Tailoring Fault-Tolerance to Trotter Circuits

Zhuangzhuang Chen and Narayanan Rengaswamy
Department of Electrical and Computer Engineering, University of Arizona, Tucson, USA 85721
E-mail: { zhuangzhuangchen , narayananr }@arizona.edu

***Abstract*—The standard approach to universal fault-tolerant quantum computing is to develop a general purpose quantum error correction mechanism that can implement a universal set of logical gates fault-tolerantly. However, we know that quantum computers provide a significant quantum advantage only for specific quantum algorithms. Hence, a universal quantum computer can likely gain from compiling such specific algorithms using tailored quantum error correction schemes. In this work, we take the first steps towards such algorithm-tailored quantum fault-tolerance. We consider Trotter circuits in quantum simulation, which is an important application of quantum computing. We develop a solve-and-stitch algorithm to systematically synthesize physical realizations of Clifford Trotter circuits on the well-known $[\![n, n-2, 2]\!]$ error-detecting code family. Our analysis shows that this family implements Trotter circuits with optimal depth, thereby serving as an illuminating example of tailored quantum error correction. We achieve fault-tolerance for these circuits using flag gadgets, which add minimal overhead. The solve-and-stitch algorithm has the potential to scale beyond this specific example and hence provide a principled approach to tailored fault-tolerance in quantum computing.**

***Index Terms*—Quantum error correction, fault-tolerance, Trotter circuits, quantum simulation, error-detecting code, Clifford gates, flag gadgets, logical Clifford synthesis (LCS)**

I. Introduction

Quantum error correction (QEC) is fundamental to the realization of scalable fault-tolerant quantum computers. In recent years, QEC has moved from theory to practice where there have been several demonstrations of small error corrected systems [1]–[4]. The next frontier is the development of *scalable* QEC schemes that enable significant quantum advantages for *certain* problem domains, when compared to the best classical supercomputers. The common approach is to pursue universal fault-tolerant quantum computing where a general-purpose QEC scheme is shown to fault-tolerantly realize a universal set of logical operations on the encoded information [5]–[7]. Given such a scheme, one can execute any quantum algorithm on the logical qubits by *composing* elements of this fault-tolerant universal set. In parallel, quantum algorithms continue to be developed for various problems. However, it is widely expected that significant quantum advantage will be achieved only for some targeted problems and applications [8]–[10]. Hence, even in a universal fault-tolerant quantum computer, there are likely gains to be achieved by compiling such specific algorithms using *tailored* QEC mechanisms. More specifically, rather than composing *individual* gates from the universal set, it could be much more efficient to directly design fault-tolerant realizations of the *logical circuit as a whole*. Such exciting opportunities form the primary motivation for this work.

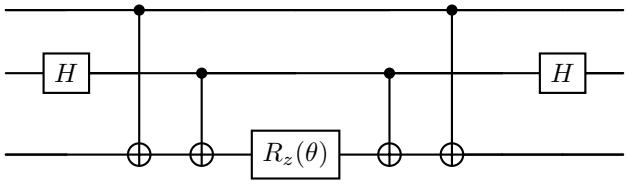


Fig. 1: Quantum Simulation Kernel (QSK) with 3 qubits

At the outset, it is unclear how one can pose the goal of tailoring QEC schemes to logical algorithms as a systematic mathematical problem. When can we say that a QEC scheme is "well-matched" to execute an algorithm? In this paper, we work with the objective of achieving a depth-optimal fault-tolerant realization of a given logical circuit. Since quantum simulation is a key motivation and application of quantum computers, we consider the problem of executing *Trotter circuits* fault-tolerantly with optimal depth [11], [12]. A Trotter circuit is sometimes referred to as the *quantum simulation kernel (QSK)* [13], so we will use these terminologies interchangeably. An example QSK circuit for 3 qubits is shown in Fig. 1, where the Z-rotation angle θ depends on the specifics of the simulation algorithm. For this work we set $\theta = \frac{\pi}{2}$ throughout, i.e., consider Clifford [14] QSK (C-QSK) circuits, to take the first steps towards the above goal. We develop a principled approach of deriving the physical realization of the logical C-QSK circuit on the well-known error-detecting family of $[\![n, n-2, 2]\!]$ codes [15], [16]. The methods have the potential to generalize beyond this family and also be amenable to optimization with respect to compilation constraints in the hardware.

We consider a *bottom-up* approach to synthesizing physical realizations of logical C-QSK circuits on the error-detecting code family. The key idea is to characterize the logical circuit via input-output Pauli tracking constraints, solve for separate small circuits to satisfy each of these constraints (the "solve" step), and then suitably merge these circuits to simultaneously satisfy all constraints (the "stitch" step), thereby realizing the logical circuit. We refer to this as the *solve-and-stitch* method. For this code family, we achieve fault-tolerance by adding flag gadgets [16] to the two-qubit gates in the solutions and proving that any single fault in the circuit remains detectable at the end. Most excitingly, by leveraging some properties of the code family, we prove that the depth of these fault-tolerant physical circuits only grows *linearly in the number of logical qubits* k. Since the logical QSK circuit itself has depth $2k+1$ (from $2(k-1)$ CNOTs, the Z-rotation, and two layers of single-

qubit gates), the depth of these physical circuits is optimal. While $k = n-2$ for this family, which means that the depth is effectively linear in n, the proof involves calculations primarily in terms of the variable k. Therefore, the proof strategy has the potential to extend beyond this specific code family. Refer to the full version of this work [17] for all technical details.

Overall, our results suggest that the error-detecting code family is indeed well-matched to efficiently realize logical C-QSK circuits fault-tolerantly. Obviously, the code doesn't allow one to correct errors, so we will consider extending the solve-and-stitch approach to more non-trivial code families in future work. In such cases, decoders can also be tailored to these circuits since the error propagation has more structure.

The remainder of the paper is organized as follows. Section II provides necessary background to understand the details of this work. Section III explores transversal implementations of the logical C-QSK circuits and shows that this appears to be impossible using stabilizer codes. Section IV develops the solve-and-stitch method to physically realize logical C-QSK circuits on the error-detecting code family. It also provides the depth analysis for the circuits produced by the method. Finally, Section V concludes the work and highlights opportunities for future work related to our results.

II. Preliminaries

A. Pauli Group and Stabilizer Codes

For a single qubit, the Hermitian Pauli operators are denoted by I, X, Y, Z. For $n \geq 1$ qubit(s), the *Pauli group* is given by

$$\mathcal{P}_n \coloneqq \langle \imath^\kappa I, X, Y, Z \,;\, \kappa \in \{0,1,2,3\}\rangle^{\otimes n}. \tag{1}$$

The *weight* of a Pauli operator is the number of non-identity elements in its tensor product, e.g., $X \otimes Z \otimes Y$ has weight 3.

Let $N \coloneqq 2^n$ for an n-qubit system. A *stabilizer group*, $\mathcal{S}$, is a group generated by commuting Hermitian Pauli operators such that $-I_N \notin \mathcal{S}$. The associated *stabilizer code*, $\mathcal{Q}_\mathcal{S}$, is the subspace of unit vectors in $\mathbb{C}^N$ that are the common $+1$-eigenvectors of operators in $\mathcal{S}$. In other words, each $|\psi\rangle \in \mathcal{Q}_\mathcal{S}$ satisfies $S|\psi\rangle = |\psi\rangle$ for all $S \in \mathcal{S}$. The *logical Pauli operators* of a code are Hermitian Pauli operators that commute with the stabilizers but are not stabilizers themselves. If $\mathcal{S}$ has $r = n-k$ independent generators and the minimum weight of any logical Pauli operator is d, then the code is an $[\![n,k,d]\!]$ code. Such a code encodes k logical qubits into n physical qubits. It can detect up to $(d-1)$ Pauli errors but can correct only up to $\lfloor \frac{d-1}{2} \rfloor$ Pauli errors using a maximum-likelihood decoder. A *CSS* code is a stabilizer code whose stabilizer group can be generated by purely X-type and purely Z-type operators.

B. The $[\![n, n-2, 2]\!]$ Code Family

This is a family of CSS codes with exactly two stabilizer generators, $X_1X_2\cdots X_n$ and $Z_1Z_2\cdots Z_n$, where n is set to be even. It is easily checked that these generators commute. The generators of logical-X and Z operators are of the form $\overline{X_i} = X_1X_{i+1}$ and $\overline{Z_i} = Z_{i+1}Z_n$, where $i = 1, 2, \ldots, n-2$. Hence, the distance of the codes in this family is always 2. The family is only error-detecting because it can detect up to 1 error but cannot correct any errors. As an example, for $n = 6$, the $[\![6,4,2]\!]$ is described by the stabilizer group $\mathcal{S} = \langle X_1\cdots X_6, Z_1\cdots Z_6\rangle$ and the logical Pauli generators

$$\begin{aligned}
&\overline{X_1} = X_1X_2 \ , \ \overline{Z_1} = Z_2Z_6 \ ; \\
&\overline{X_2} = X_1X_3 \ , \ \overline{Z_2} = Z_3Z_6 \ ; \\
&\overline{X_3} = X_1X_4 \ , \ \overline{Z_3} = Z_4Z_6 \ ; \\
&\overline{X_4} = X_1X_5 \ , \ \overline{Z_4} = Z_5Z_6 \ .
\end{aligned} \tag{2}$$

We will use this code as our running example in this paper.

C. Quantum Simulation Kernel (QSK)

In quantum (Hamiltonian) simulation, the circuit to implement an exponentiated Pauli operator $\exp(\imath\theta E)$ is called a *Trotter* circuit or a *quantum simulation kernel (QSK)* circuit. An example for 3 qubits is shown in Fig. 1, which corresponds to the case $E = Z_1X_2Z_3$. The structure generalizes for any k-qubit Pauli: start with single-qubit gates on those qubits that have a non-Z component in E (Hadamard H for X or $H_y \coloneqq \frac{1}{\sqrt{2}}\left[\begin{smallmatrix} 1 & -\imath \\ \imath & -1 \end{smallmatrix}\right]$ for Y), then perform CNOTs from the first $(k-1)$ qubits (control) to the k-th qubit (target), then perform a Z-rotation $R_z(\theta)$ of the k-th qubit, and finally apply the gates before the rotation in the reverse order. It is easily seen that the depth of this circuit is $(2k+1)$, i.e., linear in k.

We only consider Clifford QSK (C-QSK) circuits, so we set $\theta = \frac{\pi}{2}$ (i.e., $R_z(\theta) = P \coloneqq \sqrt{Z}$). In this case the circuit is completely characterized by how it maps Pauli operators at its input to Pauli operators at its output using standard Pauli conjugation relations for Clifford gates. For the $k = 3$ circuit in Fig. 1, one can check the following input-output mappings:

$$\begin{aligned}
&X_1 \mapsto Y_1X_2Z_3 \quad , \quad Z_1 \mapsto Z_1 \ , \\
&X_2 \mapsto X_2 \qquad\ \ , \quad Z_2 \mapsto -Z_1Y_2Z_3 \ , \\
&X_3 \mapsto Z_1X_2Y_3 \quad , \quad Z_3 \mapsto Z_3 \ .
\end{aligned} \tag{3}$$

Hence, for any $[\![n,k,d]\!]$ stabilizer code, the physical realization of the logical C-QSK circuit must satisfy the above relations for its k *logical* Pauli operators. Besides, the physical circuit must also preserve the stabilizer group $\mathcal{S}$ under conjugation.

III. Fault-Tolerant QSK via Transversality

A natural fault-tolerant implementation of logical circuits is via a *transversal* physical operation, i.e., one that splits into a Kronecker product of single-qubit gates. Any fault in such a physical circuit does not propagate into other qubits. While the Eastin-Knill theorem [18] prevents an error-detecting stabilizer code from implementing a *universal* set of gates fault-tolerantly, there might exist a stabilizer code that realizes the specific logical C-QSK circuit transversally. In this section, we pursue this possibility and consider transversal Hadamard or Phase gates for implementing the logical C-QSK circuit.

Theorem 1. *There exists no stabilizer code where transversal Hadamard realizes the logical C-QSK circuit.*

Theorem 2. *There exists no stabilizer code where transversal Phase realizes the logical C-QSK circuit.*

In our proofs [17], we have refrained from specifying any particular code, leading us to the compelling conclusion that finding a stabilizer code capable of realizing the logical C-QSK circuits via transversal combinations of Hadamard and Phase gates *appears* inherently impossible. Note that this encompasses all transversal Clifford circuits since Hadamard and Phase gates generate the single-qubit Clifford group. Remarkably, based on the properties of C-QSK circuits, this conclusion can likely be extended to C-QSK circuits of arbitrary size and combinations of H and H_y gates. Our analysis suggests that leveraging stabilizer degrees of freedom is unlikely to alter this conclusion. Of course, all this remains to be proven rigorously, hence we leave it as a conjecture.

Conjecture 3. *There exists no stabilizer code where transversal Clifford gates realize the logical C-QSK circuit for any non-trivial exponentiated Pauli operator.*

IV. Realizing C-QSK on $[\![n, n-2, 2]\!]$ Code Family

The discussion in Sec. III implies that transversal implementations of logical C-QSK circuits seem inherently impossible. Hence, we must incorporate two-qubit gates in the construction of physical circuits. This, however, raises a crucial question:

Given specific input-output Pauli mapping rules dictated by the logical C-QSK circuit on a stabilizer code, what can we infer about the structure of the physical circuit satisfying these Pauli constraints, even without considering fault-tolerance?

Our goal in this section is to address this question by developing a principled approach to circuit synthesis that allows us to track structural information during the synthesis. Naturally, such an approach provides a feasible solution that upper bounds depth, number of two-qubit gates etc.

The *Logical Clifford Synthesis (LCS)* algorithm [19] systematically synthesizes physical (Clifford) circuits satisfying Pauli constraints imposed by the logical (Clifford) circuit and code structure. This includes the constraints to ensure that the stabilizer group is preserved under conjugation by the physical Clifford circuit. The algorithm formulates the Pauli constraints as linear equations on a target binary symplectic matrix representing the desired physical Clifford circuit. Then it systematically solves for all feasible symplectic solutions by using symplectic transvections. Finally, it decomposes each solution into elementary Clifford gates. Despite its value, the LCS algorithm falls short of elucidating circuit properties and structure *during the construction of solutions*. This is because it directly finds the symplectic representation of the full circuit, which hides the circuit structure until it is decomposed into elementary gates. Therefore, one must determine all solutions before identifying the most desirable circuit. Unfortunately, since the number of solutions is $2^{r(r+1)/2}$, where $r = n - k$, the computational efficiency decreases super-exponentially with the dimension of the stabilizer group of the code.

We circumvent these issues of the LCS algorithm by developing a *bottom-up* approach to synthesize physical circuits. The key idea is to "solve" for a small circuit satisfying one Pauli constraint by identifying a *root qubit*. Such a qubit is involved in both the input and output Pauli operator, but the Pauli acting on that qubit changes from input to output. Then these small circuits for different constraints are "stitched" together appropriately to satisfy all constraints simultaneously. By employing this approach for logical C-QSK circuits on $[\![n, n-2, 2]\!]$ codes, we can track circuit properties, such as depth, *during the construction of the circuit*. Notably, this method yields results comparable to the optimal circuits generated by LCS for this code family, but without enumerating all solutions. The following section illustrates such realization of logical C-QSK circuits under different scenarios.

A. Circuit Synthesis via Solve-and-Stitch Approach

For an illustrative example, we assume that the logical C-QSK circuit consists of an even number of Hadamard gates. The $[\![6, 4, 2]\!]$ code serves as an excellent testbed for our exploration, in which case the logical circuit comprises 4 qubits. As an example, consider the circuit in Fig. 2, where the Hadamard gates are applied to the second and third qubits. The circuit implements the exponentiated Pauli operator $\exp\left(-\imath\frac{\pi}{4}Z_1X_2X_3Z_4\right)$. A similar approach applies for Pauli operators with Y entries, where H gates are replaced with H_y gates (see Section II-C).

This C-QSK circuit dictates the following input-output mappings of logical Pauli operators:

$$\begin{aligned}
&\overline{X_1} \mapsto \overline{Y_1}\,\overline{X_2}\,\overline{X_3}\,\overline{Z_4} \quad , \quad \overline{Z_1} \mapsto \overline{Z_1} \ , \\
&\overline{X_2} \mapsto \overline{X_2} \quad , \quad \overline{Z_2} \mapsto -\overline{Z_1}\,\overline{Y_2}\,\overline{X_3}\,\overline{Z_4} \ , \\
&\overline{X_3} \mapsto \overline{X_3} \quad , \quad \overline{Z_3} \mapsto -\overline{Z_1}\,\overline{X_2}\,\overline{Y_3}\,\overline{Z_4} \ , \\
&\overline{X_4} \mapsto \overline{Z_1}\,\overline{X_2}\,\overline{X_3}\,\overline{Y_4} \quad , \quad \overline{Z_4} \mapsto \overline{Z_4} \ .
\end{aligned} \tag{4}$$

Substituting for the logical operators of the $[\![6, 4, 2]\!]$ code, we obtain the following mappings of physical Pauli operators:

$$\begin{aligned}
&X_1X_2 \mapsto X_1Y_2X_3X_4Z_5 \quad , \quad Z_2Z_6 \mapsto Z_2Z_6 \ , \\
&X_1X_3 \mapsto X_1X_3 \quad , \quad Z_3Z_6 \mapsto -Z_2Y_3X_4Z_5Z_6 \ , \\
&X_1X_4 \mapsto X_1X_4 \quad , \quad Z_4Z_6 \mapsto -Z_2X_3Y_4Z_5Z_6 \ , \\
&X_1X_5 \mapsto X_1Z_2X_3X_4Y_5 \quad , \quad Z_5Z_6 \mapsto Z_5Z_6 \ .
\end{aligned} \tag{5}$$

Additionally, we require that the two stabilizer generators $X_1 \cdots X_6$ and $Z_1 \cdots Z_6$ are mapped to themselves, even though it is sufficient to map them to a pair of equivalent stabilizers, i.e., normalize the stabilizer group. The minimum depth solution produced by the LCS algorithm for this case is shown below in Fig. 3. This physical circuit satisfies all

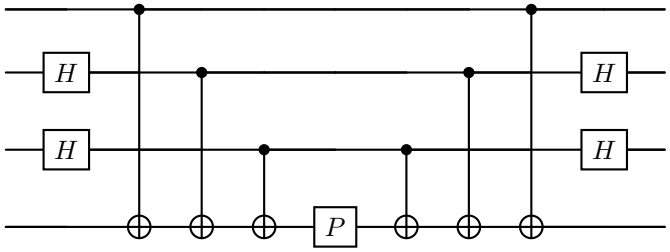


Fig. 2: C-QSK circuit with even number of Hadamard gates

stipulated Pauli constraints above while preserving stabilizers. Observe that the (naïve) depth of this circuit is 10, which is comparable to the depth of the logical circuit in Fig. 2.

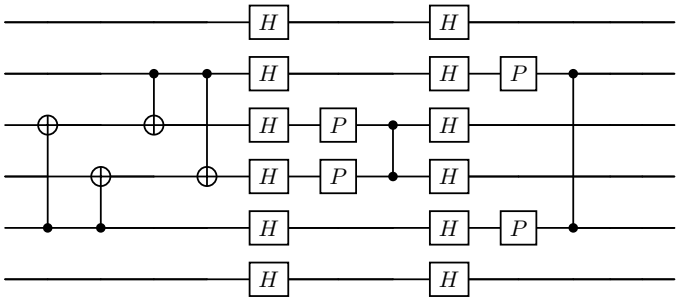


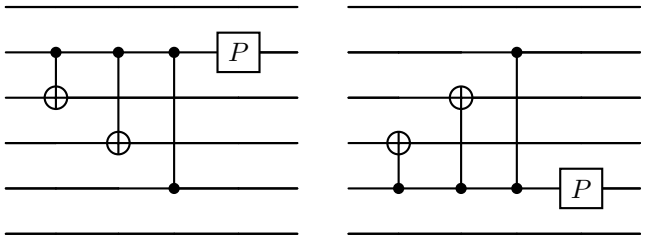
Fig. 3: The minimum depth solution of LCS algorithm for realizing the logical circuit in Fig. 2 on the $[\![6, 4, 2]\!]$ code. Note that the algorithm does not perform circuit simplification.

For the $[\![n, n-2, 2]\!]$ code family, we have $r = n - k = 2$, so the number of symplectic solutions (for a fixed set of Pauli constraints) is only $2^{r(r+1)/2} = 8$. Hence, it is easy to enumerate the solutions and identify the minimum depth circuit. However, this becomes impractical even for $r = 6$. Besides, the LCS algorithm does not enable us to predict the minimum depth until we obtain all solutions and decompose each of them into elementary Clifford gates as in Fig. 3.

To address these limitations, we propose an alternative method for constructing physical circuits by identifying a *root* qubit for each (physical) Pauli constraint. For a non-trivial constraint, we identify qubits that appear in both the input and output, and then eliminate the qubits with an unchanged Pauli gate. From the remaining root qubits, we choose one as the root. For instance, consider the mapping of X_1X_2 to $X_1Y_2X_3X_4Z_5$. In this case, we choose the second qubit as the root rather than the first qubit because X_1 remains unchanged, requiring no additional gates in the circuit. For (non-root) qubits that are absent in the input Pauli but present in the output of the constraint, we introduce two-qubit gates controlled by the root qubit because single-qubit gates like H and P cannot affect the Pauli on other qubits.

Specifically, for the above example, since CNOT gate maps XI to XX, we apply $\text{CNOT}_{2\to3}$ and $\text{CNOT}_{2\to4}$ for the third and fourth qubits with the second qubit as the root (control). Since the CZ gate maps XI to XZ, we apply $\text{CZ}_{2\to5}$ for the fifth qubit to satisfy its mapping. The output term of the second qubit is a Y gate, so we directly introduce a Phase gate to achieve the desired mapping at the end. This "local" circuit construction is illustrated in Fig. 4a.

For mappings involving $\overline{X_2}$ and $\overline{X_3}$, where the input gates are the same as the output gates, we have verified that these mappings remain unchanged as the gates propagate through the above rooted circuit we constructed. Consequently, there is no need to construct new rooted circuits for these trivial mappings. Additionally, for the Pauli constraint of $\overline{X_4}$, we choose the fifth qubit as the root and apply CNOT or CZ gates to other qubits based on the input-output Pauli relations. A Phase gate is then applied to the root qubit to complete the rooted circuit in Fig. 4b for mapping X_1X_5 to $X_1Z_2X_3X_4Y_5$.



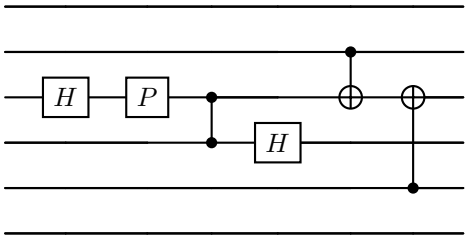
(a) $X_1X_2 \mapsto X_1Y_2X_3X_4Z_5$ (b) $X_1X_5 \mapsto X_1Z_2X_3X_4Y_5$

Fig. 4: Rooted circuit construction for logical-X constraints.

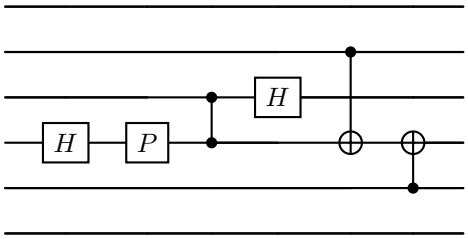
For the mappings corresponding to the logical-Z constraints, the construction of rooted circuits has some variations. For example, consider the mapping $Z_3Z_6 \mapsto Z_2Y_3X_4Z_5Z_6$. Here, just as before, we designate the third qubit as the root rather than the sixth qubit because Z_6 remains unchanged. However, unlike the logical-X constraints, where the root's mapping is processed at the end, here we process this qubit at the beginning. To transform Z_3 to Y_3 in the output, we apply H followed by P on the third qubit. Next, we observe that the output of the fourth qubit is X_4. While it might seem intuitive to apply $\text{CNOT}_{3\to4}$, which maps Y_3I_4 to Y_3Z_4, we realize that the constraint $Z_4Z_6 \mapsto Z_2X_3Y_4Z_5Z_6$ would later require a reversed CNOT, i.e., $\text{CNOT}_{4\to3}$. Since the CNOT is not symmetric with respect to swapping the control and target qubits, we apply a CZ gate on the third and fourth qubits instead, followed by H on these qubits. Subsequently, we apply CNOTs for qubits 2 and 5, with the root qubit as the target and themselves as controls. Finally, as for the logical-X constraints, we verify that these rooted circuits satisfy the trivial mappings of $\overline{Z_1}$ and $\overline{Z_4}$ as well. The final rooted circuits for logical-Z constraints are illustrated in Figs. 5a and 5b.

To consolidate the rooted circuits satisfying individual Pauli constraints into a comprehensive physical circuit that simultaneously meets all constraints, we follow a systematic "stitching" approach, initially addressing logical-Xs and subsequently logical-Zs. We begin with the rooted circuit for $\overline{X_1}$, which we have verified adheres to the trivial mappings of $\overline{X_2}$ and $\overline{X_3}$. Subsequently, we seamlessly append the rooted circuit for $\overline{X_4}$ at the end. During this process, we ensure smooth integration of shared gates; for example, since CZ_{25}, is already present, we incorporate it without duplication. Then we consolidate all P gates into a single stage.

Upon checking all mappings of the logical-X part, we confirm the circuit's validity. Moving to the rooted circuits for the logical-Z constraints in Fig. 5, we observe that the two CNOT gates in Fig. 5a have already been incorporated in the logical-X part. Consequently, we exclude them and proceed to add H_3, P_3, CZ_{34}, and H_4 into the final circuit. At this stage, we verify that all prior Pauli constraints are still satisfied. Next, in Fig.5b, we observe a similar H-P-CZ-H structure, sharing the same CZ gate. This structure is



(a) $Z_3Z_6 \mapsto Z_2Y_3X_4Z_5Z_6$ (ignoring sign)



(b) $Z_4Z_6 \mapsto Z_2X_3Y_4Z_5Z_6$ (ignoring sign)

Fig. 5: Rooted circuit construction for logical-Z constraints.

consolidated into the physical circuit, as depicted in Fig. 6. Once again, the CNOT gates in Fig. 5b are already present in previous rooted circuits, so we do not duplicate them. Through this process, we validate that all Pauli constraints are satisfied simultaneously, the two stabilizer generators are preserved, and the depth aligns with the circuit generated by the LCS algorithm in Fig. 3. The notable differences lie in the repositioning of certain gates and the cancellation of some Hadamard gates. Hence, we have a bottom-up approach to derive the best output of the LCS algorithm in this case, while being able to track the structure of the circuit at each stage of the process. This directly addresses the question we posed at the beginning of Section IV.

In the case of logical C-QSK circuits with an odd number of Hadamard gates, we encounter some variations in the established pattern. However, the fundamental idea of root qubits and the solve-and-stitch mechanism continues to prove valuable and the complete circuit structure is shown in Fig. 7.

While it may appear that we must check for past constraints when stitching each rooted circuit to the existing physical circuit, this was done for pedagogical reasons. In the next section, we prove rigorously that the stitching procedure always satisfies all constraints for logical C-QSK circuits on this code family. Therefore, the validity of the circuits is established analytically, and for all members of the code family.

B. Solve-and-Stitch Approach on all $[\![n, n-2, 2]\!]$ Codes

Define the set $[k] \coloneqq \{1, 2, 3, ..., k\}$. Let $i, j \in [k]$ index (logical) qubits of the C-QSK circuit. Let $I_h \subseteq [k]$ be the index set of qubits where Hadamard gates appear in the C-QSK circuit. Recall that the set I_h corresponds to qubits with an X in the exponential Pauli operator corresponding to Trotter circuits. For convenience of analysis, we do not discuss operators with Y terms, which correspond to H_y gates in Sec. II-C, but the ideas here will likely generalize.

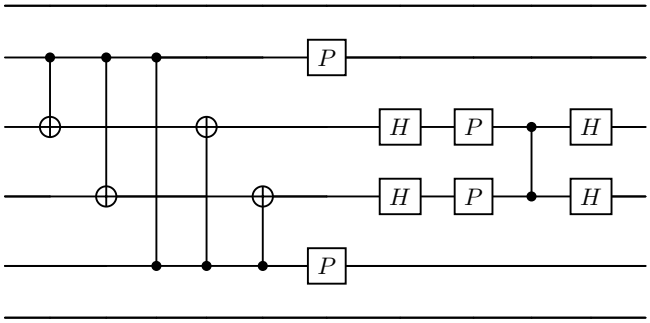


Fig. 6: Physical realization of Fig. 2 on the $[\![6, 4, 2]\!]$ code produced by the solve-and-stitch approach using root qubits. Signs for $\overline{Z_2}$ and $\overline{Z_3}$ can be fixed with Pauli gates at the end.

Lemma 4. *The logical Pauli mappings of the C-QSK circuit on k qubits can be expressed as follows:*

1) *If $i \notin I_h$, then*

$$\overline{X_i} \mapsto \overline{Y_i} \left(\prod_{j \in I_h} \overline{X_j} \right) \left(\prod_{j \in [k] \setminus (I_h \cup \{i\})} \overline{Z_j} \right), \quad (6)$$

$$\overline{Z_i} \mapsto \overline{Z_i}. \quad (7)$$

2) *If $i \in I_h$, then*

$$\overline{X_i} \mapsto \overline{X_i}, \quad (8)$$

$$\overline{Z_i} \mapsto -\overline{Y_i} \left(\prod_{j \in I_h \setminus \{i\}} \overline{X_j} \right) \left(\prod_{j \in [k] \setminus I_h} \overline{Z_j} \right). \quad (9)$$

We can substitute for the logical Pauli operators in the above result. For the $[\![n, n-2, 2]\!]$ codes, logical Pauli operators are defined as $\overline{X_i} = X_1X_{i+1}, \overline{Z_i} = Z_{i+1}Z_n, \overline{Y_i} = X_1Y_{i+1}Z_n$. These physical operators provide the physical Pauli mappings for logical C-QSK circuits on these codes for the cases of even and odd number of Hadamards. The primary distinction lies in the parity of the number of elements in I_h, which governs the number of terms in the products in Lemma 4.

Armed with these general expressions for desired Pauli mappings, we can investigate the solve-and-stitch algorithm carefully. The "solve" phase of the algorithm solves for small rooted circuits, one for each mapping, by identifying root qubits. Let us first focus on the logical-X mappings.

Lemma 5. *The rooted circuits for logical-X constraints in Lemma 4 can be stitched by concatenating them and dropping duplicate CZ gates. The stitched circuit satisfies all the logical-X constraints simultaneously.*

The proofs of all results can be found in [17].

Lemma 6. *The rooted circuits for logical-Z constraints in Lemma 4 can be stitched by concatenating them and dropping duplicate CZ gates. The stitched circuit satisfies all the logical-Z constraints simultaneously.*

$\text{CZ}_{i+1,j+1}$ gates $i \notin I_h$ $j \notin I_h$	$\text{CZ}_{i+1,n}$ gates $i \notin I_h$	P_n or $P_n^\dagger$, P_{i+1} gates $i \notin I_h$	$\text{CX}_{i+1\to j+1}$ gates $i \notin I_h$ $j \in I_h$	$\text{CX}_{n\to 1}$, $\text{CX}_{i+1\to 1}$ gates $i \notin I_h$	$\text{CX}_{n\to i+1}$ gates $i \in I_h$	H_1-P_1, H_{i+1}-P_{i+1} gates $i \in I_h$	$\text{CZ}_{i+1,1}$, $\text{CZ}_{i+1,j+1}$ gates $i,j \in I_h$	H_1, H_{i+1} gates $i \in I_h$

Fig. 7: The sequence of gates in the full physical circuit realizing the logical C-QSK (i.e., Clifford Trotter) circuit on the $[\![n, n-2, 2]\!]$ codes. The set $I_h \subseteq [n-2]$ consists of the indices of logical qubits on which the logical circuit applies a Hadamard. The white blocks and non-highlighted gates are common to the cases of even and odd number of Hadamards, i.e., $|I_h|$, but the gray blocks and the highlighted gates are needed only for the case of odd number of Hadamards.

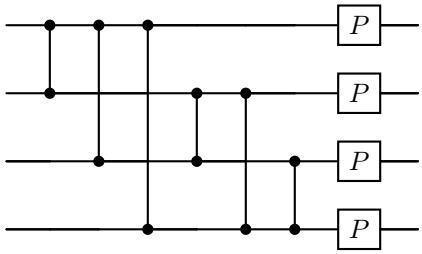


Fig. 8: Logical identity gadget for the $[\![4, 2, 2]\!]$ code.

Now, we provide the main result of this section: the construction of the complete physical circuit for logical C-QSK on $[\![n, n-2, 2]\!]$ codes by stitching the logical-X and logical-Z stitched circuits, as shown in Fig. 7 (see [17] for pseudocode).

Theorem 7. *The logical C-QSK (i.e., Clifford Trotter) circuit can be realized on the $[\![n, n-2, 2]\!]$ codes by concatenating the logical-X and logical-Z stitched circuits, then dropping duplicate CNOT gates. If the number of Hadamards in the logical circuit is odd, then CNOTs and Phase gates can be added to preserve the stabilizer group.*

Theorem 8. *Consider the physical circuit constructed by the solve-and-stitch algorithm in Theorem 7 to realize a logical C-QSK circuit on the $[\![n, n-2, 2]\!]$ codes. The depth of this circuit is upper bounded by $\frac{k(k-1)}{2} + 5$ when h is even and $\frac{(k+2)(k+1)}{2} + 5$ when h is odd, where $k = n-2$.*

We have demonstrated that the depth of physical circuits grows quadratically with the dimension of the $[\![n, n-2, 2]\!]$ codes. To mitigate this, we propose the incorporation of "complementary" CZ gates at the beginning of the physical circuit. When combined with P gates on all n qubits, these CZ gates only implement the logical identity on the code, thereby not affecting the logical functionality of the physical circuit. Figure 8 illustrates an example of this logical identity circuit for the $[\![4, 2, 2]\!]$ code, which can be extended to any member of the $[\![n, n-2, 2]\!]$ code family. Indeed, this CZ-P gadget induces the mappings $\overline{Z_i} = Z_{i+1}Z_n \mapsto Z_{i+1}Z_n$ and $\overline{X_i} = X_1X_{i+1} \mapsto X_1X_{i+1}$, which are trivial, indicating that all logical Pauli operators remain unchanged. It is easily checked that the stabilizers are also preserved. Therefore, by implementing the logical identity at the beginning of the physical circuit in Fig. 7, we can cancel the $\text{CZ}_{i+1,n}$ gates for all $i \notin I_h$. We prove that the physical circuit incorporating these cancellations has depth scaling only *linearly with* k.

Theorem 9. *Consider the physical circuit constructed by the solve-and-stitch algorithm in Theorem 7 to realize a logical C-QSK circuit with h Hadamards on the $[\![n, n-2, 2]\!]$ codes. After integrating the logical identity, the depth is at most*

$$\begin{cases} (2+2h)k - h^2 - h + 6 & \textit{for even } h, \\ (2+2h)k - h^2 + h + 7 & \textit{for odd } h. \end{cases} \tag{10}$$

In general, these physical circuits are not inherently fault-tolerant, as a single fault from a two-qubit gate can propagate to other qubits. However, by applying *flag gadgets* [16] to the physical circuits on the $[\![6, 4, 2]\!]$ code, we observe that two additional flag qubits make the circuit fault-tolerant, i.e., they enable the detection of correlated errors arising from a single fault in the circuit (see [17] for details). A similar approach could be extended to the full $[\![n, n-2, 2]\!]$ family.

V. Conclusion and Future Work

In this work, we were motivated by the fact that quantum computers are expected to show major advantages over classical computers only in specific algorithms and applications. Hence, we began to explore a new path where compilers for universal fault-tolerant quantum computers tailor the codes to execute a given target algorithm efficiently and fault-tolerantly. As an illustrative algorithm, we considered Trotter circuits for quantum simulation, which is a key application of quantum computers. We developed a systematic solve-and-stitch approach to synthesize optimal-depth realizations of Clifford Trotter circuits on the well-known $[\![n, n-2, 2]\!]$ code family. We analyzed the approach rigorously and described the steps in Fig. 7. We ensured fault-tolerance by embedding flag gadgets in the synthesized physical circuits.

Obviously, this code family can only detect errors but not correct them. In future work, we will explore the generalization of the approach to better code families. We will leverage the insights developed in Theorem 7 to understand the code properties necessary to realize Trotter circuits efficiently and fault-tolerantly. This will bring us closer to truly tailoring code design for this application.

Acknowledgments

The work of N. R. was partially supported by the National Science Foundation under Grant no. 2106189.

References

[1] L. Postler, F. Butt, I. Pogorelov, C. D. Marciniak, S. Heußen, R. Blatt, P. Schindler, M. Rispler, M. Müller, and T. Monz, "Demonstration of fault-tolerant Steane quantum error correction," *arXiv preprint arXiv:2312.09745*, 2023. [Online]. Available: https://arxiv.org/abs/2312.09745

[2] D. Bluvstein, S. J. Evered, A. A. Geim, S. H. Li, H. Zhou, T. Manovitz, S. Ebadi, M. Cain, M. Kalinowski, D. Hangleiter *et al.*, "Logical quantum processor based on reconfigurable atom arrays," *Nature*, pp. 1–3, 2023.

[3] M. da Silva, C. Ryan-Anderson, J. Bello-Rivas, A. Chernoguzov, J. Dreiling, C. Foltz, J. Gaebler, T. Gatterman, D. Hayes, N. Hewitt *et al.*, "Demonstration of logical qubits and repeated error correction with better-than-physical error rates," *arXiv preprint arXiv:2404.02280*, 2024. [Online]. Available: https://arxiv.org/abs/2404.02280

[4] H. Ali, J. Marques, O. Crawford, J. Majaniemi, M. Serra-Peralta, D. Byfield, B. Varbanov, B. M. Terhal, L. DiCarlo, and E. T. Campbell, "Reducing the error rate of a superconducting logical qubit using analog readout information," *arXiv preprint arXiv:2403.00706*, 2024. [Online]. Available: https://arxiv.org/abs/2403.00706

[5] L. Z. Cohen, I. H. Kim, S. D. Bartlett, and B. J. Brown, "Low-overhead fault-tolerant quantum computing using long-range connectivity," *Science Advances*, vol. 8, no. 20, p. eabn1717, 2022. [Online]. Available: https://arxiv.org/abs/2110.10794

[6] Y.-F. Wang, Y. Wang, Y.-A. Chen, W. Zhang, T. Zhang, J. Hu, W. Chen, Y. Gu, and Z.-W. Liu, "Efficient fault-tolerant implementations of non-Clifford gates with reconfigurable atom arrays," *arXiv preprint arXiv:2312.09111*, 2023. [Online]. Available: https://arxiv.org/abs/2312.09111

[7] G. Zhu, S. Sikander, E. Portnoy, A. W. Cross, and B. J. Brown, "Non-Clifford and parallelizable fault-tolerant logical gates on constant and almost-constant rate homological quantum LDPC codes via higher symmetries," *arXiv preprint arXiv:2310.16982*, 2023. [Online]. Available: https://arxiv.org/abs/2310.16982

[8] A. Montanaro, "Quantum algorithms: an overview," *npj Quantum Information*, vol. 2, no. 1, pp. 1–8, 2016. [Online]. Available: https://arxiv.org/abs/1511.04206

[9] R. Au-Yeung, B. Camino, O. Rathore, and V. Kendon, "Quantum algorithms for scientific applications," *arXiv preprint arXiv:2312.14904*, 2023. [Online]. Available: https://arxiv.org/abs/2312.14904

[10] A. M. Dalzell, S. McArdle, M. Berta, P. Bienias, C.-F. Chen, A. Gilyén, C. T. Hann, M. J. Kastoryano, E. T. Khabiboulline, A. Kubica *et al.*, "Quantum algorithms: A survey of applications and end-to-end complexities," *arXiv preprint arXiv:2310.03011*, 2023. [Online]. Available: https://arxiv.org/abs/2310.03011

[11] J. D. Whitfield, J. Biamonte, and A. Aspuru-Guzik, "Simulation of electronic structure Hamiltonians using quantum computers," *Molecular Physics*, vol. 109, no. 5, pp. 735–750, 2011. [Online]. Available: https://arxiv.org/abs/1001.3855

[12] A. J. Daley, I. Bloch, C. Kokail, S. Flannigan, N. Pearson, M. Troyer, and P. Zoller, "Practical quantum advantage in quantum simulation," *Nature*, vol. 607, no. 7920, pp. 667–676, 2022.

[13] G. Li, A. Wu, Y. Shi, A. Javadi-Abhari, Y. Ding, and Y. Xie, "Paulihedral: a generalized block-wise compiler optimization framework for quantum simulation kernels," in *Proceedings of the 27th ACM International Conference on Architectural Support for Programming Languages and Operating Systems*, 2022, pp. 554–569. [Online]. Available: https://arxiv.org/abs/2109.03371

[14] J. Dehaene and B. De Moor, "Clifford group, stabilizer states, and linear and quadratic operations over GF(2)," *Physical Review A*, vol. 68, no. 4, p. 042318, 2003. [Online]. Available: https://arxiv.org/abs/quant-ph/0304125

[15] D. Gottesman, "Stabilizer codes and quantum error correction," Ph.D. dissertation, 1997. [Online]. Available: https://arxiv.org/abs/quant-ph/9705052

[16] R. Chao and B. W. Reichardt, "Fault-tolerant quantum computation with few qubits," *NPJ Quantum Information*, vol. 4, no. 1, p. 42, 2018.

[17] Z. Chen and N. Rengaswamy, "Tailoring fault-tolerance to quantum algorithms," *arXiv preprint arXiv:2404.11953*, 2024. [Online]. Available: https://arxiv.org/abs/2404.11953

[18] B. Eastin and E. Knill, "Restrictions on transversal encoded quantum gate sets," *Physical review letters*, vol. 102, no. 11, p. 110502, 2009. [Online]. Available: https://arxiv.org/abs/0811.4262

[19] N. Rengaswamy, R. Calderbank, S. Kadhe, and H. D. Pfister, "Logical Clifford synthesis for stabilizer codes," *IEEE Transactions on Quantum Engineering*, vol. 1, pp. 1–17, 2020.