Lattice surgery-based logical operations in a fault-tolerant quantum software framework

Youngchul Kim
Future Computing Research Division
ETRI
Daejeon, Republic of Korea
kimyc@etri.re.kr

Soo-Cheol Oh
Future Computing Research Division
ETRI
Daejeon, Republic of Korea
ponylife@etri.re.kr

Sangmin Lee
Future Computing Research Division
ETRI
Daejeon, Republic of Korea
sanglee@etri.re.kr

Ki-Sung Jin
Future Computing Research Division
ETRI
Daejeon, Republic of Korea
ksjin@etri.re.kr

Gyuil Cha
Future Computing Research Division
ETRI
Daejeon, Republic of Korea
gicha@etri.re.kr

***Abstract*—We present a quantum software framework that supports fault-tolerant quantum computing. This framework uses the lattice surgery technique to encode logical qubits in surface codes and implements logical Clifford and T gates. By interfacing the QPlayer simulator with the framework, we have configured six two-dimensional logical qubits with a distance of three and have evaluated the lattice surgery-based logical operations, which have been presented theoretically by simulating quantum circuits composed of universal quantum gates. In conclusion, we have shown that the proposed framework can effectively perform Clifford and T gates in fault-tolerant quantum computing based on surface code logical qubits.**

***Index Terms*—fault-tolerant quantum computing, quantum software framework, surface code, lattice surgery, quantum simulator**

I. INTRODUCTION

Developing quantum computers from current noisy quantum devices requires fault tolerance using quantum error correction(QEC) [1]. There is much research on QEC to support fault tolerance with topological codes, especially surface codes [2], [4]. Surface code is considered the most prominent QEC method due to its high error threshold (up to 1%) [2] and simple two-dimensional(2D) structure with only nearest-neighbor(NN) interactions. However, encoding logical qubits with surface codes requires many physical qubits, and it is necessary to scale to larger surface codes to suppress the error rate of logical qubits. Surface code with a distance of three can correct single-qubit or, at most, two-qubit errors with the smallest number of qubits. Implementing and performing quantum circuits with surface code logical qubits fault-tolerantly, surface code needs to be encoded with a distance of at least 3.

Surface code performs logical operations by interacting between locally adjacent lattices in a 2D structure. Various techniques, such as transversal gates and teleported gates [3], have been studied to perform logical operations, but they are costly and complex. The lattice surgery(LS) [4], [5] method can alleviate these problems.

In practice, we need to provide a quantum computer that supports universal quantum gates to reap the benefits of quantum computing. To do this end, we have to support Clifford and non-Clifford gates. Clifford gates, generated by $CNOT$, H, and S gates, can be effectively simulated on a classical computer [6] and implemented via the LS. However, it is not easy to implement non-Clifford gates, typically T gate, which requires magic state distillation [7] that needs many resources and time. Therefore, in previous studies, LS-based logical Clifford+T operations have been theoretically analyzed [4] or experimentally implemented with small surface codes [5].

We have implemented a quantum software framework to support a fault-tolerant universal quantum computer. This framework provides LS-based logical Clifford+T gates. This paper uses QPlayer [8], [9] to simulate quantum circuits composed of Clifford+T gates and verify the computational results. For this purpose, the quantum circuits are translated into LS-based logical operations and converted into physical operations to perform on the simulator.

II. FAULT-TOLERANT QUANTUM SOFTWARE FRAMEWORK

The fault-tolerant quantum software framework has been implemented in a layered architecture [10]. Qubits are accessed at logical, virtual, and physical levels at each layer. A quantum program is written with a quantum programming language, and a quantum compiler translates it into LS-based logical operations defined in Table I, which are processed in this framework. The fault-tolerant software framework is outlined in Fig. 1, and the features of each layer are as follows:

Execution layer performs logical qubit mapping and logical operation translation through the Fault-tolerant layer. After executing the translated logical operations, it does post-logical operations according to the qubit measurement outcomes or

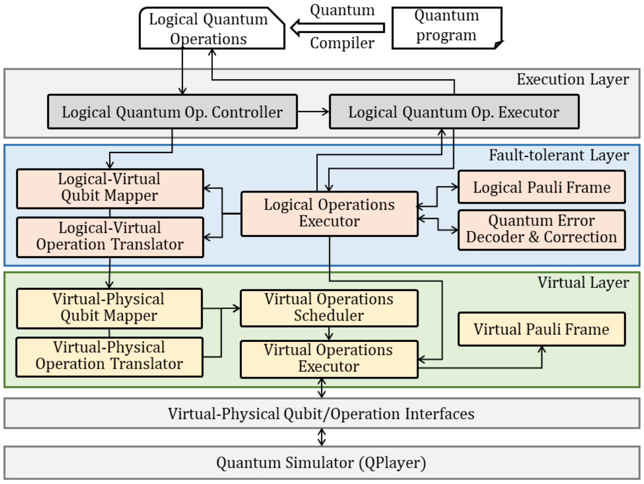


Fig. 1. Fault-tolerant quantum software framework.

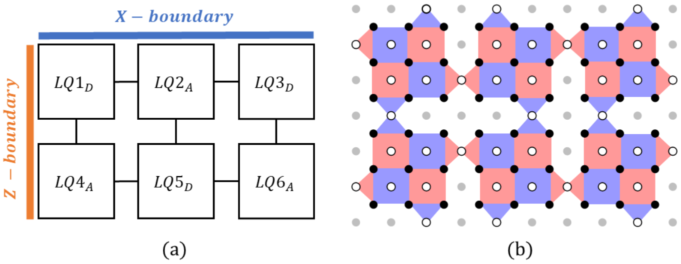
returns the qubit measurement outcomes to the quantum program.

Fault-tolerant layer encodes logical qubits in rotated surface code using virtual qubits and arranges a 2D logical qubit architecture into a checkerboard form, as shown in Fig. 2. It maps logical qubits to virtual qubits and converts logical operations to virtual operations. A logical qubit is made up of virtual qubits, and a logical operation is composed of operations on the virtual qubits that make up the logical qubit. It performs logical operations after mapping the qubits and converting the operations. Each time it does, it performs error syndrome measurement(ESM) for the logical qubit. After ESM execution, error detection and correction are performed according to the measurement outcomes. Among the logical operations, Pauli operations are handled in software through the logical Pauli frame [11].

Virtual layer maps virtual qubits to physical qubits and converts virtual operations to the corresponding physical operations. It can perform virtual operations in parallel and schedule according to the physical operation properties provided by the quantum simulator. In this paper, physical qubits and operations are emulated by the QPlayer simulator. The QPlayer simulator processes the virtual operations through the Virtual-Physical qubit/gate interface and returns the execution results. Within the virtual operations, Pauli operations are handled in software through the physical Pauli frame [11].

TABLE I
LS-BASED LOGICAL OPERATIONS SUPPORTED BY THE FRAMEWORK.

Operation types	**Operations**
Init. & Pauli	Init_X(Z), Pauli_X(Z)
Measurement	Measure_X(Z)
Lattice Surgery	Merge_Mxx(Mzz), Split_Mxx(Mzz) CNOT_Post_Mxx(Mzz) Move_Post_Mxx(Mzz) Hadamard, Deform Flip_Expand(Contract, Shift) Inject_Y(A), S(T, T_Dag)_Post



(a) (b)

Fig. 2. Logical qubit architecture. (a) shows the logical connectivity of the logical qubits. $LQ1_D$, $LQ3_D$, and $LQ5_D$ indicate logical data qubits. $LQ2_A$, $LQ4_A$, and $LQ6_A$ indicate logical ancilla qubits. (b) shows rotated surface codes constructed from physical qubits. Physical data qubits are represented by black circles, and ancilla qubits by white circles. The purple(pink) plaquettes represent the X(Z) stabilizers of the form $X^{\otimes 4}(Z^{\otimes 4})$ and $X^{\otimes 2}(Z^{\otimes 2})$, respectively.

Quantum simulator supports simulations of physical qubit operations. It provides the properties of physical operations and the connectivity between physical qubits. A physical two-qubit operation can only be performed between NN qubits connected to each other. QPlayer is a quantum simulator that provides more qubits and faster quantum operations with smaller memory. It selectively tracks realized quantum states using a reduced quantum state representation scheme instead of loading the entire quantum states into memory.

III. LS-BASED LOGICAL OPERATIONS

The LS is a fault-tolerant protocol that can perform state teleportation or gate teleportation between logical qubits encoded in a surface code. It is performed in two steps: merging and splitting. Merging and splitting perform the logical joint measurements along the X(Z)-boundaries, $M_{ZZ}(M_{XX})$, on which they operate. Fig. 3 shows circuits of logical operations using LS, such as state teleportation, $CNOT$, S, and T. In particular, the logical S and T operations require the magic state that can be prepared with the state injection process. However, since state injection is not fault-tolerant, the injected magic state has low fidelity and needs to be distilled. Magic state distillation procedures are not easy to implement because it costs a lot of resources and time to obtain a higher-fidelity magic state from multiple lower-fidelity states [2], [12]. In this work, we prepare the magic state through the injection process without the magic state distillation and assume it has high fidelity.

We have implemented the Clifford+T gates, such as H, $CNOT$, S/$S^{\dagger}$, and T/$T^{\dagger}$ using LS in the rotated surface code. The logical $CNOT$ operation performs a logical joint measurement, $M_{ZZ}(M_{XX})$, along the boundary with the adjacent ancilla qubit according to the logical connectivity of the control and target qubits in the logical qubit architecture. It then performs Pauli corrections on the measurement outcomes. The logical S/$S^{\dagger}$ operation performs a logical joint measurement on the X-boundary, M_{ZZ}, with the neighboring ancilla qubit injected magic state $|Y\rangle = \frac{1}{\sqrt{2}}(|0\rangle + i\,|1\rangle)$. It then performs a Pauli correction based on the joint measurement outcome and the ancilla qubit measurement outcome. The

TABLE II
EVALUATION OF QUANTUM CIRCUITS COMPOSED OF CLIFFORD+T GATES.

Original circuits							**LS-based circuits**					
Circuit	Qubits	X	H	CX	S/S†	T/T†	Qubits$_{L_D}$	Qubits$_{L_A}$	Qubits$_P$	Op.$_L$	Op.$_V$	Op.$_P$
deutsch_n2	2	1	3	1	0	0	2	2	70	26	1894	1608
grover_n2	2	4	10	2	0	0	2	2	70	70	5245	4416
iswap_n2	2	1	4	2	2	0	2	2	70	45	3465	2972
teleportation_n3	3	0	4	2	1	1	3	3	106	50	3817	3307
toffoli_n3	3	0	2	6	1	7	3	3	106	128	10209	8985

Pauli correction operation is processed in software using the Pauli frame. The logical $T/T^\dagger$ operation requires the magic state $|A\rangle = \frac{1}{\sqrt{2}}(|0\rangle + e^{i\frac{\pi}{4}}|1\rangle)$. It performs a logical joint measurement on the X-boundary, M_{ZZ}, with the neighboring ancilla qubit injected magic state. Then, depending on the joint measurement outcome and the ancilla qubit measurement outcome, Clifford correction, $S/S^\dagger$ operation, is applied, or Pauli correction is performed in software using the Pauli frame. In addition, the $SWAP$ gate, which three consecutive $CNOT$ gates can implement, can be performed using LS-based logical state teleportations.

We have simulated some quantum circuits in QASMBench [13] and identified the expected results by changing the input states in the quantum circuits. Table II shows the count of qubits and gates that make up the benchmark quantum circuits. The table also shows the count of logical data and ancilla qubits when these circuits are translated to LS-based circuits. The physical qubit counts include the physical qubits encoding the distance-3 rotated surface code and the syndrome physical qubits of the stabilizers newly added between the two logical qubits in the merging operation. The operation counts indicate the count of logical, virtual, and physical operations as LS-based circuits are transformed and executed through the layers in our framework. Due to the Pauli frames, the physical operation counts performed in the simulator are less than the virtual operation counts.

The T gate is a typical non-Clifford gate with significant overhead to implement fault-tolerant. For example, Fig. 4 shows a $Toffoli$ gate decomposed using H, $CNOT$, T, $T^\dagger$, and S gates. The decomposed circuit of a $Toffoli$ gate contains nine Clifford gates and seven non-Clifford gates. Therefore, it requires multiple LS operations and Clifford corrections based on the joint measurement outcomes and ancilla qubit measurement outcomes. We have performed simulations of the LS-based $Toffoli$ operation with eight input states, $|000\rangle$, $|001\rangle$, …, $|111\rangle$, and identified that the measurement outcomes are the same as the truth table of $Toffoli$ gate.

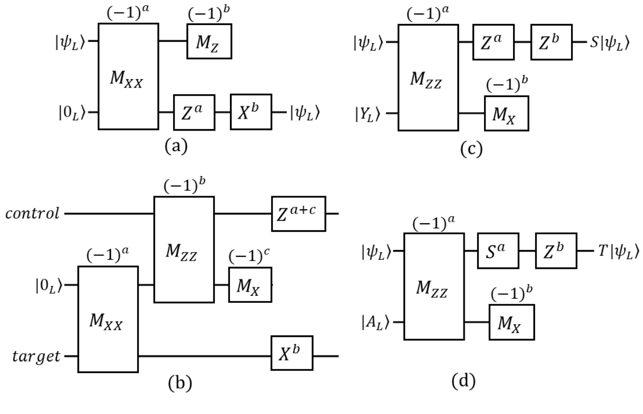


Fig. 3. LS-based logical operation circuits. (a) Logical state teleportation. (b) Logical $CNOT$ operation. (c) Logical S operation. (d) Logical T operation.

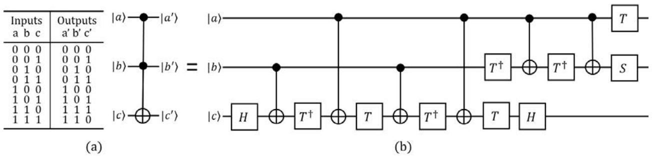


Fig. 4. (a) Truth table and quantum circuit of $Toffoli$ gate. (b) Decomposition of a $Toffoli$ gate using Clifford+T gates.

IV. CONCLUSION

We have presented a quantum software framework for fault-tolerant universal quantum computers. It implements LS-based logical Clifford+T gates using rotated surface code on logical qubit architecture. We have simulated quantum circuits with Clifford+T gates in the framework and evaluated LS-based logical operations. The quantum software framework can be extended to architectures arranged with more logical qubits, and the LS-based logical operations implemented in this work can serve as a primary reference model. As a next step, we will apply various quantum error models to verify the LS-based logical operations.

ACKNOWLEDGMENT

This work was supported by the Institute of Information & Communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (No.2020-0-00014, A Technology Development of Quantum OS for Fault-tolerant Logical Qubit Computing Environment).

REFERENCES

[1] M. A. Nielsen and I. L. Chuang, "Quantum Computation and Quantum Information, Cambridge University Press, 2010.

[2] A. G. Fowler, M. Mariantoni, J. M. Martinis, and A. N. Cleland, "Surface codes: Towards practical large-scale quantum computation," Physical Review A. 86, 2012.

[3] D. Gottesman and I. L. Chuang, "Demonstrating the viability of universal quantum computation using teleportation and single-qubit operations," Nature. 402, 1999.

[4] C. Horsman, A. G. Fowler, S. Devitt, and R. V. Meter, "Surface code quantum computing by lattice surgery," New Journal of Physics. 14, 2012.

[5] A. Erhard, H. P. Nautrup, M. Meth, L. Postler, R. Stricker, M. Stadler, V. Negnevitsky, M. Ringbauer, P. Schindler, H. J. Briegel, R. Blatt, N. Friis and T. Monz, “Entangling logical qubits with lattice surgery,” Nature. 589, 2021.

[6] D. Gottesman, “The Heisenberg Representation of Quantum computers,” arXiv:quant-ph/9807006, 1998.

[7] S. Bravyi and A. Kitaev, “Universal quantum computation with ideal Clifford gates and noisy ancillas,” Physical Review A. 71, 2005.

[8] K. S. Jin and G. I. Cha, “QPlayer: Lightweight, scalable, and fast quantum simulator,” ETRI Journal. 45, (2023), 304–317.

[9] K. S. Jin and G. I. Cha, “Multilayered logical qubits and synthesized quantum bits,” Quantum Science and Technology. 8 035008, 2023.

[10] N. C. Jones, R. V. Meter, A. G. Fowler, P. L. McMahon, J. S. Kim, T. D. Ladd, and Y. Yamamoto, “Layered Architecture for Quantum Computing,” Physical Review X. 2, 2012.

[11] J. H. On, C. Y. Kim, S. C. Oh, S. M. Lee, and G. I. Cha, “A multilayered Pauli tracking architecture for lattice surgery-based logical qubits,” ETRI Journal. (2022), 1–17, doi:10.4218/etrij.2022-0037.

[12] L. Lao and B. Criger, “Magic state injection on the rotated surface code,” ACM Int’l Conf. on Computing Frontiers, 2022.

[13] A. Li, S. Stein, S. Krishnamoorthy, and J. Ang, “QASMBench: A Low-Level Quantum Benchmark Suite for NISQ Evaluation and Simulation,” ACM Trans. on Quantum Computing, 2022.