Parallel syndrome extraction with shared flag qubits for CSS codes of distance three

Pei-Hao Liou\(^1\) and Ching-Yi Lai\(^{1,2}\)

\(^1\)Institute of Communications Engineering, National Yang Ming Chiao Tung University, Hsinchu 30010, Taiwan
\(^2\)Physics Division, National Center for Theoretical Sciences, Taipei 10617, Taiwan

(Dated: August 2, 2022)

To perform fault-tolerant quantum computation, one can use flagged syndrome extraction with fewer ancilla qubits. However, it suffers from long circuit depth if one stabilizer is measured at a time. Previously, Reichardt showed that it is possible to measure multiple stabilizers in parallel with at most one shared flag qubit for certain small quantum codes. In this paper, we propose a procedure for general CSS codes of distance three so that multiple Z-stabilizers (X-stabilizers) can be fault-tolerantly measured in parallel with one shared flag qubit. We simulate the memory and computation pseudo-thresholds for various code schemes. In particular, our parallel scheme based on Shor’s nine-qubit code performs better than known nine-qubit schemes in the literature when the memory error rate is high.

I. INTRODUCTION

Physical platforms for quantum computing usually have finite qubit coherent time and high gate error rates \([1–4]\). To have reliable quantum computers, it is necessary to realize fault-tolerant quantum computation (FTQC) \([5–12]\). The central idea of FTQC is to compute directly on quantum states encoded in a quantum error-correcting code. Encoded logical operations are implemented and quantum error corrections are constantly applied, so that an arbitrarily accurate quantum computation is possible when the error rates of physical gates are below a certain threshold \([5]\).

To perform a fault-tolerant syndrome extraction for error correction, one may use one of the Shor, Steane, and Knill syndrome extraction procedures \([6,11,13]\). Usually repeated syndrome measurements are performed with the Shor syndrome extraction \([5]\) to obtain a reliable error syndrome. One may also design quantum data-syndrome codes to handle syndrome measurement errors \([14,15]\). If the Steane or Knill syndrome extraction is used, no repeated measurements are required since one has to handle only an effective error and a residual error will be treated in the next error correction cycle \([16]\). However, large ancilla blocks are required for the Steane or Knill syndrome extraction. Huang and Brown recently proposed another syndrome extraction method, which is a compromise of both the Shor and Steane extractions \([17]\).

It is also possible to perform error correction for certain quantum codes with direct syndrome extractions \([18,19]\) but usually a decoding strategy is required, such as repeated syndrome measurements followed by a perfect matching.

In \([20]\), Yoder and Kim showed that fewer ancilla qubits are required for FTQC based on certain quantum codes. Following that, Chao and Reichardt introduced the idea of flag qubits for syndrome extraction so that fewer ancilla qubits are required for a syndrome extraction procedure to be fault-tolerant \([21,22]\). They also showed that multiple stabilizers can be measured with one shared flag qubit for certain small codes. The central idea is to enlarge the space of syndrome bits by introducing additional flag qubits and CNOT gates such that any single location failure will have a unique error syndrome and thus can be corrected. Since a flag FTQC scheme requires fewer qubits, it is suitable for near-term quantum devices. More general flag schemes have been studied for FTQC based on stabilizer codes and magic state distillation \([23,24]\).

A standard flagged syndrome extraction circuit measures a single stabilizer generator at a time. If a complete syndrome extraction circuit is composed of several flagged syndrome extraction circuits in a cascade form, most qubits are idle most of the time. When the error rate for idling is high, the benefit of using fewer ancilla qubits with the flagged syndrome extraction may be compensated. To handle this problem of long circuit depth, Reichardt proposed the idea of parallel flagged syndrome extraction for multiple stabilizers for certain small codes of distance three \([27]\). Moreover, he showed that two or three stabilizer of the \([[[7,1,3]]]\) code \([28]\) can be fault-tolerant measured in parallel without additional flag qubits. (Reichardt’s Parallel flagged syndrome extraction is reviewed in Appendix A).

In this paper, we propose a procedure (Algorithm 3) for parallel syndrome extraction with shared flag qubits for a general CSS code of distance three. We show that multiple Z-stabilizers can be measured with one shared flag qubit and similarly for multiple X-stabilizers. In this way, the circuit depth can be greatly reduced compared to the standard flagged syndrome extraction. In particular, we show that all the Z-stabilizers (X-stabilizers) can be measured in parallel with one flag qubit in the scenario of fault-tolerant error detection.

In addition, we show that for flagged syndrome extraction of a CSS code, a complete unflagged syndrome extraction is not necessary when a flag rises. More precisely, we observe that in a parallel syndrome extraction of multiple Z-stabilizers, the residual errors when a flag
risers will have at most a weight-one Pauli $X$ error, which can be corrected in the next error correction cycle, so only $X$-stabilizers need to be measured to catch high-weight $Z$ errors. In this way, the circuit depth can be further reduced.

We remark that one $X$-stabilizer and one $Z$-stabilizer (two dual operators) can be measured in parallel with a shared flag qubit using Reichardt’s method for the $[[15,7,3]]$ code. On the contrary, our procedure applies to only stabilizers of the same type. Thus our procedure applies to asymmetric CSS codes with unequal numbers of $X$- and $Z$- stabilizers, such as the $[[15,1,3]]$ Reed-Muller code.

Finally, we simulate and compare several flag and parallel schemes based on the $[[4,2,2]]$, $[[7,1,3]]$, and Shor’s $[[9,1,3]]$ codes for error detection and error correction, respectively. Both the memory and computation pseudo-thresholds are simulated for each scheme. Note that we do not assume a two-dimensional layout for a FTQC scheme with physical restrictions and non-local CNOTs are allowed. In addition to the standard decoder for flagged syndrome extraction, we also consider a two-step decoder: if the error syndrome is nonzero, then measure it again and use the later results. Our simulations show that this two-step decoder improves the standard one (Table IV).

Assume all the gate error rates and memory error rate are the same. We can observe that a parallel scheme will perform better than its nonparallel counterpart. In addition, our $[[9,1,3]]$ parallel scheme has better memory and computation pseudo-thresholds than Reichardt’s $[[7,1,3]]$ parallel scheme (Table IV): we obtain the memory pseudo-threshold of $1.75 \times 10^{-4}$ and $8.08 \times 10^{-4}$ for the $[[7,1,3]]$ and $[[9,1,3]]$ parallel schemes, respectively.

We compare our $[[9,1,3]]$ parallel scheme with the literature. In [19], Li et al. proposed a fault-tolerant quantum memory scheme for the $[[9,1,3]]$ Bacon-Shor-13 and they showed this scheme outperforms a scheme based on the $[[9,1,3]]$ surface codes using eight ancilla qubits. From the circuit design, one can observe that Bacon-Shor-13 performs better when the memory rate is low since it has fewer qubits involved (a total of 13 qubits). When there are no memory errors, we obtain a high memory pseudo-threshold of $8.70 \times 10^{-3}$ for Bacon-Shor-13, which is slightly better than our $[[9,1,3]]$ parallel scheme with a memory pseudo-threshold of $7.74 \times 10^{-3}$. On the other hand, when the memory error rate is high, our $[[9,1,3]]$ parallel scheme has a better threshold of $2.0 \times 10^{-3}$, compared to $1.19 \times 10^{-3}$ for Bacon-Shor-13 (Table V).

This paper is organized as follows. In Section II, we briefly introduce the basics of stabilizer codes, FTQC, and flagged syndrome extraction. In Section III, we show how to perform parallel syndrome extraction for multiple stabilizers of a CSS code with one shared flag qubit. Simulations our parallel schemes are provided and compared in Section IV. Then we conclude in Section V.

II. PRELIMINARIES

A. Stabilizer Codes

We consider quantum errors that are tensor product of Pauli matrices $I = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$, $X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$, $Y = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix}$, and $Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$ in the computational basis $\{|0\}, \{|1\}\}$. For convenience, we omit the symbol of tensor product for $n$-fold Pauli operators. For example, $X \otimes Y \otimes Z \otimes I \otimes I$ will be denoted as $X_1Y_2Z_3$. An $X$- or $Z$-type Pauli operator has nonidentity components that are all $X$ or all $Z$ matrices. For an $X$- or $Z$-type Pauli operator, the redundant $X$s or $Z$s will be suppressed. For example, $X_1X_2X_3$ may also be denoted as $X_{1,2,3}$. The weight of an $n$-fold Pauli operator is the number of its nonidentity elements. Usually a high-weight error is less likely than a low-weight error in an error model. Consequently, we would like to handle the more-likely events in error correction.

An $[[n,k,d]]$ stabilizer code $C(S)$ is defined as the joint $+1$ eigenspace of an abelian subgroup $S$ of the $n$-fold Pauli operators with $n-k$ independent generators $g_1, \ldots , g_{n-k}$ and the operators in $S$ are called stabilizers. It encodes $k$ logical qubits into $n$ physical qubits and can detect errors up to weight $d-1$ or correct errors up to weight $\lfloor \frac{d-1}{2}\rfloor$ [9].

Definition 1 The error syndrome of a Pauli error $E$ with respect to the $n-k$ stabilizer generators $g_i \in S$ is a binary string of length $n-k$, whose $i$-th bit is 0 if $E$ commutes with $g_i$, and 1, otherwise.

If an error has a nonzero error syndrome, it can be detected and possibly corrected.

Proposition 2 If each operator of a set of Pauli errors has a unique error syndrome with respect to a stabilizer group, then this set of Pauli errors is correctable.

Clearly one can build a lookup table that consists of each error operator and its error syndrome for error correction [33].

A Calderbank-Shor-Steane (CSS) code is a stabilizer code defined by $X$-type or $Z$-type stabilizer generators [34][35]. $X$ and $Z$ errors can be separately treated in a CSS code. A CSS code of distance three can correct a single $X$ error and a single $Z$ error, simultaneously.

B. Fault-tolerant quantum computation

In a quantum circuit, quantum errors occur due to imperfect quantum gates or memory and they may propagate through multiple-qubit gates.
Definition 3 A location failure is referred to as an event that a Pauli error occurs after a perfect single-qubit gate, two-qubit gate, ancilla preparation, qubit measurement, or an idle qubit in a circuit.

To perform quantum computation with imperfect gates and memory, quantum information has to be encoded in a quantum stabilizer code and encoded quantum operations are performed so that quantum error correction is constantly performed.

Definition 4 A location in a procedure is called a bad location if one of its location failures may evolve into an uncorrectable (undetectable) error for a code of distance three (two). Such location failures are called bad location failures.

Definition 5 A procedure is called fault-tolerant if there are no bad locations in the procedure.

For example, a fault-tolerant circuit of a controlled-NOT (CNOT) gate on two qubits is encoded to the circuit in Fig. 1 where each qubit is encoded into a CSS codeword and bitwise CNOT gates are applied. Note that an error detection (ED) or error correction (EC) block is performed to each codeword prior to and subsequent to the bitwise CNOT gates. Such implementation of a logical CNOT gate is called an extended rectangle (ex-Rec) of the CNOT gate [36].

Definition 6 The error threshold for a procedure is the (physical) error rate below which the logical error rate of the procedure would be lower than the error rate.

We consider two types of thresholds: memory threshold and computation threshold. The memory threshold is the error threshold for error correction or detection on a quantum codeword. The computation threshold is the error threshold for an ex-Rec CNOT gate since the ex-Rec CNOT is usually the most complicated procedure in a universal set of gates.

Simulations in this paper assume the following error model. We add independent depolarizing errors with rate \( p \) as quantum operations after gates or before measurements in the circuit. Assume that each gate takes the same unit of time. Each single-qubit location undergoes \( X \), \( Y \), or \( Z \) with probability \( p/3 \) and each qubit-measurement is corrupted with probability \( 2p/3 \). Each CNOT gate is followed by one of the 15 non-identity two-qubit Pauli operators with probability \( p/15 \). An idle qubit suffers depolarizing errors with rate \( \gamma p \) for \( 0 \leq \gamma \leq 1 \).

C. Fault-tolerant syndrome extraction with flag qubits

In general a syndrome extraction circuit has many bad locations. By introducing additional ancillary qubits, called flag qubits, one can carefully place controlled-NOT (CNOT) gates so that bad location failures will trigger these flag qubits and thus be detected [21, 22].

Figure 2 (a) illustrates how a weight-4 stabilizer \( Z_1 Z_2 Z_3 Z_4 \) is measured. Four CNOT gates are used to check the parity of four data qubits and the result is left in the first ancilla qubit (marked in red), which is initialized in \(|0\rangle\). Observe that location a in Fig. 2 (a) is a bad location since a \( Z \) error here will induce a weight-2 \( Z \) error on the data qubits. This event cannot be detected by the stabilizer measurements but it will be detected using a flag qubit with elegantly positioned CNOT gates (marked in blue) as shown in Fig. 2 (a). We say that this failure triggers the flag qubit and the flag rises. This method is called flagged syndrome extraction [21, 22]. The syndrome extraction circuit without the flag qubit and the two CNOT gates will be called an unflagged syndrome extraction circuit.

![FIG. 2. (a) Flagged fault-tolerant syndrome extraction for \( Z_1 Z_2 Z_3 Z_4 \). The ancilla qubit initialized in \(|+\rangle\) is a flag qubit, which is coupled to the raw syndrome measurement circuit via two additional CNOTs. (b) Flagged fault-tolerant syndrome extraction for \( X_1 X_2 X_3 X_4 \).](image)

The measurement of a weight-4 stabilizer \( X_1 X_2 X_3 X_4 \) is similarly implemented in Fig. 2 (b). These circuits can be directly generalized for measuring \( Z \)- or \( X \)-stabilizers of higher weight as shown in Fig. 3 This is called a standard flagged syndrome extraction circuit. A circuit for measuring the error syndrome with respect to a set of stabilizer generators will be called a complete syndrome extraction circuit.

In the following, we discuss how flagged syndrome extraction is used in fault-tolerant quantum error detection...
and correction.

1. Quantum error detection with flag qubits

A fault-tolerant error-detection scheme has to be designed such that a detectable error will not evolve into an undetectable error. A flagged syndrome extraction circuit ensures that any bad location failure will trigger a flag qubit and be detected. Therefore, one learns that some error occurs if anyone of the measurement outcomes in the flagged syndrome extraction circuits is $-1$, and the codeword has to be discarded.

Consider the $[[4, 2, 2]]$ error-detecting code \cite{9}, which is defined by two stabilizer generators $X_1X_2X_3X_4$ and $Z_1Z_2Z_3Z_4$. Its complete syndrome extraction consists a sequential implementation of Fig. 2(a) and Fig. 2(b) as shown in Fig. 3. This procedure is fault-tolerant.

2. Quantum error correction with flag qubits

In a fault-tolerant error correction scheme with flagged syndrome extraction, error recovery is adaptively performed according to the measured error syndrome and flag qubits. The basic error correction rules for distance-three quantum codes are as follows.

A complete flagged syndrome extraction circuit is first implemented. If no flag qubits rise, it is most likely that no bad location failure occurs and a usual syndrome decoding is performed according to the measured error syndrome. If one flag qubit rises during the complete syndrome extraction, a round of complete unflagged syndrome extraction will be performed right away and error recovery is performed according to the flag qubit as well as its measured error syndrome.

When a flag qubit rises, we know that there is some high-weight error in the codeword. In this case, the conventional minimum-weight decoding rule does not work and we are not aware of any general decoding algorithm. We have to choose the most likely error according to the flag qubits and the error syndrome from a complete unflagged syndrome extraction.

The standard decoding procedure in \cite{21} is summarized in Algorithm 1.

\begin{algorithm}
\caption{Standard decoding procedure for flagged syndrome extraction.}
\begin{algorithmic}
\STATE Suppose that there are $r$ stabilizer generators $g_1, g_2, \ldots, g_r$;
\FOR{$i = 1, 2, \ldots, r$}
\STATE Apply $C(g_i)$ and get syndrome bit $m_i$;
\STATE Let $f_i$ be the measurement outcome on the flag qubit of $C(g_i)$;
\IF{$f_i = 1$}
\STATE Apply a complete unflagged syndrom
\STATE $m_{12} \ldots r$;
\STATE Use $m_{12} \ldots r$ and the syndrome table for $f_i$ to do error correction;
\STATE break;
\ENDIF
\ENDFOR
\ENDFOR
\STATE Use the syndrome bits $m_{12} \ldots r$ to do error correction;
\end{algorithmic}
\end{algorithm}

### III. FLAG-SHARED SYNDROME EXTRACTION FOR CODES OF DISTANCE TWO OR THREE

In this section, we propose a procedure for parallel syndrome extraction with shared flag qubits for a general CSS code of distance three. We begin with the notation.

Suppose that there are $r$ stabilizer generators $g_1, \ldots, g_r$ to be measured for an $[[n, k, 3]]$ CSS code. Let $C(g_i)$ be a (standard) flagged syndrome extraction circuit for $g_i$ as shown in Fig. 3 for a $Z$-stabilizer. The flagged syndrome extraction circuit is designed so that each bad location failure will trigger the flag qubit. Let $B(C(g_i))$ denote the number of location failures in $C(g_i)$ that will trigger the flag qubit.

An observation is that $B(C(g_i))$ depends on the weight of the stabilizer $g_i$.

**Proposition 7** Suppose that $g_i$ is an $X$- or $Z$-stabilizer of weight $w$. Then $B(C(g_i)) \leq w - 1$.

**Proof.** Consider a $Z$-stabilizer $g_i$ of weight $w$ with a standard flagged syndrome extraction circuit $C(g_i)$ as shown in Fig. 3. There are two CNOTs connecting the flag and the ancillary qubits. Another $w - 2$ CNOTs are placed in sequence between these two CNOT gates. All the location failures at $a, b, h$, and $i$ will not trigger the flag qubit, while a Pauli $Z$ (or $Y$) at the other $w - 1$ locations $c, d$, up to $g$ will.

Note that a $Z$ error at $g$ is not a bad failure since it remains to be a weight-one error at the end of the
circuit. The other $w - 2$ location failures may evolve to high-weight $Z$ errors, which are potential bad failures.

Another observation is that flagged syndrome extraction circuits for stabilizers of the same type may be joined together and a shared flag qubit may be sufficient for fault-tolerant syndrome extraction. For example, Fig. 4 (B) shows the joint flagged syndrome extraction circuit with one shared flag qubit for two stabilizers.

Moreover, we find that no additional bad location failures are introduced in this joint circuit. Consider the two CNOTs connecting an ancilla qubit and the flag qubit. A Z error on the flag qubit prior to the first CNOT will be absorbed to the flag qubit, an X error on the flag qubit will not propagate to any other ancillas and it triggers the flag qubit. A Z error on the flag qubit following the second CNOT has no effect on the flag qubit. An X error on the flag qubit will not propagate to any other ancillas and it triggers the flag qubit. No errors on one ancilla could propagate through the flag qubit to another ancilla because of the directions of CNOTs. Hence we have the following proposition.

**Proposition 8** Suppose that $g_1, \ldots, g_s$ are stabilizers of the same type (X or Z). Suppose that a flag qubit is shared by $C(g_1), \ldots, C(g_s)$ and the CNOTs connecting the ancillas and the flag qubit are placed in sequence. Then a bad location failure in this joint circuit must have appeared in one of the $C(g_i)$; no additional bad location failure is introduced in the joint circuit.

The next question is: how many stabilizer generators can be fault-tolerantly measured in parallel with only one shared flag qubit? Suppose that there are $x$ X-type stabilizers and $z$ Z-type stabilizers. If $g_i$ is a Z-stabilizer, we observe that in $C(g_i)$, a bad location failure always evolves to a high-weight Z error. For example, in Fig. 2(a) a Z error at location $a$ evolves to $Z_3Z_4$ at the end of the circuit. When a flag rises, this high-weight Z error will be identified by measuring the X-stabilizers in an unflagged syndrome extraction circuit. Consequently, we must have $B(C(g_i)) \leq 2^x$ so that all the location failures that trigger the flag qubit can be identified using an unflagged syndrome extraction. Similar for the flagged syndrome extraction of an X-stabilizer.

Consequently, we must have the following proposition for a flag-shared syndrome extraction scheme of multiple stabilizers of the same type, following Proposition 8.

**Proposition 9** Suppose that $g_1, g_2, \ldots, g_s$ are stabilizers of the same type and $C(g_1), C(g_2), \ldots, C(g_s)$ can be performed in parallel using a shared flag qubit. Then the following conditions hold.

1. Each distinct bad location failure has a unique error syndrome when a flag rises.

2. \[
\begin{align*}
\sum_{i=1}^{s} B(C(g_i)) & \leq 2^x, & \text{if} \ g_1, \ldots, g_s \text{ are X-type;}
\sum_{i=1}^{s} B(C(g_i)) & \leq 2^z, & \text{if} \ g_1, \ldots, g_s \text{ are Z-type,}
\end{align*}
\] assuming that there are $x$ X-type stabilizers and $z$ Z-type stabilizers.

According to Propositions 8 and 9 we have a procedure (Algorithm 2) for finding a parallel flagged syndrome extraction circuit for a set of stabilizers with a shared flag qubit.

**Algorithm 2:** Finding a parallel flagged syndrome extraction circuit with one shared flag qubit.

1. Choose stabilizers $g_1, g_2, \ldots, g_s$ of Z (X) type such that $\sum_{i=1}^{s} B(C(g_i)) \leq 2^x (2^z)$, where $x$ ($z$) is the number of independent X (Z) stabilizer generators;
2. Let $C(g_1), C(g_2), \ldots, C(g_s)$ share one flag qubit;
3. Let unique = false;
4. while unique do
5. randomly adjust the orders of a subset of CNOTs in $C(g_1), C(g_2), \ldots, C(g_s)$;
6. if each of the locations that trigger the flag qubit has a unique error syndrome from an unflagged syndrome extraction;
7. then
8. let unique = true;
9. end
10 end

We remark that the number of possible error syndromes is exponential in the number of stabilizer generators, while the number of locations that trigger the flag qubit is linear in the number of stabilizer generators in parallel. As a consequence, this randomized algorithm is good enough when $x$ or $z$ is sufficiently large. We provide examples of Shor’s [[9,1,3]] code and the [[15,1,3]] (punctured) Reed-Muller code in the following subsections.

We can extend the above idea for error detection. Note that the conditions for error detection is that any bad location failure can be detected. Since a bad location failure will trigger the flag qubit in a flagged syndrome extraction circuit, we have the following result.

**Lemma 10** All the Z-stabilizers of a CSS code of distance two or three can be fault-tolerantly measured in parallel with a shared flag qubit for error detection. Similarly for the X-stabilizers.

### A. Shor’s [[9,1,3]] code

Shor’s [[9,1,3]] code is constructed by concatenating a three-qubit phase-flip code with a three-qubit bit-flip QECC and can correct an arbitrary X error and an arbitrary Z error simultaneously [29]. It is defined by stabilizers

\[
\begin{align*}
g_1 &= Z_1Z_2, \ g_2 = Z_2Z_3, \ g_3 = Z_4Z_5, \\
g_4 &= Z_5Z_6, \ g_5 = Z_7Z_8, \ g_6 = Z_9Z_9, \\
g_7 &= X_1X_2X_3X_4X_5X_6, \\
g_8 &= X_4X_5X_6X_7X_8X_9.
\end{align*}
\]

(1)

Since each of $g_1, g_2, g_3, g_4, g_5$ and $g_6$ has weight two, its syndrome extraction circuit consists of two CNOTs. One
can check that it has no bad location and no flag qubit is required. They can be measured in parallel as shown in Part (A) of Fig. 4.

The two stabilizers \( g_7 \) and \( g_8 \) can be fault-tolerantly measured using a parallel flagged syndrome extraction circuit by Algorithm 3. There are five location failures in \( C(g_7) \) (or \( C(g_8) \)) that trigger the flag qubits, which satisfies Proposition 7 with equality. Since there are six \( Z \)-stabilizers, Proposition 7 holds with \( 10 < 2^6 = 64 \). Consequently, a parallel syndrome extraction circuit can be easily found as shown in Part (B) of Fig. 4.

Following Lemma 11 and Algorithm 3, the unflagged syndrome extraction circuit \( U \) consists of solely Part (A) of Fig. 4. The location failures are listed in Table I, together with their error syndromes from \( U \).

![FIG. 4. Parallel flagged syndrome extraction circuit for Shor’s [[9,1,3]] code with one shared flag qubit. Part (A) extracts the syndrome bits for \( Z_1 Z_2 \) (red), \( Z_2 Z_3 \) (blue), \( Z_4 Z_5 \) (green), \( Z_6 Z_7 \) (yellow), \( Z_7 Z_8 \) (purple) and \( Z_8 Z_9 \) (gray). Part (B) extracts the syndrome bits for \( X_1 X_2 X_3 X_4 X_5 X_6 \) (blue) and \( X_4 X_5 X_6 X_7 X_8 X_9 \) (red).](image)

### B. The [[15,1,3]] Reed-Muller code

The [[15,1,3]] Reed-Muller code is defined by stabilizers

\[
g_1 = Z_1, Z_3, 5, 7, 9, 11, 13, 15, \quad g_6 = Z_5, 7, 13, 15, \quad g_{10} = Z_9, 11, 13, 15
\]

\[
g_2 = Z_2, 3, 6, 7, 10, 11, 14, 15, \quad g_5 = Z_3, 7, 11, 15, \quad g_8 = Z_{10}, 11, 14, 15
\]

\[
g_3 = Z_{4,5}, 6, 7, 12, 13, 14, 15, \quad g_7 = Z_6, 7, 14, 15
\]

\[
g_4 = Z_8, 9, 10, 11, 12, 13, 14, 15, \quad g_9 = Z_{12,13,14,15}
\]

\[
g_{11} = X_1, 3, 5, 7, 9, 11, 13, 15, \quad g_{12} = X_2, 3, 6, 7, 10, 11, 14, 15
\]

\[
g_{13} = X_4, 5, 6, 7, 12, 13, 14, 15, \quad g_{14} = X_8, 9, 10, 11, 12, 13, 14, 15
\]

There are ten \( Z \)-stabilizers \( g_1, \ldots, g_{10} \) and four \( X \)-stabilizers \( g_{11}, \ldots, g_{14} \) of weight four or eight. By Propositions 7 and 9, we can have at most three \( Z \)-stabilizers measured in parallel with one shared flag qubit, while the four \( X \)-stabilizers can be easily measured in parallel with one shared flag qubit. The resulting parallel syndrome extraction circuit is shown in Fig. 5, where Part (A) measures \( g_1, g_6 \) and \( g_{10} \), Part (B) measures \( g_2, g_5 \) and \( g_8 \), Part (C) measures \( g_3 \) and \( g_7 \), Part (D) measures \( g_4 \) and \( g_9 \), and Part (E) measures \( g_{11}, g_{12}, g_{13} \) and \( g_{14} \). Each stabilizer measurement in each part is differently colored. \( U_X \) and \( U_Z \) are the unflagged complete syndrome extraction circuits for the \( X \)-stabilizers and \( Z \)-stabilizers, respectively, following Algorithm 3. One can verify that each bad location failure has a unique error syndrome when a flag rises and the syndrome table is omitted.

### C. Decoding procedures

Finally, we propose a decoding procedure for flagged syndrome extraction of a CSS code. We begin with the following lemma.

**Lemma 11** A location failure in a flagged syndrome extraction circuit of an \( X \)-stabilizer (\( Z \)-stabilizer) will leave
solely $X$ ($Z$) errors and possibly a $Z$ ($X$) error on the data qubits.

**proof.** Consider the flagged syndrome extraction circuit in Fig. 3. Consider a location failure at one of the CNOTs that triggers the flag qubit. It may generate a Pauli error on the control (data) qubit and a $Z$ error on the target (ancillary) qubit, and this $Z$ error will propagate to multiple data qubits. The condition is similar for the syndrome extraction of a $Z$-stabilizer.

Note that in Fig. 3 a bad location failure will trigger the flag qubit, and the high-weight $Z$ error can be detected by measuring the $X$-stabilizers. In the meanwhile, a possible residual $X$ error can be detected and corrected by measuring the $Z$-stabilizers.

We remark that this residual $X$ error can be left uncorrected to the next error correction cycle, while the syndrome extraction procedure remains to be fault-tolerant. This suggests a decoding procedure for CSS codes as shown in Algorithm 3. Since we do not perform a complete syndrome extraction, the overhead for FTQC can be greatly reduced.

We further consider a two-step decoder for FTQC as shown in Algorithm 4, which is particularly helpful for codes of distance three.

We will demonstrate the advantages of our procedures in the error analysis of several codes in Section IV.
Algorithm 3: Decoding procedure for flagged syndrome extraction of a CSS code.

1. Suppose a CSS code of distance three has
   \( X \)-stabilizers \( g_1^X, \ldots, g_i^X \) and \( Z \)-stabilizers \( g_1^Z, \ldots, g_i^Z \);
2. Apply \( C(g_i^X), \ldots, C(g_i^Z) \) to get the \( Z \) error syndrome;
   if anyone of the flag qubits rises then
3. Apply a separated unflagged syndrome extraction circuit for the \( Z \)-stabilizers;
4. Perform error correction accordingly;
5. end
6. Apply \( C(g_i^X), \ldots, C(g_i^Z) \) to get the \( X \) error syndrome;
   if anyone of the flag qubits rises then
7. Apply a separated unflagged syndrome extraction circuit for the \( X \)-stabilizers;
8. Perform error correction accordingly;
9. end
10. if none of the flag qubits rises then
11. Perform error correction accordingly;
12. end

Algorithm 4: Two-step decoder for flagged syndrome extraction.

1. Suppose that there are \( r \) stabilizer generators
   \( g_1, g_2, \ldots, g_r \);
2. Apply \( C(g_1), \ldots, C(g_r) \) to get error syndrome \( m_1, \ldots, r \);
3. if \( m_i = 1 \) for some \( i \) then
4. Discard all the syndrome bits \( m_1, \ldots, r \);
5. Apply \( C(g_1), \ldots, C(g_r) \) to get error syndrome \( m_1, \ldots, r \);
6. end
7. if anyone of the flag rises then
8. Use the error syndrome \( m'_{12} \ldots r \) from the unflagged syndrome extraction circuit to do error correction accordingly;
9. end
10. if none of the flag qubits rises then
11. Use the syndrome \( m_{12} \ldots r \) to do error correction;
12. end

IV. SIMULATION RESULTS

In this section, we simulate the memory and computation thresholds of the error detection scheme based on the \([4,2,2]\) code with standard and parallel flagged syndrome extraction, respectively, the scheme based on the \([7,1,3]\) Steane code with parallel syndrome extraction, and the scheme based on Shor’s \([9,1,3]\) code with parallel syndrome extraction. In particular, we will also use the two-step decoder in Algorithm 4 for Shor’s \([9,1,3]\) code.

We consider depolarizing errors and assume all the gates have the same error rate and let \( \gamma \) be the ratio of memory error rate (idle qubits) to the gate error rate. Simulations will be conducted at \( \gamma = 0 \) and \( \gamma = 1 \).

The number of locations in an ex-Rec CNOT gate of each scheme is provided in Table II. Note that the \([4,2,2]\), \([7,1,3]\), and \([9,1,3]\) flag schemes are with standard flagged syndrome extraction; the \([4,2,2]\), \([7,1,3]\), parallel schemes are given in [27]; the \([9,1,3]\) parallel scheme is proposed in Section II A with the two-step decoder in Algorithm 4.

| Code           | \([4,2,2]\) | \([4,2,2]\) | \([7,1,3]\) | \([7,1,3]\) | \([9,1,3]\) | \([9,1,3]\) |
|----------------|-------------|-------------|-------------|-------------|-------------|-------------|
| Flag           | 16          | 8           | 192         | 72          | 88          | 60          |
| Parallel       | 8           | 4           | 96          | 36          | 8           | 8           |
| Flag + Parallel| 8           | 4           | 96          | 36          | 80          | 52          |
| CNOT           | 52          | 36          | 727         | 343         | 217         | 109         |
| Idle qubit     | 192         | 64          | 5208        | 824         | 720         | 388         |
| SWAP           | 0           | 8           | 0           | 0           | 0           | 0           |
| Total          | 276         | 124         | 6319        | 1311        | 1113        | 677         |

\( a \) These numbers are overestimated since the locations in all the unflagged syndrome extraction circuits are counted.
\( b \) \( P_s \) denotes ancilla state preparation.

TABLE II. The number of locations in an ex-Rec CNOT gate for various schemes.

A. Comparisons of procedures for unflagged syndrome extraction

We proposed Algorithm 3 for CSS codes so that a complete unflagged syndrome extraction is not required for CSS codes when some flag qubits rise. In this subsection, we demonstrate this on the parallel flagged syndrome extraction for the error correction scheme based on the \([9,1,3]\) code proposed in Section III A.

The memory and computation pseudo-thresholds using the standard and the two-step decoders are summarized in Table III. The results show that the unflagged syndrome extraction by Algorithm 3 are better than the complete unflagged syndrome extraction in all respects. Consequently, we will adopt this strategy in the following simulations.

B. Error detection

Simulations of the \([4,2,2]\), \([7,1,3]\), and \([9,1,3]\) codes with error detection are given in Fig. 6. The \([4,2,2]\), \([7,1,3]\), and \([9,1,3]\) flag schemes are with standard flagged syndrome extraction; the \([4,2,2]\), \([7,1,3]\) parallel schemes are given in [27] and can be found in Appendix A; the \([9,1,3]\) Parallel is the scheme proposed in Section III A.

Figures 6 (a) and (b) provide the memory pseudo-thresholds at \( \gamma = 0 \) and \( \gamma = 1 \), respectively, while Figures 6 (c) and (d) provide the computation pseudo-thresholds at \( \gamma = 0 \) and \( \gamma = 1 \), respectively.

In general, the flag and parallel schemes have comparable pseudo-thresholds at \( \gamma = 0 \). At \( \gamma = 1 \), the advantages of parallel scheme is more obvious as expected since it has fewer qubits and gates involved.
Memory pseudo-thresholds at $\gamma = 0$.

Memory pseudo-thresholds at $\gamma = 1$.

Computation pseudo-thresholds at $\gamma = 0$.

Computation pseudo-thresholds at $\gamma = 1$.

FIG. 6. Error detection: simulations of the memory and computation pseudo-thresholds for various schemes at $\gamma = 0$ and $\gamma = 1$.

| Standard decoder | $\gamma = 0$ | $\gamma = 1$ |
|------------------|--------------|--------------|
| Scheme           | Memory pseudo-threshold | |
| $[9, 1, 3]$ Parallel (comp.) | $4.56 \times 10^{-2}$ | $7.13 \times 10^{-4}$ |
| $[9, 1, 3]$ Parallel (Alg. 3) | $5.13 \times 10^{-3}$ | $8.08 \times 10^{-4}$ |
| Scheme           | Computation pseudo-threshold | |
| $[9, 1, 3]$ Parallel (comp.) | $6.61 \times 10^{-4}$ | $1.6 \times 10^{-4}$ |
| $[9, 1, 3]$ Parallel (Alg. 3) | $8.56 \times 10^{-3}$ | $1.98 \times 10^{-4}$ |
| Scheme           | Memory pseudo-threshold | |
| $[9, 1, 3]$ Parallel (comp.) | $7.49 \times 10^{-3}$ | $1.95 \times 10^{-4}$ |
| $[9, 1, 3]$ Parallel (Alg. 3) | $7.74 \times 10^{-3}$ | $2.01 \times 10^{-3}$ |
| Scheme           | Computation pseudo-threshold | |
| $[9, 1, 3]$ Parallel (comp.) | $1.48 \times 10^{-3}$ | $2.89 \times 10^{-4}$ |
| $[9, 1, 3]$ Parallel (Alg. 3) | $1.69 \times 10^{-3}$ | $3.02 \times 10^{-4}$ |

TABLE III. Comparisons of the two procedures for unflagged syndrome extraction. (Alg. 3) denotes the X and Z separated unflagged syndrome extraction by Algorithm 3, while (comp.) denotes a complete unflagged syndrome extraction.

At $\gamma = 0$, both the $[7,1,3]$ and $[9,1,3]$ codes have computation pseudo-thresholds around 5%, while their memory pseudo-thresholds are above 10%. At $\gamma = 1$, the $[7,1,3]$ parallel is slightly better than the $[9,1,3]$ parallel scheme for error detection.

C. Error correction

Simulations of the $[7,1,3]$ and $[9,1,3]$ codes with error correction are given in Fig. 7

We first consider the memory pseudo-threshold of the $[7,1,3]$ and $[9,1,3]$ schemes using the standard decoder for error correction, which are shown in Fig. 7 (a) and (b), respectively. At $\gamma = 0$, we obtain memory pseudo-thresholds of $1.32 \times 10^{-3}$ and $7.97 \times 10^{-3}$ for the $[7,1,3]$ flag and $[9,1,3]$ flag schemes, respectively, which are slightly better than their parallel counterparts since the effects of idle qubits are minimized in this case. At $\gamma = 1$, it is obvious from the simulations that the parallel schemes are much better than their flag counterparts, and we obtain the memory pseudo-threshold of $1.75 \times 10^{-4}$ and $8.08 \times 10^{-4}$ for the $[7,1,3]$ and $[9,1,3]$ parallel schemes, respectively. This can be observed from Table II that the parallel schemes have fewer qubits and gates.

The simulation results for the computation pseudo-thresholds using the standard decoder have similar behaviors as shown in Fig. 7 (b). However, our $[9,1,3]$ parallel scheme outperforms the $[7,1,3]$ parallel scheme in the computation pseudo-threshold.
Computation pseudo-thresholds with the standard decoder.

Computation pseudo-thresholds with the two-step decoder.

Logical error rate (Logarithmic scale)

Depolarizing error rate (Logarithmic scale)

The memory and computation pseudo-thresholds for these schemes using the standard decoder are summarized in Table IV. As a comparison, we include the pseudo-threshold of the $[[7,1,3]]$ flag scheme using a two-round decoder in [24], which has a better threshold than the $[[7,1,3]]$ flag scheme using the standard decoder.

| Scheme          | $\gamma = 0$          | $\gamma = 1$          |
|-----------------|------------------------|------------------------|
| $[[7,1,3]]$ Flag| $1.32 \times 10^{-3}$  | $2.97 \times 10^{-3}$  |
| $[[7,1,3]]$ Flag| $-1.29 \times 10^{-3}$ | $3.39 \times 10^{-3}$  |
| $[[7,1,3]]$ Parallel | $7.97 \times 10^{-3}$  | $6.03 \times 10^{-4}$  |
| $[[9,1,3]]$ Parallel | $5.13 \times 10^{-3}$  | $8.08 \times 10^{-4}$  |

| Scheme          | Computation pseudo-threshold |
|-----------------|------------------------------|
| $[[7,1,3]]$ Flag| $2.03 \times 10^{-4}$        | $3.95 \times 10^{-6}$ |
| $[[7,1,3]]$ Parallel | $1.73 \times 10^{-4}$        | $3.05 \times 10^{-5}$ |
| $[[9,1,3]]$ Flag| $9.83 \times 10^{-4}$        | $1.42 \times 10^{-4}$ |
| $[[9,1,3]]$ Parallel | $8.56 \times 10^{-4}$        | $1.98 \times 10^{-4}$ |

* A two-round decoder is used.

TABLE IV. Memory and computation pseudo-thresholds using the standard decoder.

Next we consider the simulations with the two-step decoder. The simulation results for the memory and computation pseudo-thresholds using the two-step decoder are shown in Figs. 7 (c) and (d). The parallel scheme outperforms the flag scheme in all respects. The memory and computation pseudo-thresholds are summarized in Table V.

Finally, we compare our $[[9,1,3]]$ parallel scheme with Bacon-Shor-13 in [19]. For a fair comparison, we reproduce their results and simulate the memory pseudo-threshold for the Bacon-Shor-13 scheme at $\gamma = 0$ and $\gamma = 1$ based on the circuit for Bacon-Shor-13 in Fig. 8 and the results are also shown in Table V and Fig. 7(c).

Since Bacon-Shor-13 uses only four ancilla qubits for syndrome extraction, it has long circuit depth and many idle qubit locations. Consequently, it is more useful when the memory error rate is low. At $\gamma = 0$, Bacon-Shor-13 has the best memory pseudo-threshold because its CNOT gates are elaborately arranged to avoid the generation of weight-2 errors.

On the other hand, when the memory error rate is high, our $[[9,1,3]]$ parallel scheme outperforms Bacon-Shor-13. At $\gamma = 1$, the memory pseudo-threshold for Bacon-Shor-13 is $1.19 \times 10^{-3}$, while that for the $[[9,1,3]]$ parallel scheme is $2.01 \times 10^{-3}$.
In particular, they consider a variant of the standard flagged syndrome extraction to reduce the circuit depth. Consequently, the CNOT gates for a stabilizer measurement may be equally connect to an ancilla qubit and the flag qubit. The circuit depth can be decreased in this way. However, it may not be easy to have parallel syndrome extraction for multiple stabilizers with one shared flag qubit since it will be difficult to find an arrangement of the CNOT gates such that each bad location failure will have a unique error syndrome.

We consider only quantum codes of distance two and three in this paper. It is possible to extend our framework for CSS codes of higher distances, where more shared flag qubits are required. However, it is difficult to arrange the CNOTs so that each bad location failure will have a unique error syndrome by a complete unflagged syndrome extraction.

V. CONCLUSION

We proposed a general parallel flagged syndrome extraction scheme for a CSS code of distance three with one shared flag qubit (Algorithm 3), together with a procedure for unflagged syndrome extraction (Algorithm 3). Examples of the [[9,1,3]] Shor and [[15,1,3]] Reed-Muller codes were provided. Simulations on the [[9,1,3]] code demonstrated that our [[9,1,3]] parallel scheme has several benefits over the [[7,1,3]] code. Also it has a high memory pseudo-threshold around 0.2%, assuming all the gates and memory have the same error rate. The [[9,1,3]] parallel scheme uses six ancillas for the Z-stabilizer measurements, where three ancillas can be reset for the X-stabilizer measurements in the second part. It is better than Bacon-Shor-13 at a high memory error rate at the cost of two additional qubits for parallelism.

Our procedure works for CSS codes. However, there are examples of non-CSS codes with shared flag qubits in [21,27]. A procedure for a general stabilizer code maybe be possible but it is beyond our scope.

In [27], Lao and Almudever proposed a two-dimensional layout for the [[7,1,3]] Steane code with a shared flag qubit [21]. We explain Reichardt’s parallel syndrome extraction with flag qubits [27] in the following. Figure 9 shows a parallel syndrome extraction circuit for the [[4,2,2]] code, whose circuit depth is lower than the cascade form in Fig. 1. Only two ancilla qubits are used to extract two syndrome bits, and each of them serves as a flag qubit for the other. Note that two SWAP gates are introduced, instead of CNOT gates.

We explain Reichardt’s parallel syndrome extraction with flag qubits [27] in the following. Figure 9 shows a parallel syndrome extraction circuit for the [[4,2,2]] code, whose circuit depth is lower than the cascade form in Fig. 1. Only two ancilla qubits are used to extract two syndrome bits, and each of them serves as a flag qubit for the other. Note that two SWAP gates are introduced, instead of CNOT gates.

Each single location failure in the parallel circuit Fig. 9 and its resulting data error are provided in Table V. where $X_a$ or $Z_a$ denotes a corresponding Pauli error at location $a$. A few notes on the table are as follows. The ancilla failures like $X_i$ and $Z_m$ do not introduce any data qubit errors; however, they cannot be identified from the

**TABLE V.** Memory and computation pseudo-thresholds using the two-step decoder.

| Scheme          | $\gamma = 0$     | $\gamma = 1$     |
|-----------------|------------------|------------------|
| [[9,1,3]] Flag  | 8.04 x 10^{-3}  | 1.56 x 10^{-3}  |
| [[9,1,3]] Parallel | 7.74 x 10^{-3}  | 2.01 x 10^{-3}  |
| Bacon-Shor-13   | 8.70 x 10^{-3}  | 1.19 x 10^{-3}  |

**ACKNOWLEDGMENTS**

CYL was supported by the Ministry of Science and Technology (MOST) in Taiwan, under Grant MOST110-2628-E-A49-007, MOST111-2119-M-A49-004, and MOST111-2119-M-001-002.

**Appendix A: Parallel flagged syndrome extraction**

FIG. 8. The circuit for the Bacon-Shor-13 scheme.

FIG. 9. Parallel syndrome extraction of $X_1X_2X_3X_4$ and $Z_1Z_2Z_3Z_4$ for the [[4,2,2]] code.

* This circuit is a modification of the circuit in [27] so that the code space is preserved.
TABLE VI. Equivalent errors on the data qubits for various single location failures and their syndromes in the parallel syndrome extraction circuit of $X_1X_2X_3X_4$ and $Z_1Z_2Z_3Z_4$ in Fig. [9] for the [[4,2,2]] code.

| Failure | Data error | $m_{1,2}$ | Failure | Data error | $m_{1,2}$ |
|---------|------------|-----------|---------|------------|-----------|
| $X_a$   | $X_1$      | $10$      | $Z_a$   | $Z_1$      | $01$      |
| $X_b$   | $X_2$      | $00$      | $Z_b$   | $Z_2$      | $01$      |
| $X_c$   | $X_3$      | $00$      | $Z_c$   | $Z_3$      | $01$      |
| $X_d$   | $X_4$      | $10$      | $Z_d$   | $Z_4$      | $01$      |
| $X_e$   | $X_1$      | $10$      | $Z_e$   | $Z_1$      | $00$      |
| $X_f$   | $X_2$      | $10$      | $Z_f$   | $Z_2$      | $01$      |
| $X_g$   | $X_3$      | $10$      | $Z_g$   | $Z_3$      | $01$      |
| $X_h$   | $X_4$      | $10$      | $Z_h$   | $Z_4$      | $00$      |
| $X_i$   | None       | $10$      | $Z_i$   | $Z_{1,2,3,4}$ | $00$      |
| $X_j$   | None       | $10$      | $Z_j$   | $Z_{2,4}$  | $01$      |
| $X_k$   | None       | $10$      | $Z_k$   | $Z_{2,4}$  | $01$      |
| $X_l$   | None       | $10$      | $Z_l$   | $Z_{2,4}$  | $01$      |
| $X_m$   | $X_{1,2,3,4}$ | $00$ | $Z_m$   | None      | $01$      |
| $X_n$   | $X_{2,3,4}$ | $10$ | $Z_n$   | None      | $01$      |
| $X_o$   | $X_{2,4}$  | $10$      | $Z_o$   | None      | $01$      |
| $X_p$   | $X_2$      | $00$      | $Z_p$   | None      | $01$      |

Table VI shows equivalent errors on the data qubits for various single location failures and their syndromes in the parallel syndrome extraction circuit of $X_1X_2X_3X_4$ and $Z_1Z_2Z_3Z_4$ in Fig. [9] for the [[4,2,2]] code.

 syndromes and the uncorrupted codewords have to be discarded. On the other hand, $X_m$ and $Z_i$ induce certain data errors that are stabilizer generators and hence the data qubits are not affected. Certain failures like $X_p$ and $Z_j$ induce weight-1 errors with zero syndrome but these errors will be detected and corrected in the next round of error detection.

Next we explain parallel syndrome extraction with shared flag qubits for the [[7,1,3]] Steane code [35], which is defined by the following stabilizer generators

$$
\begin{align*}
X_1X_3X_5X_7, & \quad Z_2Z_3Z_6Z_7, & \quad Z_4Z_5Z_6Z_7, \\
X_1Z_4Z_5Z_7, & \quad X_2X_3X_6X_7, & \quad X_4X_5X_6X_7.
\end{align*}
$$

A1

Reichardt demonstrated that two or three stabilizer generators in Eq. (A1) can be simultaneously measured in a fault-tolerant way [27]. Figure (10) illustrates the case of measuring three stabilizers $X_1X_3X_5X_7$, $Z_2Z_3Z_6Z_7$, and $Z_4Z_5Z_6Z_7$ in parallel and only three ancillas are required [27]. The two additional CNOTs connecting the ancillas are elegantly placed so that each of the ancillas can serve as a flag qubit for the others. The block $U$ in Fig. (10) is a complete unflagged syndrome extraction circuit. $U$ will be applied if anyone of the binary measurement outcomes $m_1$, $m_2$, $m_3$ is 1. The remaining three stabilizers can be similarly measured in parallel.

This idea also works for the [[5,1,3]] code and the [[15,7,3]] CSS code so that two stabilizers can be measured in parallel but an extra flag qubit is necessary [27].

[1] 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).

[2] C. Vuillot, Is error detection helpful on IBM 5Q chips?, Quant. Inf. Comput. 18, 0949 (2018).

[3] L. Postler, S. Heußen, I. Pogorelov, M. Rispler, T. Feldker, M. Meth, C. D. Marciniak, R. Stricker, M. Ringbauer, R. Blatt, P. Schindler, M. Müller, and T. Monz, Demonstration of fault-tolerant universal quantum gate operations (2021).

[4] 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 598, 281 (2021).

[5] D. Aharonov and M. Ben-Or, Fault-tolerant quantum computation with constant error, in Proceedings of the twenty-ninth annual ACM symposium on Theory of computing STOC ’97 (ACM, New York, NY, USA, 1997) pp. 176–188.

[6] P. W. Shor, Fault-tolerant quantum computation, in Proceedings of the 37th Annual Symposium on the Theory of Computer Science (IEEE Press, Los Alamitos, 1996) pp. 56–65.

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

[8] E. Knill, R. Laflamme, and W. Zurek, Threshold accuracy for quantum computation (1996).

[9] D. Gottesman, Stabilizer codes and quantum error correction, Ph.D. thesis, California Institute of Technology, Pasadena, CA (1997).

[10] J. Preskill, Reliable quantum computers, in Proc. R. Soc. London A, Vol. 454 (1998) pp. 385–410.

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

[12] A. M. Steane, Efficient fault-tolerant quantum computing, Nature 399, 124 (1999).

[13] E. Knill, Quantum computing with realistically noisy devices, Nature 434, 39 (2005).

[14] A. Ashikhmin, C.-Y. Lai, and T. A. Brun, Quantum
data-syndrome codes, IEEE J. Sel. Area. Comm. 38, 449 (2020).
[15] K.-Y. Kuo, I.-C. Chern, and C.-Y. Lai, Decoding of quantum data-syndrome codes via belief propagation, in 2021 IEEE International Symposium on Information Theory (ISIT) (2021) pp. 1552–1557.
[16] Y.-C. Zheng, C.-Y. Lai, T. A. Brun, and L.-C. Kwek, Constant depth fault-tolerant clifford circuits for multi-qubit large block codes, Quant. Sci. Tech. 5, 045007 (2020).
[17] S. Huang and K. R. Brown, Between shor and steane: A unifying construction for measuring error syndromes, Phys. Rev. Lett. 127, 090505 (2021).
[18] 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).
[19] M. Li, D. Miller, and K. R. Brown, Direct measurement of bacon-shor code stabilizers, Phys. Rev. A 98, 050301 (2018).
[20] T. J. Yoder and I. H. Kim, The surface code with a twist, Quantum 1, 2 (2017).
[21] R. Chao and B. W. Reichardt, Quantum error correction with only two extra qubits, Phys. Rev. Lett. 121, 050502 (2018).
[22] R. Chao and B. W. Reichardt, Fault-tolerant quantum computation with few qubits, npj Quantum Inf. 4, 42 (2018).
[23] C. Chamberland and M. E. Beverland, Flag fault-tolerant error correction with arbitrary distance codes, Quantum 2, 53 (2018).
[24] C. Chamberland and A. W. Cross, Fault-tolerant magic state preparation with flag qubits, Quantum 3, 143 (2019).
[25] T. Tansuwannont, C. Chamberland, and D. Leung, Flag fault-tolerant error correction, measurement, and quantum computation for cyclic calderbank-shor-steane codes, Phys. Rev. A 101, 012342 (2020).
[26] R. Chao and B. W. Reichardt, Flag fault-tolerant error correction for any stabilizer code, PRX Quantum 1, 010302 (2020).
[27] B. W. Reichardt, Fault-tolerant quantum error correction for steane’s seven-qubit color code with few or no extra qubits, Quantum Sci. Tech. 6, 015007 (2020).
[28] A. M. Steane, Multiple particle interference and quantum error correction, Proc. R. Soc. London A 452, 2551 (1996).
[29] P. W. Shor, Scheme for reducing decoherence in quantum computer memory, Phys. Rev. A 52, 2493 (1995).
[30] K. M. Svore, D. P. Divincenzo, and B. M. Terhal, Noise threshold for a fault-tolerant two-dimensional lattice architecture, Quant. Inf. Comput. 7, 297 (2007).
[31] F. M. Spedalieri and V. P. Roychowdhury, Latency in local, two-dimensional, fault-tolerant quantum computing, Quant. Inf. Comput. 9, 666 (2009).
[32] C.-Y. Lai, G. Paz, M. Suchara, and T. Brun, Performance and error analysis of Knill’s postselection scheme in a two-dimensional architecture, Quant. Inf. Comput. 14, 807 (2014).
[33] F. J. MacWilliams and N. J. A. Sloane, The Theory of Error-Correcting Codes (North-Holland, Amsterdam, The Netherlands, 1977).
[34] A. R. Calderbank and P. W. Shor, Good quantum error-correcting codes exist, Phys. Rev. A 54, 1098 (1996).
[35] A. M. Steane, Error correcting codes in quantum theory, Phys. Rev. Lett. 77, 793 (1996).
[36] P. Aliferis, D. Gottesman, and J. Preskill, Quantum accuracy threshold for concatenated distance-3 codes, Quant. Inf. Comput. 6, 97 (2006).
[37] L. Lao and C. G. Almudever, Fault-tolerant quantum error correction on near-term quantum processors using flag and bridge qubits, Phys. Rev. A 101, 032333 (2020).