good Quantum Error-Correcting Codes Exist*, *Phys. Rev. A* **54**, 1098 (1996).
- [10] A. M. Steane, *Error Correcting Codes in Quantum Theory*, *Phys. Rev. Lett.* **77**, 793 (1996).
- [11] D. Aharonov and M. Ben-Or, *Fault-Tolerant Quantum Computation with Constant Error Rate*, *SIAM J. Comput.* **38**, 1207 (2008).
- [12] A. Y. Kitaev, *Quantum Error Correction with Imperfect Gates*, *Quantum Communication, Computing, and Measurement* (Springer, New York, 1997), pp. 181–188.
- [13] E. Knill, R. Laflamme, and W. Zurek, *Threshold Accuracy for Quantum Computation*, [arxiv:quant-ph/9610011](https://arxiv.org/abs/quant-ph/9610011).
- [14] D. P. DiVincenzo, *The Physical Implementation of Quantum Computation*, *Fortschr. Phys.* **48**, 771 (2000).
- [15] D. Gottesman, *Stabilizer Codes and Quantum Error Correction*, [arxiv:quant-ph/9705052](https://arxiv.org/abs/quant-ph/9705052).
- [16] D. Gottesman, *An Introduction to Quantum Error Correction and Fault-Tolerant Quantum Computation*, in *Proceedings of Symposia in Applied Mathematics* (American Mathematical Society, Providence, RI, 2010), Vol. 68, pp. 13–58.
- [17] D. Gottesman, *Theory of Fault-Tolerant Quantum Computation*, *Phys. Rev. A* **57**, 127 (1998).
- [18] Z. Chen, K. J. Satzinger, J. Atalaya, A. N. Korotkov, A. Dunsworth, D. Sank, C. Quintana, M. McEwen, R. Barends, P. V. Klimov *et al.* (Google Quantum AI), *Exponential Suppression of Bit or Phase Errors with Cyclic Error Correction*, *Nature (London)* **595**, 383 (2021).
- [19] D. G. Cory, M. D. Price, W. Maas, E. Knill, R. Laflamme, W. H. Zurek, T. F. Havel, and S. S. Somaroo, *Experimental Quantum Error Correction*, *Phys. Rev. Lett.* **81**, 2152 (1998).
- [20] J. Chiaverini, D. Leibfried, T. Schaetz, M. D. Barrett, R. Blakestad, J. Britton, W. M. Itano, J. D. Jost, E. Knill, C. Langer *et al.*, *Realization of Quantum Error Correction*, *Nature (London)* **432**, 602 (2004).
- [21] P. Schindler, J. T. Barreiro, T. Monz, V. Nebendahl, D. Nigg, M. Chwalla, M. Hennrich, and R. Blatt, *Experimental Repetitive Quantum Error Correction*, *Science* **332**, 1059 (2011).
- [22] J. Cramer, N. Kalb, M. A. Rol, B. Hensen, M. S. Blok, M. Markham, D. J. Twitchen, R. Hanson, and T. H. Taminiau, *Repeated Quantum Error Correction on a Continuously Encoded Qubit by Real-Time Feedback*, *Nat. Commun.* **7**, 11526 (2016).
- [23] M. D. Reed, L. DiCarlo, S. E. Nigg, L. Sun, L. Frunzio, S. M. Girvin, and R. J. Schoelkopf, *Realization of Three-Qubit Quantum Error Correction with Superconducting Circuits*, *Nature (London)* **482**, 382 (2012).
- [24] D. Riste, S. Poletto, M.-Z. Huang, A. Bruno, V. Vesterinen, O.-P. Saira, and L. DiCarlo, *Detecting Bit-Flip Errors in a Logical Qubit Using Stabilizer Measurements*, *Nat. Commun.* **6**, 6983 (2015).
- [25] D. Ristè, L. C. Govia, B. Donovan, S. D. Fallick, W. D. Kalfus, M. Brink, N. T. Bronn, and T. A. Ohki, *Real-Time Processing of Stabilizer Measurements in a Bit-Flip Code*, *npj Quantum Inf.* **6**, 71 (2020).
- [26] J. R. Wootton and D. Loss, *Repetition Code of 15 Qubits*, *Phys. Rev. A* **97**, 052313 (2018).
- [27] N. M. Linke, M. Gutierrez, K. A. Landsman, C. Figgatt, S. Debnath, K. R. Brown, and C. Monroe, *Fault-Tolerant Quantum Error Detection*, *Sci. Adv.* **3**, 1701074 (2017).
- [28] M. Takita, A. W. Cross, A. D. Córcoles, J. M. Chow, and J. M. Gambetta, *Experimental Demonstration of Fault-Tolerant State Preparation with Superconducting Qubits*, *Phys. Rev. Lett.* **119**, 180501 (2017).
- [29] A. D. Córcoles, E. Magesan, S. J. Srinivasan, A. W. Cross, M. Steffen, J. M. Gambetta, and J. M. Chow, *Demonstration of a Quantum Error Detection Code Using a Square Lattice of Four Superconducting Qubits*, *Nat. Commun.* **6**, 6979 (2015).
- [30] R. Harper and S. T. Flammia, *Fault-Tolerant Logical Gates in the IBM Quantum Experience*, *Phys. Rev. Lett.* **122**, 080504 (2019).
- [31] J. Kelly, R. Barends, A. G. Fowler, A. Megrant, E. Jeffrey, T. C. White, D. Sank, J. Y. Mutus, B. Campbell, Y. Chen *et al.*, *State Preservation by Repetitive Error Detection in a Superconducting Quantum Circuit*, *Nature (London)* **519**, 66 (2015).
- [32] C. K. Andersen, A. Remm, S. Lazar, S. Krinner, N. Lacroix, G. J. Norris, M. Gabureac, C. Eichler, and A. Wallraff, *Repeated Quantum Error Detection in a Surface Code*, *Nat. Phys.* **16**, 875 (2020).
- [33] A. Erhard, H. P. Nautrup, M. Meth, L. Postler, R. Stricker, M. Ringbauer, P. Schindler, H. J. Briegel, R. Blatt, N. Friis *et al.*, *Entangling Logical Qubits with Lattice Surgery*, *Nature (London)* **589**, 220 (2021).
- [34] M. Gong, X. Yuan, S. Wang, Y. Wu, Y. Zhao, C. Zha, S. Li, Z. Zhang, Q. Zhao, Y. Liu *et al.*, *Experimental Exploration of Five-Qubit Quantum Error Correcting Code with Superconducting Qubits*, *Natl. Sci. Rev.*, [nwab011](https://doi.org/10.1093/nsr/nwab011) (2021).
- [35] E. Knill, R. Laflamme, R. Martinez, and C. Negrevergne, *Benchmarking Quantum Computers: The Five-Qubit Error Correcting Code*, *Phys. Rev. Lett.* **86**, 5811 (2001).

- [36] 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*, arXiv:2108.01646.
- [37] D. Nigg, M. Mueller, E. A. Martinez, P. Schindler, M. Hennrich, T. Monz, M. A. Martin-Delgado, and R. Blatt, *Quantum Computations on a Topologically Encoded Qubit*, *Science* **345**, 302 (2014).
- [38] J. Hilder, D. Pijn, O. Onishchenko, A. Stahl, M. Orth, B. Lekitsch, A. Rodriguez-Blanco, M. Miller, F. Schmidt-Kaler, and U. Poschinger, *Fault-Tolerant Parity Readout on a Shuttling-Based Trapped-Ion Quantum Computer*, arXiv:2107.06368.
- [39] L. Egan, D. M. Debroy, C. Noel, A. Risinger, D. Zhu, D. Biswas, M. Newman, M. Li, K. R. Brown, M. Cetina, and C. Monroe, *Fault-Tolerant Control of an Error-Corrected Qubit*, *Nature (London)* **598**, 281 (2021).
- [40] Y.-H. Luo, M.-C. Chen, M. Erhard, H.-S. Zhong, D. Wu, H.-Y. Tang, Q. Zhao, X.-L. Wang, K. Fujii, L. Li *et al.*, *Quantum Teleportation of Physical Qubits into Logical Code-Spaces*, *Proc. Natl. Acad. Sci. U.S.A.* **118**, e2026250118 (2021).
- [41] T. Aoki, G. Takahashi, T. Kajiya, J.-I. Yoshikawa, S. L. Braunstein, P. Van Loock, and A. Furusawa, *Quantum Error Correction beyond Qubits*, *Nat. Phys.* **5**, 541 (2009).
- [42] L. Hu, Y. Ma, W. Cai, X. Mu, Y. Xu, W. Wang, Y. Wu, H. Wang, Y. Song, C.-L. Zou *et al.*, *Quantum Error Correction and Universal Gate Set Operation on a Binomial Bosonic Logical Qubit*, *Nat. Phys.* **15**, 503 (2019).
- [43] R. W. Heeres, P. Reinhold, N. Ofek, L. Frunzio, L. Jiang, M. H. Devoret, and R. J. Schoelkopf, *Implementing a Universal Gate Set on a Logical Qubit Encoded in an Oscillator*, *Nat. Commun.* **8**, 94 (2017).
- [44] C. Flühmann, T. L. Nguyen, M. Marinelli, V. Negnevitsky, K. Mehta, and J. P. Home, *Encoding a Qubit in a Trapped-Ion Mechanical Oscillator*, *Nature (London)* **566**, 513 (2019).
- [45] N. Ofek, A. Petrenko, R. Heeres, P. Reinhold, Z. Leghtas, B. Vlastakis, Y. Liu, L. Frunzio, S. Girvin, L. Jiang *et al.*, *Extending the Lifetime of a Quantum Bit with Error Correction in Superconducting Circuits*, *Nature (London)* **536**, 441 (2016).
- [46] P. Campagne-Ibarcq, A. Eickbusch, S. Touzard, E. Zalys-Geller, N. E. Frattini, V. V. Sivak, P. Reinhold, S. Puri, S. Shankar, R. J. Schoelkopf *et al.*, *Quantum Error Correction of a Qubit Encoded in Grid States of an Oscillator*, *Nature (London)* **584**, 368 (2020).
- [47] X.-C. Yao, T.-X. Wang, H.-Z. Chen, W.-B. Gao, A. G. Fowler, R. Raussendorf, Z.-B. Chen, N.-L. Liu, C.-Y. Lu, Y.-J. Deng *et al.*, *Experimental Demonstration of Topological Error Correction*, *Nature (London)* **482**, 489 (2012).
- [48] C.-Y. Lu, W.-B. Gao, J. Zhang, X.-Q. Zhou, T. Yang, and J.-W. Pan, *Experimental Quantum Coding against Qubit Loss Error*, *Proc. Natl. Acad. Sci. U.S.A.* **105**, 11050 (2008).
- [49] H. Bombin and M. A. Martin-Delgado, *Topological Quantum Distillation*, *Phys. Rev. Lett.* **97**, 180501 (2006).
- [50] 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).
- [51] 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).
- [52] D. Kielpinski, C. Monroe, and D. J. Wineland, *Architecture for a Large-Scale Ion-Trap Quantum Computer*, *Nature (London)* **417**, 709 (2002).
- [53] J. M. Pino, J. M. Dreiling, C. Figgatt, J. P. Gaebler, S. A. Moses, M. S. Allman, C. H. Baldwin, M. Foss-Feig, D. Hayes, K. Mayer, C. Ryan-Anderson, and B. Neyenhuis, *Demonstration of the Trapped-Ion Quantum CCD Computer Architecture*, *Nature (London)* **592**, 209 (2021).
- [54] M. D. Barrett, J. Chiaverini, T. Schaetz, J. Britton, W. M. Itano, J. D. Jost, E. Knill, C. Langer, D. Leibfried, R. Ozeri, and D. J. Wineland, *Deterministic Quantum Teleportation of Atomic Qubits*, *Nature (London)* **429**, 737 (2004).
- [55] E. Dennis, A. Kitaev, A. Landahl, and J. Preskill, *Topological Quantum Memory*, *J. Math. Phys. (N.Y.)* **43**, 4452 (2002).
- [56] A. G. Fowler, M. Mariantoni, J. M. Martinis, and A. N. Cleland, *Surface Codes: Towards Practical Large-Scale Quantum Computation*, *Phys. Rev. A* **86**, 032324 (2012).
- [57] H. Bombin, R. S. Andrist, M. Ohzeki, H. G. Katzgraber, and M. A. Martin-Delgado, *Strong Resilience of Topological Codes to Depolarization*, *Phys. Rev. X* **2**, 021004 (2012).
- [58] M. A. Nielsen and I. L. Chuang, *Quantum Computation and Quantum Information* (Cambridge University Press, Cambridge, England, 2000).
- [59] F. Splatt, M. Harlander, M. Brownnutt, F. Zähringer, R. Blatt, and W. Hänsel, *Deterministic Reordering of  $^{40}\text{Ca}^+$  Ions in a Linear Segmented Paul Trap*, *New J. Phys.* **11**, 103008 (2009).
- [60] A. W. Cross, L. S. Bishop, J. A. Smolin, and J. M. Gambetta, *Open Quantum Assembly Language*, arXiv:1707.03429.
- [61] H. Goto, *Minimizing Resource Overheads for Fault-Tolerant Preparation of Encoded States of the Steane Code*, *Sci. Rep.* **6**, 19578 (2016).
- [62] R. Chao and B. W. Reichardt, *Quantum Error Correction with Only Two Extra Qubits*, *Phys. Rev. Lett.* **121**, 050502 (2018).
- [63] R. Chao and B. W. Reichardt, *Fault-Tolerant Quantum Computation with Few Qubits*, *Quantum Inf.* **4**, 42 (2018).
- [64] S. Aaronson and D. Gottesman, *Improved Simulation of Stabilizer Circuits*, *Phys. Rev. A* **70**, 052328 (2004).
- [65] J. Anderson, *Fault-Tolerance in Two-Dimensional Topological Systems*, Ph.D. thesis, University of New Mexico, 2012.
- [66] C. Ryan-Anderson, *Quantum Algorithms, Architecture, and Error Correction*, Ph.D. thesis, University of New Mexico, 2018.
- [67] E. Knill, *Quantum Computing with Realistically Noisy Devices*, *Nature (London)* **434**, 39 (2005).

- [68] B. Efron, *The Jackknife, the Bootstrap and Other Resampling Plans* (Society for Industrial and Applied Mathematics, Philadelphia, 1982).
- [69] S. Bravyi and A. Kitaev, *Universal Quantum Computation with Ideal Clifford Gates and Noisy Ancillas*, *Phys. Rev. A* **71**, 022316 (2005).
- [70] J. Preskill, *Reliable Quantum Computers*, *Proc. R. Soc. A* **454**, 385 (1998).
- [71] X. Zhou, D. W. Leung, and I. L. Chuang, *Methodology for Quantum Logic Gate Construction*, *Phys. Rev. A* **62**, 052316 (2000).
- [72] B. W. Reichardt, *Quantum Universality from Magic States Distillation Applied to CSS Codes*, *Quantum Inf. Process.* **4**, 251 (2005).
- [73] H. Goto, *Step-by-Step Magic State Encoding for Efficient Fault-Tolerant Quantum Computation*, *Sci. Rep.* **4**, 1 (2014).
- [74] C. Ryan-Anderson, PECOS: Performance Estimator of Codes on Surfaces, <https://github.com/PECOS-packages/PECOS>.
- [75] J. Iverson, *Coherence in Logical Channels*, *New J. Phys.* **22**, 073066 (2020).
- [76] V. Negnevitsky, M. Marinelli, K. K. Mehta, H.-Y. Lo, C. Flühmann, and J. P. Home, *Repeated Multi-Qubit Readout and Feedback with a Mixed-Species Trapped-Ion Register*, *Nature (London)* **563**, 527 (2018).
- [77] D. Hayes, D. Stack, B. Bjork, A. C. Potter, C. H. Baldwin, and R. P. Stutz, *Eliminating Leakage Errors in Hyperfine Qubits*, *Phys. Rev. Lett.* **124**, 170501 (2020).
- [78] M. Suchara, A. W. Cross, and J. M. Gambetta, *Leakage Suppression in the Toric Code*, in *Proceedings of the 2015 IEEE International Symposium on Information Theory (ISIT)* (IEEE, Piscataway, NJ, 2015), pp. 1119–1123.
- [79] N. C. Brown, A. Cross, and K. R. Brown, *Critical Faults of Leakage Errors on the Surface Code*, in *Proceedings of the 2020 IEEE International Conference on Quantum Computing and Engineering (QCE)* (IEEE, Piscataway, NJ, 2020), pp. 286–294.
- [80] R. I. Tobey, K. W. Lee, A. M. Hankin, D. N. Gresh, D. J. Francois, J. G. Bohnet, D. Hayes, and M. J. Bohn, *A High-Power, Low-Noise, Ultraviolet Laser System for Trapped-Ion Quantum Computing*, in *Proceedings of the Conference on Lasers and Electro-Optics* (Optical Society of America, Washington, DC, 2020), p. AF3K.3.
- [81] P. Wang, C.-Y. Luan, M. Qiao, M. Um, J. Zhang, Y. Wang, X. Yuan, M. Gu, J. Zhang, and K. Kim, *Single Ion Qubit with Estimated Coherence Time Exceeding One Hour*, *Nat. Commun.* **12**, 233 (2021).
- [82] E. Jordan, K. A. Gilmore, A. Shankar, A. Safavi-Naini, J. G. Bohnet, M. J. Holland, and J. J. Bollinger, *Near Ground-State Cooling of Two-Dimensional Trapped-Ion Crystals with More Than 100 Ions*, *Phys. Rev. Lett.* **122**, 053603 (2019).
- [83] M. W. van Mourik, E. A. Martinez, L. Gerster, P. Hrmov, T. Monz, P. Schindler, and R. Blatt, *Coherent Rotations of Qubits within a Surface Ion-Trap Quantum Computer*, *Phys. Rev. A* **102**, 022611 (2020).
- [84] M. Ivory, W. J. Setzer, N. Karl, H. McGuinness, C. DeRose, M. Blain, D. Stick, M. Gehl, and L. P. Parazzoli, *Integrated Optical Addressing of a Trapped Ytterbium Ion*, *Phys. Rev. X* **11**, 041033 (2021).
- [85] K. Wright, J. M. Amini, D. L. Faircloth, C. Volin, S. C. Doret, H. Hayden, C.-S. Pai, D. W. Landgren, D. Denison, T. Killian, R. E. Slusher, and A. W. Harter, *Reliable Transport through a Microfabricated X-Junction Surface-Electrode Ion Trap*, *New J. Phys.* **15**, 033004 (2013).
- [86] Y. Li and S. C. Benjamin, *Efficient Variational Quantum Simulator Incorporating Active Error Minimization*, *Phys. Rev. X* **7**, 021050 (2017).
- [87] K. Temme, S. Bravyi, and J. M. Gambetta, *Error Mitigation for Short-Depth Quantum Circuits*, *Phys. Rev. Lett.* **119**, 180509 (2017).
- [88] S. Endo, S. C. Benjamin, and Y. Li, *Practical Quantum Error Mitigation for Near-Future Applications*, *Phys. Rev. X* **8**, 031027 (2018).
- [89] A. Kandala, K. Temme, A. D. Córcoles, A. Mezzacapo, J. M. Chow, and J. M. Gambetta, *Error Mitigation Extends the Computational Reach of a Noisy Quantum Processor*, *Nature (London)* **567**, 491 (2019).
- [90] J. J. Wallman and J. Emerson, *Noise Tailoring for Scalable Quantum Computation via Randomized Compiling*, *Phys. Rev. A* **94**, 052325 (2016).
- [91] A. Hashim, R. K. Naik, A. Morvan, J.-L. Ville, B. Mitchell, J. M. Kreikebaum, M. Davis, E. Smith, C. Iancu, K. P. O’Brien, I. Hincks, J. J. Wallman, J. Emerson, and I. Siddiqi, *Randomized Compiling for Scalable Quantum Computing on a Noisy Superconducting Quantum Processor*, *Phys. Rev. X* **11**, 041039 (2021).
- [92] D. M. Debroy, M. Li, M. Newman, and K. R. Brown, *Stabilizer Slicing: Coherent Error Cancellations in Low-Density Parity-Check Stabilizer Codes*, *Phys. Rev. Lett.* **121**, 250502 (2018).
- [93] B. Zhang, S. Majumder, P. H. Leung, S. Crain, Y. Wang, C. Fang, D. M. Debroy, J. Kim, and K. R. Brown, *Hidden Inverses: Coherent Error Cancellation at the Circuit Level*, [arXiv:2104.01119](https://arxiv.org/abs/2104.01119).
- [94] P. Parrado-Rodriguez, C. Ryan-Anderson, A. Bermudez, and M. Miller, *Crosstalk Suppression for Fault-Tolerant Quantum Error Correction with Trapped Ions*, *Quantum* **5**, 487 (2021).
- [95] D. K. Tuckett, S. D. Bartlett, and S. T. Flammia, *Ultra-high Error Threshold for Surface Codes with Biased Noise*, *Phys. Rev. Lett.* **120**, 050505 (2018).
- [96] A. J. Landahl and C. Ryan-Anderson, *Quantum Computing by Color-Code Lattice Surgery*, [arXiv:1407.5103](https://arxiv.org/abs/1407.5103).
- [97] 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).
- [98] C. H. Baldwin, B. J. Bjork, J. P. Gaebler, D. Hayes, and D. Stack, *Subspace Benchmarking High-Fidelity Entangling Operations with Trapped Ions*, *Phys. Rev. Research* **2**, 013317 (2020).
- [99] E. Magesan, J. M. Gambetta, and J. Emerson, *Scalable and Robust Randomized Benchmarking of Quantum Processes*, *Phys. Rev. Lett.* **106**, 180504 (2011).
- [100] S. Olmschenk, K. C. Younge, D. L. Moehring, D. N. Matsukevich, P. Maunz, and C. Monroe, *Manipulation*

*and Detection of a Trapped Yb<sup>+</sup> Hyperfine Qubit*, *Phys. Rev. A* **76**, 052314 (2007).

- [101] D. S. Steiger, T. Häner, and M. Troyer, *ProjectQ: An Open Source Software Framework for Quantum Computing*, *Quantum* **2**, 49 (2018).
- [102] R. Ozeri, W. M. Itano, R. B. Blakestad, J. Britton, J. Chiaverini, J. D. Jost, C. Langer, D. Leibfried, R. Reichle, S. Seidelin, J. H. Wesenberg, and D. J. Wineland, *Errors in*

*Trapped-Ion Quantum Gates due to Spontaneous Photon Scattering*, *Phys. Rev. A* **75**, 042329 (2007).

*Correction:* An errant duplication of Eq. (5) has been fixed, resulting in the renumbering of the subsequent equation and its citation in the caption to Fig. 6.