Implementing Quantum Polar Codes in a Superconducting Processor

From State Preparation to Decoding

Handy Kurniawan, Universidad Complutense de Madrid

Valentin Savin, Université Grenoble Alpes, CEA-LETI

Carmen G. Almudéver, Universitat Politècnica de València

Francisco García Herrero, Universidad Complutense de Madrid

// A full-stack software framework for implementing quantum polar codes is presented, integrating a noise-aware compilation approach that optimizes resource usage by combining quantum and classical software. Experimental results demonstrate significant improvements in logical state preparation rates, resulting in reduced comsumption of both quantum and classical resources. //

Digital Object Identifier 10.1109/MS.2025.3569447

ACHIEVING LARGE-SCALE FAULT-TOLERANT quantum computation relies heavily on implementing quantum error correction (QEC), where the quantum information is protected from noise by encoding it using a quantum error-correcting code (QECC).[1] Most QEC proposals detect and correct errors by measuring the stabilizers of the encoded quantum state. Ideally, stabilizer measurements provide perfect syndrome information, accurately indicating the presence of errors.[2] However, on real quantum devices, noise in syndrome measurements reduces computational reliability and increases the complexity of classical decoding required to identify errors.

Several studies have explored the application of various QECCs on real quantum hardware, such as the surface code,[3] the repetition code,[4] the Steane code,[5] cluster-state-based codes,[6] and quantum low-density parity-check (LDPC) codes.[7] These codes have been implemented in a wide range of platforms that include trapped-ion,[5] photonic,[6] neutral atom,[7] and superconducting[3,4] quantum processors.

Quantum polar codes[8] are a class of QECCs with unequaled error correction performance, as they are known to achieve the symmetric coherent information (one-shot capacity) of any quantum channel while being equipped with an efficient, log-linear complexity decoding algorithm for Pauli channels. Recently, a fault-tolerant preparation procedure of polar code states using two-qubit (2Q) Pauli measurements and error detection has been proposed[9,10] which, coupled with Steane error correction, was shown to outperform by several orders of magnitude the error rate performance of a surface code with similar

parameters.[10] Yet, the fault-tolerant implementation of quantum polar codes requires interactions between distant qubits, which is challenging for quantum processors that only allow nearest-neighbor interactions. In this case, it is necessary to insert swap gates (SWAP) gates that bring nonadjacent qubits to physically adjacent positions whenever they need to perform a 2Q gate. This results in an overhead of 2Q gates, which increases the noise level. Simply reducing the 2Q gate overhead might not be the optimal solution, since a higher count of 2Q gates with lower error rates may improve the overall fidelity.[11]

This article proposes a full-stack approach to implementing quantum polar codes, motivated by their strong performance and efficient decoding, with a focus on addressing the challenge of local interaction constraints on quantum hardware. The key contributions of this work are as follows:

1. demonstrating the effectiveness of noise-aware compilation techniques on real superconducting quantum processors with limited connectivity, using quantum polar codes as a challenging case study that underscores the importance of these techniques
2. implementing a platform that spans the entire process, from the compilation and optimization of quantum circuits for state preparation to the classical software system responsible for processing syndrome measurements with error detection and correction algorithms
3. experimentally demonstrating that our proposal results in significant time and space resource savings, without degrading the logical error rate (LER) performance.

Background

This section provides an overview of the relevant key concepts, including a short description of quantum polar codes, the main types of noise models considered for simulation, and a review of the existing noise-aware compilation techniques.

Quantum Polar Codes

Quantum polar codes are a class of QECCs with high-weight stabilizers, which prevent the implementation of fault-tolerant error correction through repeated syndrome measurements, similar to what is done for topological, or more generally, quantum LDPC codes. The preparation procedure[9] progressively constructs the generators of the stabilizer group using 2Q Pauli measurements. As the measured operators anticommute, the preparation procedure cannot be repeated. Instead, it can be complemented by a specific error detection mechanism, making it fault-tolerant. This allows fault-tolerant preparation of polar code states, which can be coupled with Steane error correction to implement fault-tolerant error correction. Figure 1 illustrates a high-level block diagram of the QEC layer based on polar codes. This hybrid process integrates quantum operations, such as state preparation, with classical operations to implement error detection and correction algorithms. Note that implementing the control unit is out of the scope of this work.

State Preparation. We consider polar codes of length $N = 2^n$ data qubits ($n > 0$), encoding one logical qubit, denoted as Q_1 codes.[9] The state preparation process is done in n steps, where each step consists of either $N/2$ Pauli $Z \otimes Z$ measurements, or $N/2$ Pauli $X \otimes X$ measurements. The total number of physical qubits is $\bar{N} := N + N/2$, comprising the N data qubits, storing the encoded quantum information, and $N/2$ ancilla qubits used to implement the measurements. The specific pattern of the Pauli measurements depends on the information index in the polar code[9] and the prepared logical state. Figure 2 illustrates the fault-tolerant preparation protocol for a Q_1 code of length $N = 8$.

Error Detection. In the noiseless case, at each step of the preparation procedure, the $N/2$ classical bits obtained from the $N/2$ Pauli measurements can be organized as a concatenation of codewords of some smaller classical polar codes. One can exploit this property to detect errors by computing the corresponding error syndromes (not to be confused with the quantum polar code syndrome). This is illustrated in Figure 3, where error syndromes are computed for each block of $N/2$ bits. A nonzero syndrome indicates the presence of an error, in which case the prepared state is discarded and the preparation process is repeated. If all of the syndromes are zero, the prepared state is accepted. This makes the state preparation probabilistic, and we refer to the probability that the state is successfully prepared as preparation rate. Preparation failures can be addressed through time redundancy, where the state preparation is repeated until it succeeds, or space redundancy through a factory-based approach,[10] where multiple code states are prepared in parallel to

increase the likelihood of a valid state. Note that a successfully prepared state is still prone to errors, as some errors may go undetected by the error detection mechanism. Finally, within the error detection unit, the signs of the stabilizer group generators and logical operators (for the prepared logical state) are also computed and transmitted to the error correction unit, in case of successful state preparation (N bits, for $N-1$ generators and one logical operator).

Error Correction. In Steane error correction, the prepared state serves as an ancilla state, onto which the errors from the protected state are copied through transversal controlled-NOT (CNOT) gates and measured out to perform error correction. Implementing the full Steane error correction procedure is out of the scope of this work, but we are still interested in determining the LER of the prepared logical state. To do so, we directly measure the prepared state and perform error correction. Precisely, we measure the data qubits on either X or Z basis, depending on the prepared logical state, and perform successive cancelation (SC) decoding on the measurement outputs to determine the sign of the logical operator. Note that SC decoding only uses the information on the signs of the $N-1$ generators received from the error detection unit. A logical error occurs if the decoded sign of the logical operator does not match the one determined within the error detection unit.

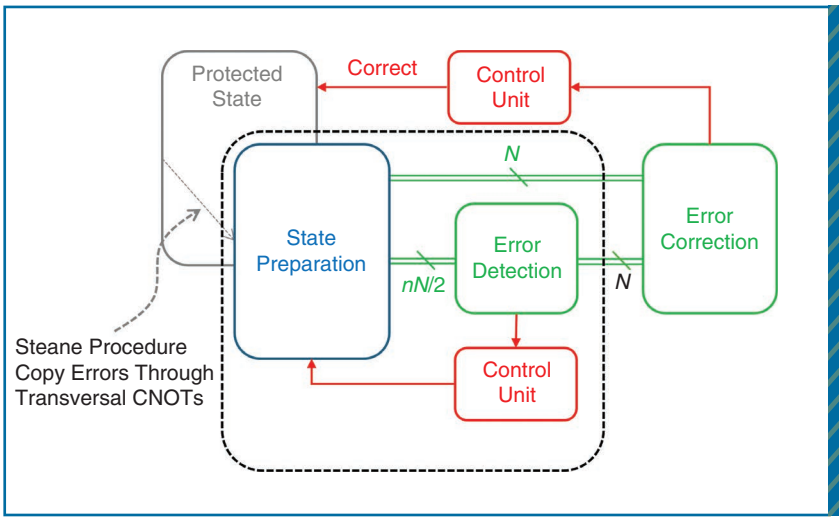

FIGURE 1. Block diagram of QEC layer based on polar codes. The blue color represents the quantum software, the green color represents the classical software, and the red and gray colors are out of the scope of this work.

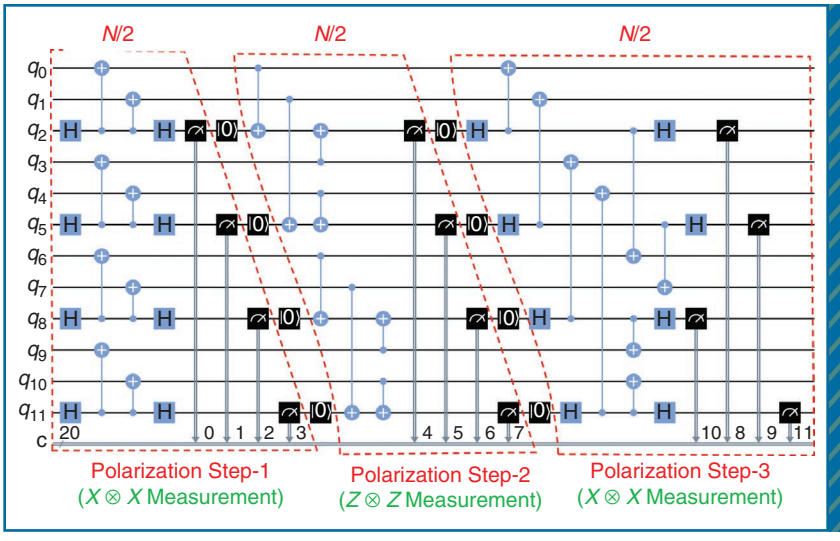

FIGURE 2. Measurement-based preparation of the logical $|+\rangle$ state for the $Q_1(N=8, i=3)$ code,[9] where i indicates the information position in the polar code. Red dashed lines indicate the three polarization steps in the preparation procedure. Steps one and three consist of Pauli $X \otimes X$ measurements, while step two consists of Pauli $Z \otimes Z$ measurements.

Noise Models

To simulate the behavior of quantum computers, we employ the calibration-based noise model, which provides a more accurate representation but is resource-intensive. This model is constrained by the native gates and topology of the actual hardware, mimicking the circuit executed on real devices.[12] As shown in Figure 4, additional gates (compared to the original circuit in Figure 2) are needed due to hardware limitations, which significantly increases the circuit depth and noise. This model also accounts for error sources such as qubit parameters (relaxation and dephasing times, frequency,

anharmonicity, and readout errors), and the specific error rates of native gates provided by hardware vendors.[12] More considered noise models are presented in the supplementary downloadable material available at https://doi.org/10.1109/MS.2025.3569447, provided by the authors.

Noise-Aware Compilation

Noise-aware compilation techniques employ error rate data from quantum device calibrations to improve circuit fidelity. These techniques are useful in quantum processors with limited connectivity, where long-distance qubit interactions require numerous SWAPs, increasing the number of 2Q gates that lead to higher error accumulation and degrade circuit fidelity. Noise-aware compilation optimizes qubit placement and routing by selecting qubits and gates with lower error rates to reduce errors.

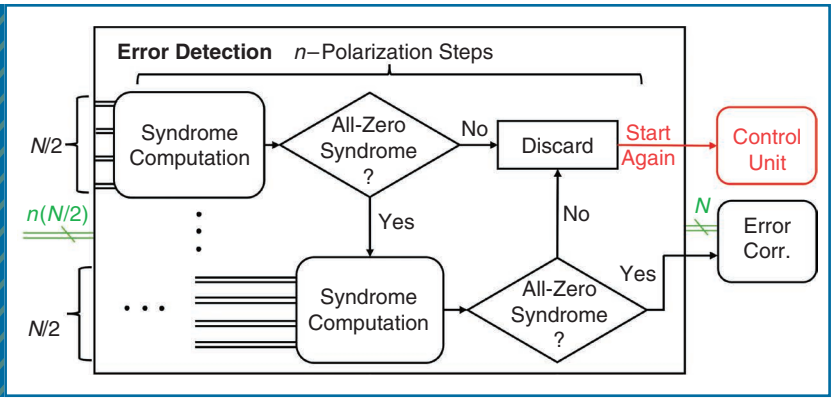

FIGURE 3. Error detection unit processing $n(N/2)$ classical bits. Error syndromes are computed and verified. If an error is detected (i.e., the syndrome is not all-zero), the prepared state is discarded, and the process restarts. If no error is detected, the information on the sign of the stabilizers and logical operators is passed to the error correction unit.

Proposed Architecture

Figure 5 illustrates the proposed architecture, which consists of quantum software, classical software, and cloud infrastructure. See the supplementary downloadable material available at https://doi.org/10.1109/MS.2025.3569447, provided by the authors for implementation details.

The quantum software implements the state preparation phase through a noise-aware compiler. The compiler combines SABRE's initial qubit placement strategy[13] with TriQ's noise-aware qubit routing approach[14] (S-TriQ). This hybrid

FIGURE 4. Measurement-based preparation of the logical $|+\rangle$ state for the $Q_1(N=8, i=3)$ code from Figure 2, mapped into **ibm_sherbrooke** topology using noise-aware compilation technique (top). The original CNOT gate was mapped using the native ECR gate and single-qubit prerotations. Partial detail (bottom).

method allows scalable and efficient compilation, even for larger code lengths N, as analyzed in the next section. The compiler's input includes the target device's topology and the calibration data. Once compiled, the resulting quantum circuits are executed on real quantum hardware and on a noisy simulator. For simulation, Qiskit Aer simulator[12] is used, incorporating the calibration-based noise model for higher accuracy in the analysis. Additionally, this simulator allows us to scale down the noise model to explore how the LER will change with future improvements in quantum hardware.

The classical software handles the postprocessing of the measured classical bits through error detection and correction. The error detection stage provides the preparation rate results, while the error correction and verification software enable us to compute the LER.

All the classical software was implemented using Python 3.11.5 on a virtual machine configured with 13 GB of memory and eight CPUs with model 12th Gen Intel(R) Core(TM) i7-1255U (4.70 GHz), running Ubuntu 20.04. The software stack included Qiskit 0.44.3, Qiskit-Aer 0.14.2, Qiskit-IBM-Runtime 0.24.0,[12] and the updated TriQ package.[14] The quantum back ends used in this study are the 127-qubit **ibm_brisbane** and **ibm_sherbrooke** devices with versions 1.1.40 and 1.5.38, respectively.

Results and Discussion

We evaluate the performance of our proposed noise-aware compilation technique (S-TriQ) against a standard workflow that uses nonnoise-aware compilation (Qiskit-3), which does not consider calibration data. Our comparison focuses on key metrics such as preparation rate, LER, compilation latency, and decoding latency.

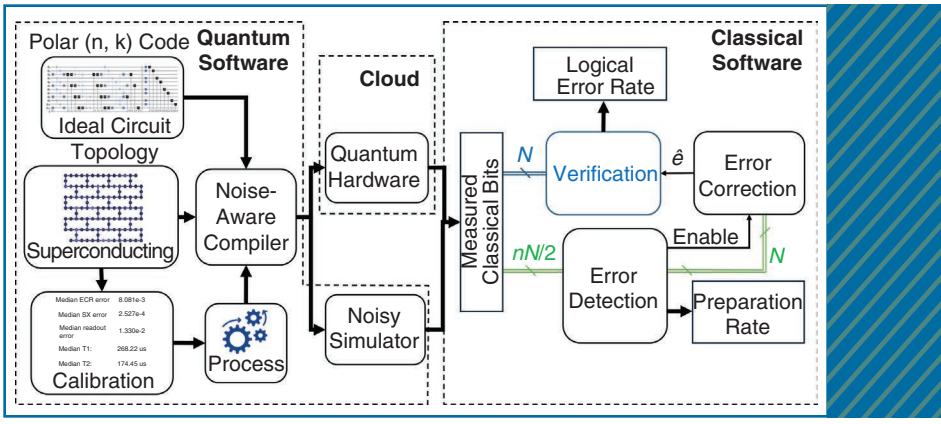

FIGURE 5. Block diagram of the proposed implementation.

The experiments were run over 20 days, with four runs per day on average. Measurements for code lengths $N = 4,\ 8,$ and 16, corresponding to $\bar{N} = 6,\ 12,$ and 24 physical qubits, respectively, were taken in batches of 20,000, 10,000, and 10. These values were chosen to consider both hardware and time replication. By using the factory method,[10] the $N = 16$ code can be replicated five times within the 127-qubit topology of devices like **ibm_brisbane** and **ibm_sherbrooke**.

As illustrated in Figure 6(a) and (b), using S-TriQ results in an average improvement in the preparation rate ranging from 17 to 36% for **ibm_brisbane**, and 25 to 55% for **ibm_sherbrooke**, compared to Qiskit-3. Notably, at an error scale of 0.2, the highest improvement is observed for both devices, indicating that the noise-aware compilation technique remains effective even as the physical error rate decreases. This enhancement in preparation rate can also be interpreted as resource savings in time, by reducing the number of repetitions needed for successful state preparation, or in space, by preparing circuits in parallel within a single execution, with the factory approach. The resource savings reach 10% to 54% for **ibm_brisbane**, and 20% to 81% for **ibm_sherbrooke**. The simulation of codes with $N \geq 32$ was not performed due to memory limitations in our local systems, as more than 100 GB of RAM was required. However, we executed them on real quantum computers, and the increased circuit depth and the elevated noise level made successful state preparation improvements negligible. With the current quantum devices, a length of code $N = 32$ achieved a preparation rate improvement of 0.06% and 1.7% for logical $|0\rangle$ and $|+\rangle$ respectively in **ibm_sherbrooke**. Nevertheless, it is reasonable to expect similar gains in the near future, when physical devices improve to the defined target physical error rate on the order of 10^{-4}.

Figure 6(c) and (d) compares the LERs for noise-aware and nonnoise-aware compilation techniques, showing almost identical results across different code lengths and noise levels. This suggests that the state preparation is functioning effectively in the proposed architecture, that is, the improvement in the preparation rate does not come at the expense of an increase in the number of undetected errors. Moreover, the observed

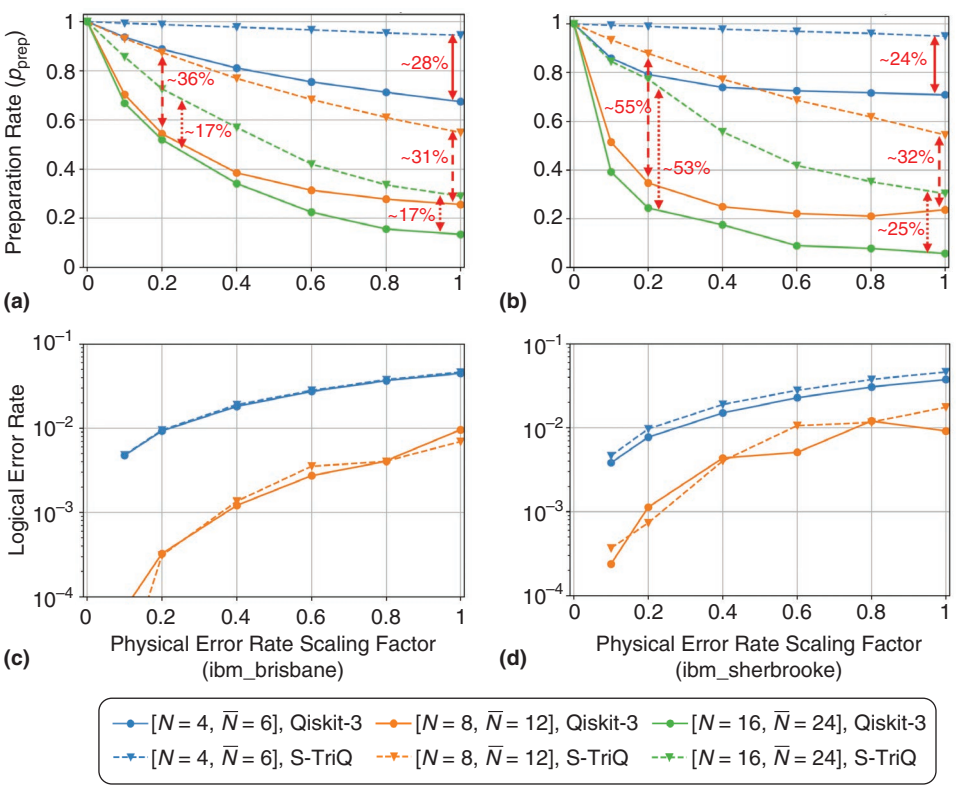

FIGURE 6. (a) and (b) show the preparation rate of the logical $|+\rangle$ state, while (c) and (d) depict the X logical error rate (for logical $|0\rangle$ state), for Q_1 codes of length $N = 4$ (blue), $N = 8$ (orange), and $N = 16$ (green) with quantum processors: (a) and (c) **ibm_brisbane**, and (b) and (d) **ibm_sherbrooke**. On the abscissa, the physical error scaling factor is the factor applied to the actual physical error rate of the device, as determined by the latest calibration data. Red arrows show the performance gap between noise-aware and nonnoise-aware compilation techniques.

LER is consistent with the minimum X-distance of the code ($d_X = 2$ for $N = 4$, and $d_X = 4$ for $N = 8$), decaying as $p^{d_X/2}$ for small physical error rates p, which also serves as evidence of the fault-tolerance property of the compiled circuits.

Regarding the proposed compiler, S-TriQ introduces nearly twice as many CNOT gates as Qiskit-3 but increases the preparation rate. This is due to the optimized qubit placement and routing in S-TriQ, which selects qubits and gates with lower error rates, thereby improving overall circuit fidelity. This tradeoff between gate count and preparation success highlights the effectiveness of noise-aware strategies, especially as code lengths increase. Additionally, the compilation time remains relatively stable, taking under 5 s for code lengths up to 32 qubits. However, at 64 qubits, the compilation time increases significantly, ranging from 45–66 s. This rise is due to the increased complexity of finding better routing paths with the amount of calibration data that needs to be processed for the derived quantum circuit. Conversely, the decoding time for one accepted state grows almost linearly with code length. Specifically, it was measured at around 0.3, 43, and 110 µs for a code length of 4, 8, and 16, respectively. The results are promising, as the decoding time remains within hundreds of microseconds, suggesting that real-time performance is achievable even with nonembedded system software. While 110 µs may seem like a large time compared to the latency requirements associated with QEC, given the current state of quantum technology, with a latency budget of tens of microseconds for decoding, we believe our approach demonstrates the feasibility of achieving low latencies even with a high-level

ABOUT THE AUTHORS

HANDY KURNIAWAN is a Ph.D. student in computer science at the Universidad Complutense de Madrid, 28040 Madrid, Spain. His current research interests include quantum compilers, quantum error mitigation, and quantum error correction. Kurniawan received his M.S. in computer science from the University of Tartu. Contact him at handykur@ucm.es.

CARMEN G. ALMUDÉVER is an associate professor at the Universitat Politècnica de València, 46022 Valencia, Spain. Her research interests include quantum compilers, quantum error correction, fault-tolerant quantum computation, mapping of quantum algorithms, and scalable (modular) quantum computing architectures. Almudéver received her Ph.D. in electronic engineering from Universitat Politècnica de Catalunya. Contact her at cargara2@disca.upv.es.

VALENTIN SAVIN is a senior researcher at the Institute for Electronics and Information Technologies (LETI), operated by the French Alternative Energies and Atomic Energy Commission (CEA), F-38054 Grenoble, France. His research interests include the field of classical and quantum error correction, for reliable communication and fault-tolerant information storage and processing. Savin received his Ph.D. in mathematics from the University of Grenoble. Contact him at valentin.savin@cea.fr.

FRANCISCO GARCÍA HERRERO is an associate professor at the Universidad Complutense de Madrid, 28040 Madrid, Spain. His research interests include classical and quantum error correction, fault-tolerant computing, and the design of VLSI architectures. Herrero received his Ph.D. in electronic engineering from the Universitat Politècnica de València. Contact him at francg18@ucm.es.

language (Python) implementation. Also, we expect that the time budget for decoding will expand as advancements are made, as discussed in the supplementary downloadable material available at https://doi.org/10.1109/MS.2025.3569447, provided by the authors. Additionally, with further optimizations, particularly through the use of compiled languages like C/C++, we anticipate a significant reduction in latency. For even faster decoding, hardware accelerators such as field-programmable gate array or application-specified integrated circuit (ASIC) implementations could be key to obtaining sub-microsecond implementations. As illustrated in Giard et al.,[15] an ASIC implementation can achieve a latency of 630 ns for a larger classical code length of 1024, substantially longer than the codes targeted in our work.

Conclusion

We presented a full-stack approach based on quantum and classical software that integrates noise-aware compilation with the polar code QEC layer, addressing challenges from state preparation to decoding on superconducting processors with constrained connectivity. We evaluated its performance using a simulator with a calibration-based noise model and real quantum devices, comparing two compilation techniques for quantum polar codes and demonstrating the clear advantages of the noise-aware approach. We further highlighted that as physical error rates improve, such as through error scaling, the performance of our proposed architecture also improves. Although the proposed method offers a path forward, achieving efficient fault-tolerant operation at scale remains a significant challenge,

particularly due to the current limitation of superconducting qubit technology, which restricts the code length to $N = 32$. As a result, future work may explore extending scalability through a factory-based approach, implementing the full Steane error correction procedure, and applying the method to processors with increased connectivity as they become available.

Code Availability

https://github.com/HandyKurniawan/na_polar_codes_framework

Acknowledgment

This work was supported by the QuantERA project EQUIP (Grants PCI2022-133004 and PCI2022-132922, funded by the AEI, Ministerio de Ciencia e Innovación, Gobierno de España, MCIN/AEI/10.13039/501100011033, and ANR-22-QUA2-0005-01, funded by the Agence Nationale de la Recherche, France), and by the European Union "NextGenerationEU/PRTR". This research is part of the project PID2023-147059OB-I00 funded by MCIU/AEI/10.13039/501100011033/FEDER, UE. Handy Kurniawan acknowledges support from the Comunidad de Madrid under Grant PIPF-2023/COM-30051. Carmen G. Almudéver acknowledges support from the Spanish Ministry of Science, Innovation, and Universities through the Beatriz Galindo program 2020 (BG20-00023) and the European ERDF PID2021-123627OB-C51. We acknowledge the use of IBM Quantum services for this work. The views expressed are those of the authors and do not reflect the official policy or position of IBM or the IBM Quantum team. This article has supplementary downloadable material available at https://doi.org/10.1109/MS.2025.3569447, provided by the authors.

References

1. A. Chatterjee, K. Phalak, and S. Ghosh, "Quantum error correction for dummies," in *Proc. IEEE Int. Conf. Quantum Comput. Eng. (QCE)*, vol. 1, Piscataway, NJ, USA: IEEE Press, 2023, pp. 70–81, doi: 10.1109/QCE57702.2023.00017.
2. Y. Fujiwara, "Ability of stabilizer quantum error correction to protect itself from its own imperfection," *Phys. Rev. A*, vol. 90, no. 6, 2014, Art. no. 062304, doi: 10.1103/PhysRevA.90.062304.
3. Google Quantum AI, "Suppressing quantum errors by scaling a surface code logical qubit," *Nature*, vol. 614, no. 7949, pp. 676–681, 2023.
4. M. M. Hasan, M. M. Rahman, M. M. Ali, and P. Machado, "QuantoTrace: Quantum error correction as a service for robust quantum computing," in *Proc. 6th Int. Conf. Elect. Eng. Inf. Commun. Technol. (ICEE-ICT)*, Piscataway, NJ, USA: IEEE Press, 2024, pp. 616–621.
5. L. Postler et al., "Demonstration of fault-tolerant Steane quantum error correction," *PRX Quantum*, vol. 5, no. 3, 2024, Art. no. 030326, doi: 10.1103/PRXQuantum.5.030326.
6. H. Zhang et al., "Encoding error correction in an integrated photonic chip," *PRX Quantum*, vol. 4, no. 3, 2023, Art. no. 030340, doi: 10.1103/PRXQuantum.4.030340.
7. Q. Xu et al., "Constant-overhead fault-tolerant quantum computation with reconfigurable atom arrays," *Nature Phys.*, vol. 20, no. 7, pp. 1–7, 2024, doi: 10.1038/s41567-024-02479-z.
8. J. M. Renes, F. Dupuis, and R. Renner, "Efficient polar coding of quantum information," *Phys. Rev. Lett.*, vol. 109, no. 5, 2012, Art. no. 050504, doi: 10.1103/PhysRevLett.109.050504.
9. A. Goswami, M. Mhalla, and V. Savin, "Fault-tolerant preparation of quantum polar codes encoding one logical qubit," *Phys. Rev. A*, vol. 108, no. 4, 2023, Art. no. 042605, doi: 10.1103/PhysRevA.108.042605.
10. A. Goswami, M. Mhalla, and V. Savin, "Factory-based fault-tolerant preparation of quantum polar codes encoding one logical qubit," *Phys. Rev. A*, vol. 110, no. 1, 2024, Art. no. 012438, doi: 10.1103/PhysRevA.110.012438.
11. H. Kurniawan, L. Rodríguez-Soriano, D. Cuomo, C. G. Almudéver, and F. G. Herrero, "On the use of calibration data in error-aware compilation techniques for NISQ devices," in *Proc. IEEE Int. Conf. Quantum Comput. Eng. (QCE)*, Piscataway, NJ, USA: IEEE Press, 2024, pp. 338–348, doi: 10.1109/QCE60285.2024.00048.
12. A. Javadi-Abhari et al., "Quantum computing with Qiskit," 2024, *arXiv:2405.08810*.
13. G. Li, Y. Ding, and Y. Xie, "Tackling the qubit mapping problem for NISQ-era quantum devices," in *Proc. 24th Int. Conf. Architectural Support Program. Lang. Operating Syst.*, 2019, pp. 1001–1014, doi: 10.1145/3297858.3304023.
14. P. Murali, N. M. Linke, M. Martonosi, A. J. Abhari, N. H. Nguyen, and C. H. Alderete, "Full-stack, real-system quantum computer studies: Architectural comparisons and design insights," in *Proc. 46th Int. Symp. Comput. Archit.*, pp. 527–540, 2019.
15. P. Giard et al., "PolarBear: A 28-nm FD-SOI ASIC for decoding of polar codes," *IEEE J. Emerg. Sel. Topics Circuits Syst.*, vol. 7, no. 4, pp. 616–629, Dec. 2017, doi: 10.1109/JETCAS.2017.2745704.